

Nowość
EDYCJA 2024

NOWA Informatyka na czasie

2 PODRĘCZNIK • LICEUM • TECHNIKUM
ZAKRES PODSTAWOWY

ZMIANY
W PODSTAWACH
2024
✂️
podręcznik w pełni
dostosowany

nowa
era

NOWA

Informatyka na czasie

2 PODRĘCZNIK • LICEUM • TECHNIKUM

ZAKRES PODSTAWOWY

JANUSZ MAZUR

PAWEŁ PEREKIETKA

ZBIGNIEW TALAGA

JANUSZ S. WIERZBICKI

NOWA

Informatyka

na czasie

Podręcznik dopuszczony do użytku szkolnego przez ministra właściwego do spraw oświaty i wychowania i wpisany do wykazu podręczników przeznaczonych do kształcenia ogólnego do nauczania informatyki, na podstawie opinii rzeczoznawców:

mgr inż. Anny Koludo, dr. hab. Henryka Nogi, mgr Teresy Zawiszy-Chlebowskiej.

Etap edukacyjny: III

Typ szkoły: liceum ogólnokształcące i technikum

Rok dopuszczenia: 2025

Numer ewidencyjny w wykazie MEN: 1220/2/2025

Podręcznik przygotowano na podstawie *Programu nauczania informatyki w liceach i technikach*. „NOWA Informatyka na czasie”. Zakres podstawowy autorstwa Janusza Mazura.

Podręcznik został opracowany z uwzględnieniem zmian w zasadach polskiej ortografii, uchwalonych przez Radę Języka Polskiego w 2024 r. i obowiązujących od 01.01.2026 r.

Nabyta przez Ciebie publikacja jest dziełem twórcy i wydawcy. Prosimy o przestrzeganie praw, jakie im przysługują. Zawartość publikacji możesz udostępnić nieodpłatnie osobom bliskim lub osobiście znanym, ale nie umieszczaj jej w internecie. Jeśli cytujesz jej fragmenty, to nie zmieniaj ich treści i koniecznie zaznacz, czyje to dzieło. Możesz skopiować część publikacji jedynie na własny użytek.

Szanujemy cudzą własność i prawo. Więcej na www.legalnakultura.pl

Wykorzystywanie treści niniejszego Utworu, w jakiegokolwiek formie lub w jakimkolwiek procesie, do celów eksploracji tekstu i danych, uczenia maszynowego, rozwoju i/lub szkolenia i/lub wzbogacania sztucznej inteligencji jakiegokolwiek rodzaju jest surowo zabronione.



© Copyright by Nowa Era Sp. z o.o. 2025
ISBN 978-83-267-5277-3

Konsultacja merytoryczno-dydaktyczna: Bartosz Bieganowski.

Redakcja merytoryczna: Przemysław Szydzik, Krzysztof Fit, Anna Nasiadka.

Współpraca redakcyjna: Jacek Detlaf, Dorota Rzeszewska, Joanna Sawicka, Agnieszka Sieczak.

Redakcja językowa: Marta Zuchowicz, Agnieszka Szymanowska-Pancer, Justyna Dobrowolska.

Opieka graficzna: Maciej Galiński, Alicja Skupin-Podwojewska.

Projekt okładki: Maciej Galiński, Alicja Skupin-Podwojewska.

Projekt graficzny: Sławomir Włodarczyk.

Opracowanie graficzne: Alicja Skupin-Podwojewska, Maciej Galiński, Magdalena Kata, Marcin Oleksak, Sławomir Włodarczyk, Studio Graficzne Fastwork Grzegorz Afeltowicz, Ewa Kaletyn, Marcin Kołacz, Mazaki Studio Graficzne Beata Kostuch, Aleksandra Szpunar, Grażyna Truchlińska.

Ilustracje: Marcin Kołacz, Natalia Helman, Marcin Ptak, Ewa Sowulewska.

Realizacja projektu graficznego: Artur Polakowski.

Fotoedycja: Katarzyna Iwan-Malawska.

Nowa Era Sp. z o.o.

Aleje Jerozolimskie 146 D, 02-305 Warszawa

www.nowaera.pl, e-mail: nowaera@nowaera.pl, Centrum Kontakt: 58 721 48 00

Druk i oprawa: ArtDruk Kobyłka

Spis treści

▶ 1. Tworzenie treści internetowych	7
1. Tworzenie stron internetowych	8
1.1. Jak stworzyć stronę WWW?	8
1.2. Poznajemy język HTML	10
1.3. Podstawowe znaczniki języka HTML	13
1.4. Pozycjonowanie stron WWW	22
1.5. Kolory na stronach WWW	22
1.6. Kaskadowe arkusze stylów	23
1.7. Responsywność i dostępność stron WWW	26
2. Wizualne tworzenie stron internetowych	30
2.1. Na czym polega wizualne tworzenie treści?	30
2.2. Blokowe układanie treści na stronie	31
2.3. Edycja segmentów strony internetowej	38
2.4. Publikowanie witryny WWW	41
3. Sztuka publikowania w sieci	43
3.1. Jak zainteresować odbiorców swoimi publikacjami?	43
3.2. Teksty informacyjne	48
3.3. Grafika i identyfikacja wizualna	51
3.4. Nagrania audio i wideo oraz wystąpienia na żywo	56
3.5. Jak poznać (lepiej) swojego odbiorcę?	62
Z informatyką w przyszłość	64
▶ 2. Relacyjne bazy danych	65
4. Projektowanie bazy danych	66
4.1. Zastosowania baz danych	66
4.2. Tabele i relacje w bazie danych	67
5. Tworzenie bazy danych	75
5.1. Od projektu do bazy danych	75
5.2. Tabele w programie Access	75
5.3. Definiowanie relacji	81
5.4. Kwerendy – wyszukiwanie i łączenie danych	85
Z informatyką w przyszłość	88

▶ 3. Algorytmika i programowanie w języku Python	89
Przypomnij sobie	90
6. Od problemu do programu	91
6.1. Algorytm – precyzyjna instrukcja	91
6.2. Od algorytmu do programu komputerowego	93
6.3. Tworzymy własne programy użytkowe	95
7. Systemy liczbowe	103
7.1. Liczby dziesiętne i dwójkowe	103
7.2. Cyfrowy zapis informacji	105
7.3. Algorytmy zamiany reprezentacji liczb	109
7.4. Program zamieniający liczbę dziesiętną na binarną	111
8. Liczby pierwsze	117
8.1. Liczby złożone i liczby pierwsze	117
8.2. Podzielność liczb	121
8.3. Test pierwszości liczby	124
9. Sortowanie liczb i wyszukiwanie	128
9.1. Szukamy największej liczby	128
9.2. Do czego służy sortowanie danych?	133
9.3. Sortowanie a wyszukiwanie	137
9.4. Algorytmy sortowania	139
9.5. Sortowanie bąbelkowe	140
9.6. Sortowanie przez wstawianie	143
Z informatyką w przyszłość	148
▶ 4. Algorytmika i programowanie w języku C++	149
Przypomnij sobie	150
10. Od problemu do programu	151
10.1. Algorytm – precyzyjna instrukcja	151
10.2. Od algorytmu do programu komputerowego	153
10.3. Tworzymy własne programy użytkowe	155
11. Systemy liczbowe	164
11.1. Liczby dziesiętne i dwójkowe	164
11.2. Cyfrowy zapis informacji	166
11.3. Algorytmy zamiany reprezentacji liczb	170
11.4. Program zamieniający liczbę dziesiętną na binarną	172

12. Liczby pierwsze	179
12.1. Liczby złożone i liczby pierwsze	179
12.2. Podzielność liczb	183
12.3. Test pierwszości liczby	186
13. Sortowanie liczb i wyszukiwanie	190
13.1. Szukamy największej liczby	190
13.2. Do czego służy sortowanie danych?	195
13.3. Sortowanie a wyszukiwanie	199
13.4. Algorytmy sortowania	201
13.5. Sortowanie bąbelkowe	202
13.6. Sortowanie przez wstawianie	205
Z informatyką w przyszłość	210
■ Projekt zespołowy	211
Dodatki	213
1. Podstawowe znaczniki języka HTML	214
2. Typy danych i maski w programie Microsoft Access	216
3. Jak zaprojektować i utworzyć bazę danych?	217
4. Sposoby zapisu algorytmów	218
5. Logika matematyczna i operatory logiczne	220
6. Typy danych w języku Python	223
7. Podstawowe typy instrukcji w języku Python	224
8. Operatory i funkcje matematyczne w języku Python	227
9. Słowa kluczowe w języku Python	228
10. Typy danych w języku C++	229
11. Podstawowe typy instrukcji w języku C++	231
12. Operatory i funkcje matematyczne w języku C++	233
13. Słowa kluczowe w języku C++	234
14. Systemy liczbowe	235
Indeks	236

O czym jest podręcznik?

Podręcznik *NOWA Informatyka na czasie 2* pomoże ci utworzyć własną stronę internetową. Dowiesz się, do czego służą bazy danych, oraz zaprojektujesz i od podstaw zbudujesz taką bazę. Napiszesz również proste programy w języku Python lub C++.



Czemu służą poszczególne elementy podręcznika?

Ważne w tym temacie

Informacje o najważniejszych zagadnieniach poruszanych w danym temacie.

Zapamiętaj

Najważniejsze pojęcia omawiane na lekcji, które należy zapamiętać.

Ćwiczenie 1

Proste zadania w toku lekcji. Odpowiedzi nie należy zapisywać w podręczniku.

Dobra rada

Podpowiedzi ułatwiające wykonanie omawianych czynności.

Warto wiedzieć

Informacje poszerzające wiedzę o omawianych zagadnieniach.

A to ciekawe

Ciekawostki pokazujące omawiane zagadnienia z zaskakującej perspektywy.

Z informatyką w przyszłość

Przykłady zastosowania treści omawianych w danym dziale w przyszłym życiu zawodowym.

Oznaczenia w podręczniku

- ➔ Odsyłacz do zagadnienia omawianego szerzej w innym miejscu w podręczniku.
- 📌 Wskaźnik pojęcia; pojęcia obok wskaźników układają się w siatkę pojęć z danej lekcji.
- 🔗 Odsyłacz do omówienia danego zagadnienia poza podręcznikiem.
- *** Oznaczenie stopnia trudności zadań.

Zadania

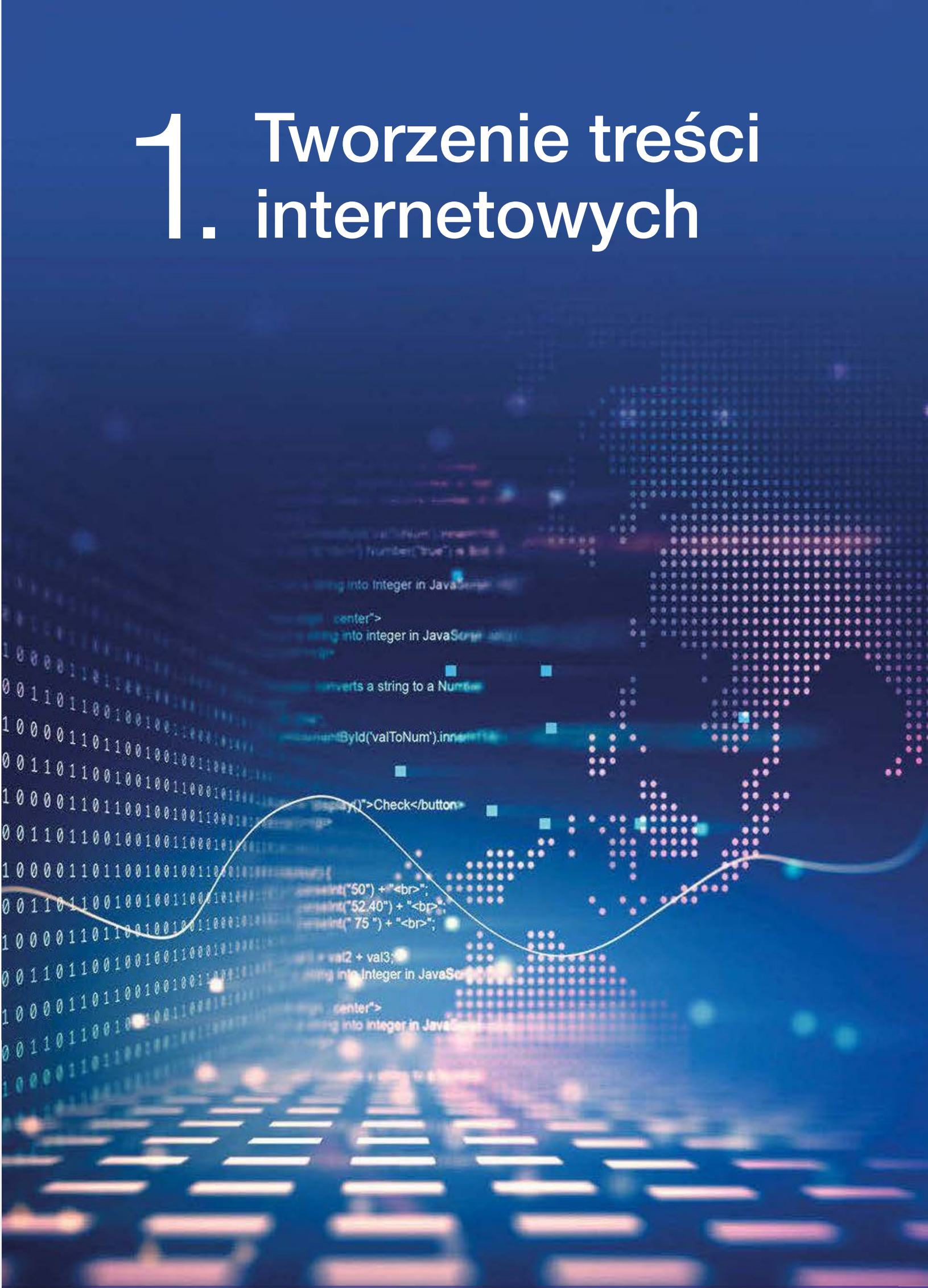
Zestaw poleceń o różnym stopniu trudności. Odpowiedzi nie należy zapisywać w podręczniku.

Projekt zespołowy

Zadanie do wykonania w grupie, pozwalające wykorzystać w praktyce wiedzę z kilku rozdziałów.

Dodatki

Przydatne informacje uzupełniające wiedzę z podręcznika.



1 Tworzenie stron internetowych

Codziennie odwiedzasz wiele stron internetowych. Nie musisz się za każdym razem zastanawiać, czy dana strona uruchomi się na urządzeniu, które właśnie obsługujesz. Nie musisz też sprawdzać wersji systemu operacyjnego, jak podczas pobierania programu. Wygodne, prawda? A przecież systemy operacyjne i urządzenia różnią się bardzo wyraźnie. Wygoda, o której na co dzień nie myślimy, wynika m.in. z tego, że strony internetowe są prostymi w swojej formie dokumentami tekstowymi. Warto nauczyć się je tworzyć.

Ważne w tym temacie

- Podstawowa struktura strony WWW
- Popularne znaczniki języka HTML
- Pozycjonowanie strony WWW
- Kaskadowe arkusze stylów
- Responsywność i dostępność stron WWW

1.1. Jak stworzyć stronę WWW?

Strona internetowa ➤ **Strona internetowa** to plik tekstowy, w którym za pomocą specjalnego języka opisano elementy strony i jej wygląd. Choć strony internetowe, jakie znamy dzisiaj, zupełnie nie przypominają tych sprzed lat, to nadal opierają się w dużej mierze na języku HTML, którego pierwsza dokumentacja powstała w 1991 r.

Język HTML ➤ **Język HTML** (ang. *HyperText Markup Language*) to język, który

Znacznik (tag) ➤ to podstawowy element języka HTML. Za pomocą znaczników opisuje się strony internetowe.

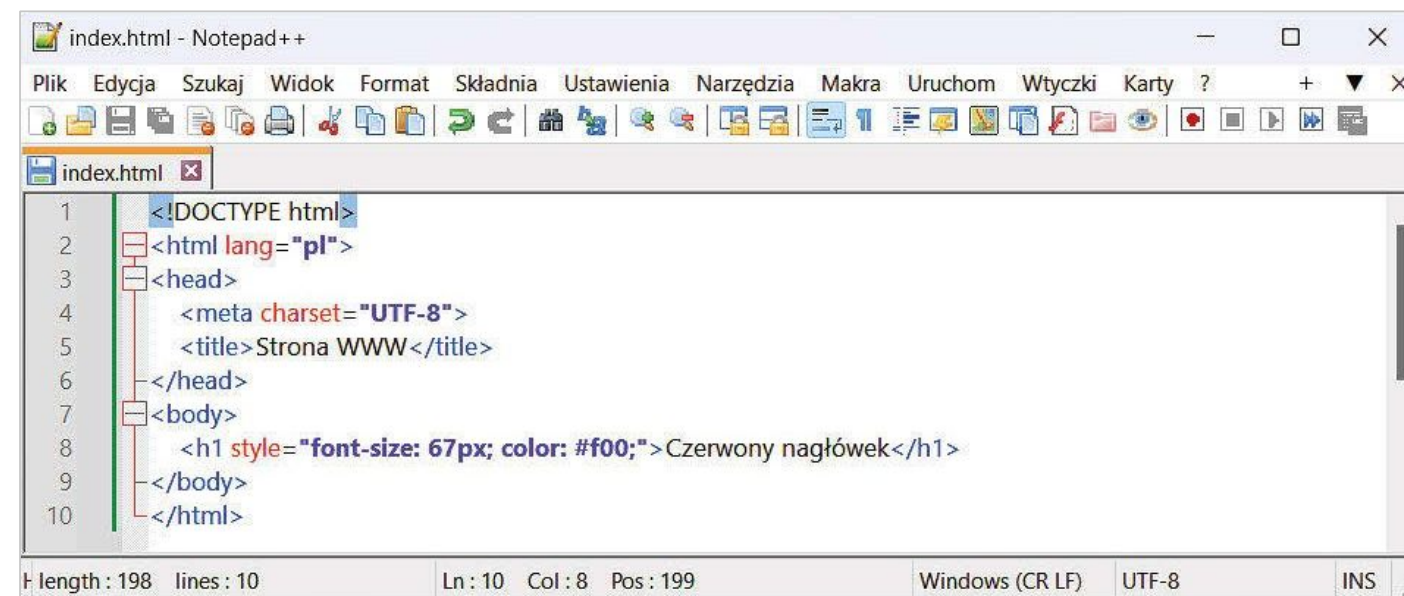
Składnia języka ➤ to w informatyce zasady budowania – z wyrażeni charakterystycznych dla danego języka – poleceń dla procesora lub programu.

Serwis internetowy ➤ **Serwis internetowy (witryna)** to strona internetowa, która składa się z wielu powiązanych ze sobą stron internetowych. Przechodzenie między nimi odbywa się za pomocą hiperłączy (linków).

Po wpisaniu przez użytkownika adresu strony w przeglądarce internetowej plik *index.html* jest pobierany z serwera i wyświetlany w oknie przeglądarki.

Narzędzia twórców stron internetowych

Oprócz znajomości języka HTML podczas pisania strony WWW będą nam potrzebne edytor tekstu i przeglądarka internetowa. Edytora tekstu użyjemy do zapisania strony w języku HTML. Wykorzystamy darmowy program Notepad++, który ułatwia pisanie kodu, m.in. dzięki kolorowaniu składni języka HTML (rys. 1.1).



Rys. 1.1. Fragment kodu strony internetowej w edytorze kolorującym składnię

Przeglądarka internetowa interpretuje zapisy z pliku HTML i wyświetla stronę w taki sposób, w jaki opisuje to plik HTML. W przeglądarce będziemy uruchamiać i testować tworzoną stronę.

Aby zapisać plik strony internetowej w Notatniku w systemie Windows, musimy zmienić typ pliku na **Wszystkie pliki** i po nazwie pliku wprowadzić rozszerzenie *html* lub *htm*. W edytorze Notepad++ wystarczy przed zapisaniem pliku wybrać **Składnia** → **H** → **HTML**. Wówczas program rozpozna składnię i będzie domyślnie zapisywał pliki z rozszerzeniem *html*.

Ćwiczenie 1

Utwórz katalog o nazwie *StronaWWW*. W dowolnym edytorze tekstu utwórz nowy plik i zapisz go pod nazwą *index.html* w katalogu *StronaWWW*. Sprawdź, czy rozszerzenie pliku jest poprawne.

Zapamiętaj

Strony internetowe są to pliki tekstowe z rozszerzeniem *html* lub *htm*, zapisane w języku HTML. Język ten opisuje zawartość i strukturę strony.

Dobra rada

Istnieje wiele darmowych narzędzi ułatwiających tworzenie stron internetowych, m.in. Atom, Brackets, gedit, Visual Studio Code.

Dobra rada

Za pomocą narzędzi administracyjnych wbudowanych w przeglądarki możesz badać zawartość stron internetowych. W Google Chrome otworzysz te narzędzia, naciskając klawisz F12.

Warto wiedzieć

Rozszerzenie *htm* stworzono w czasie, gdy popularny był system DOS obsługujący maksymalnie trzyliterowe rozszerzenia.

1.2. Poznajemy język HTML

Jak już wiesz, język HTML jest zbudowany ze znaczników (słów) oraz reguł składniowych (gramatyki), które pozwalają tworzyć zrozumiałe dla przeglądarki opisy strony internetowej. Poznasz teraz zasady tworzenia kodu stron internetowych.

Budowa znacznika w języku HTML

Znaczniki w języku HTML działają podobnie do nawiasów w matematyce: jeżeli otworzymy nawias, to musimy go zamknąć. Omówimy to na przykładzie jednego z najbardziej podstawowych znaczników języka HTML. Ten znacznik odpowiada za tworzenie hiperłączy (linków) do plików lub innych stron internetowych:

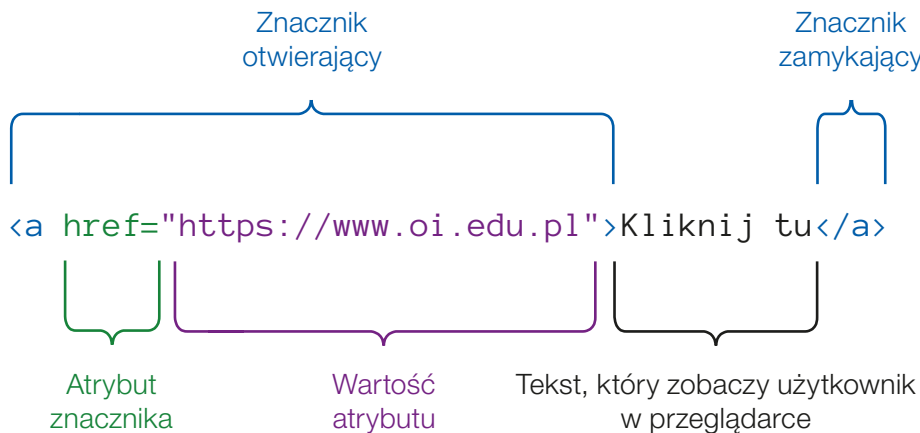
`<a>` – znacznik otwierający kod tworzenia hiperłącza,
`</a>` – znacznik zamykający kod tworzenia hiperłącza.

Znacznik otwierający i **Znacznik zamykający**

Atrybut znacznika

Poza pewnymi wyjątkami w języku HTML znaczniki występują w parach: **znacznik otwierający** i **znacznik zamykający** fragment kodu strony. Znaczniki zamykające zawierają dodatkowo znak slash / przed wyrażeniem określającym znacznik. Możliwości wielu znaczników możemy rozszerzać za pomocą **atrybutów** pozwalających dookreślić działanie znacznika. Na przykład znacznik tworzenia hiperłącza można rozbudować o atrybut, który będzie wymuszał wyświetlenie strony w nowej karcie przeglądarki. Atrybuty zapisujemy wewnątrz znacznika otwierającego, oddzielając je spacją. Wartości, które przypisujemy atrybutom, ujmujemy w cudzysłów i poprzedzamy znakiem `=`.

Kod, dzięki któremu utworzymy hiperłącze do strony www.oi.edu.pl, może wyglądać tak jak na rysunku 1.2.



Rys. 1.2. Kod tworzący hiperłącze w języku HTML

Znaczniki możemy zagnieżdżać. Oznacza to, że jeśli chcielibyśmy, aby tekst `Kliknij tu` z powyższego przykładu był dodatkowo pogrubiony, możemy umieścić służące do tego znaczniki `<b>` i `</b>` wewnątrz znaczników `<a>` i `</a>`.

Dobra rada
Wartości atrybutów w znacznikach zapisuj w cudzysłowie. Jest to zgodne z rekomendacją organizacji W3C, która opracowała standard języka HTML. Mimo to większość przeglądarek poprawnie zinterpretuje wartości wpisane bez cudzysłowu.

Dobra rada
Choć umieszczenie znaczników `<b>` i `</b>` wewnątrz znaczników `<a>` i `</a>` jest poprawne, lepiej będzie, jeśli w takim miejscu wykorzystasz atrybut `style` znacznika `<a>`: `style="font-weight: bold;"`.

Struktura strony WWW

Zawartość każdej strony WWW umieszczamy między znacznikami `<html>` i `</html>`. W strukturze strony możemy wyróżnić dwie podstawowe i rozłączne części: HEAD i BODY.

W **części HEAD** (między znacznikami `<head>` i `</head>`) zapisujemy informacje dla przeglądarek internetowych, np.:

- ▶ tytuł strony,
- ▶ słowa kluczowe dla wyszukiwarek,
- ▶ informacje o sposobie kodowania znaków,
- ▶ linki do zewnętrznych plików,
- ▶ style,
- ▶ skrypty.

Dane wprowadzone w części HEAD są niezwykle istotne, choć nie widać ich na stronie wyświetlanej przez przeglądarkę. Wyjątkiem jest tytuł, który jest widoczny na karcie przeglądarki.

Opis strony widoczny w wynikach wyszukiwania również umieszczamy w tej części, korzystając ze znacznika do określania metadanych. Podstawowymi znacznikami w części HEAD są znaczniki `<meta>` i `<title>`. Pierwszy z nich odpowiada za metadane strony internetowej, czyli informacje opisujące dokument HTML. Mogą to być dane o autorze strony czy sposobie kodowania znaków lub słowa kluczowe. Znacznik `<title>` odpowiada za tytuł strony, który pojawi się w wynikach wyszukiwania i na karcie przeglądarki internetowej. Warto zauważyć, że znacznik `<meta>` może wystąpić kilka razy, natomiast znacznik `<title>` umieszczamy w części HEAD tylko raz.

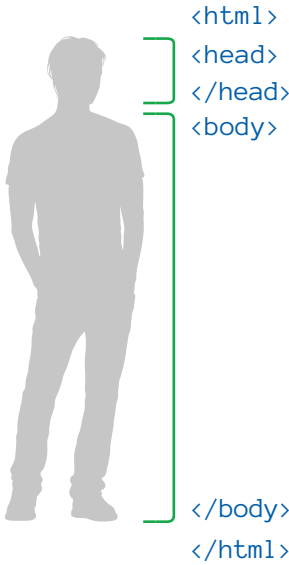
Część BODY określa zawartość strony, czyli zawiera wszystko to, co użytkownik widzi w przeglądarce, np.:

- ▶ teksty,
- ▶ zdjęcia,
- ▶ tabele,
- ▶ filmy,
- ▶ linki.

Podstawowa struktura strony internetowej, od której zaczynamy kodowanie, wygląda następująco:

```
1. <!DOCTYPE html>
2. <html lang="pl">
3. <head>
4.   <meta charset="UTF-8">
5.   <title>Tytuł strony WWW</title>
6. </head>
7. <body>
8. </body>
9. </html>
```

Część HEAD
Dobra rada
Strukturę kodu strony internetowej możesz sobie wyobrazić jako części ciała człowieka:



Warto wiedzieć

W podstawowej strukturze strony wykorzystuje się poniższe znaczniki i atrybuty:
meta (ang. *metadata* – metadane) – pozwala opisać zawartość strony,
lang (ang. *language* – język) – określa język strony,
charset (ang. *character set* – zestaw znaków) – określa sposób kodowania treści na stronie.

Zapis `<!DOCTYPE html>` w pierwszej linii informuje przeglądarkę o tym, że wyświetlany przez nią dokument to plik HTML. W drugiej linii wprowadzono znacznik `<html>`, który otwiera zawartość całej strony. Umieszczono w nim atrybut **lang** z wartością **pl**, co oznacza, że językiem strony będzie język polski. W części HEAD (linie 3.–6.) umieszczono znacznik `<meta>`, zawierający informacje o kodowaniu znaków, oraz znacznik `<title>`, który wyświetla tytuł strony WWW.

Zwróć uwagę na to, że znaczniki w tej części zostały wpisane z wcięciem. Jest to bardzo dobra praktyka, która ułatwia czytanie kodu i znajdowanie w nim błędów.

W części HEAD wprowadza się również metadane opisujące stronę, takie jak nazwisko autora, słowa kluczowe i opis strony. Wykorzystuje się do tego znacznik `<meta>` z atrybutami **name** i **content**:

```
1. <meta name="author" content="Jan Kowalski">
2. <meta name="keywords" content="cuda, świat,
   zwiedzanie">
3. <meta name="description" content="Strona
   o podróżach">
```

Część BODY na razie jest pusta – jeśli wpisemy tu dowolny tekst, wyświetli się on w przeglądarce. Całość kodu strony zamyka znacznik `</html>`. Przeglądarka internetowa interpretuje kolejne linie kodu zapisane w pliku i wyświetla jako stronę WWW.

Tworzenie strony internetowej

Tworzenie strony internetowej krok po kroku:

1. Otwórz dowolny edytor tekstowy.
2. Za pomocą znaczników języka HTML opisz strukturę oraz treść strony WWW.
3. Zapisz plik z rozszerzeniem *html* lub *htm*.
4. Otwórz plik w przeglądarce internetowej, aby zobaczyć utworzoną stronę WWW.

Dobra rada

Jeśli chcesz sprawdzić poprawność kodu HTML, użyj narzędzi do walidacji kodu, np. <https://validator.v3.org>.

Ćwiczenie 2

W pliku *index.html* utworzonym w ćwiczeniu 1. wpisz kod podstawowej struktury strony internetowej. W części BODY wpisz dowolny tekst. Następnie zapisz plik i otwórz go w przeglądarce.

Zapamiętaj

Strona WWW składa się z dwóch zasadniczych części: HEAD i BODY. Pierwsza odpowiada za opis strony, druga – za jej zawartość. Obie te części znajdują się między znacznikami `<html>` i `</html>` oznaczającymi początek i koniec strony internetowej.

1.3. Podstawowe znaczniki języka HTML

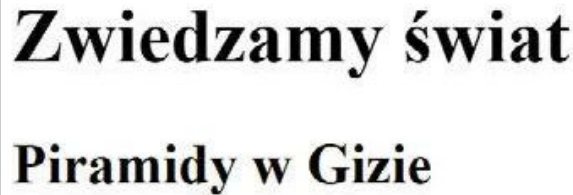
Wiemy już, że część BODY odpowiada za wyświetlanie treści na stronie. Poznamy teraz znaczniki, które służą do tworzenia treści i budowania powiązań między różnymi stronami.

Nagłówek

W języku HTML przewidziano 6 stopni nagłówków oznaczanych znacznikami od `<h1>` do `<h6>`. Im niższy numer nagłówka, tym większą czcionką jest zapisany jego tekst.

Tekst wyświetlany w formie nagłówka powinien się zawierać między znacznikiem otwierającym i zamykającym. Kod wyświetlający kolejne nagłówki widoczne na rysunku 1.3 ma postać:

```
1. <h1>Zwiedzamy świat</h1>
2. <h2>Piramidy w Gizie</h2>
```

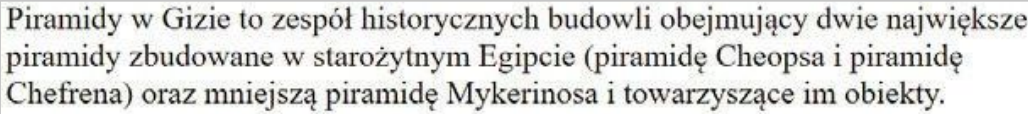


Rys. 1.3. Nagłówki pierwszego i drugiego stopnia

Akapit

Aby umieścić na stronie tekst i podzielić go na akapity, używamy znacznika `<p>`. Tekst powinien się znaleźć między znacznikami `<p>` i `</p>`. Kod wyświetlający akapit tekstu widoczny na rysunku 1.4 wygląda następująco:

```
<p>Piramidy w Gizie to zespół historycznych budowli
obejmujący dwie największe piramidy zbudowane
w starożytnym Egipcie (piramidę Cheopsa i piramidę
Chefrena) oraz mniejszą piramidę Mykerinosa
i towarzyszące im obiekty.</p>
```



Rys. 1.4. Akapit tekstu na stronie internetowej

Przydatnym atrybutem znacznika `<p>` jest **align**, którym można ustalić wyrównanie tekstu. Na przykład `<p align="center">` wyśrodkuje tekst akapitu względem bocznych krawędzi strony.

Warto wiedzieć

Nazwa znaczników od `<h1>` do `<h6>` pochodzi od angielskiego słowa *heading*, które oznacza nagłówek. Liczba reprezentuje stopień nagłówka.

Warto wiedzieć

Nazwa znacznika `<p>` pochodzi od angielskiego słowa *paragraph*, które oznacza akapit (paragraf).

Dobra rada

Atrybutu **align** możesz również używać w innych znacznikach, np. w nagłówkach.

Ćwiczenie 3

Otwórz do edycji plik *index.html* zapisany w ćwiczeniu 2. Usuń zawartość części BODY i wstaw do niej nagłówek pierwszego poziomu. Umieść w nim tytuł strony. Po nim, w nagłówku drugiego poziomu, wpisz *Piramidy w Gizie*. Dodaj opis piramid widoczny na poprzedniej stronie, wykorzystując znacznik `<p>`.

Warto wiedzieć

Nazwa znacznika `<br>` pochodzi od angielskiego wyrażenia *break line*, które oznacza złamanie wiersza i przejście do kolejnej linii.

Nie wszystkie znaczniki występują w parach (otwierający i zamykający). Podstawowym znacznikiem występującym pojedynczo jest znacznik pozwalający przełamać tekst (przenieść go do nowej linii): `<br>`. Znaczniki tego typu nazywamy pustymi. Jeden z takich znaczników – `<meta>` – wykorzystywaliśmy już przy tworzeniu podstawowej struktury strony internetowej.

Linia pozioma

Elementem poprawiającym czytelność strony internetowej jest pozioma linia, która może rozdzielać różne części strony (rys. 1.5). Linie poziomą możemy wstawić w kodzie za pomocą znacznika `<hr>`. Wygląd linii możemy dostosować, dodając do tego znacznika atrybuty `size` i `width`. Pierwszy z nich określa grubość linii, a drugi – jej długość.

Zwiedzamy świat

Piramidy w Gizie

Rys. 1.5. Zastosowanie linii poziomej do oddzielenia podtytułu od tytułu

Ćwiczenie 4

Uzupełnij część BODY pliku *index.html* z ćwiczenia 3. o poziomą linię oddzielającą tytuł strony od nazwy opisywanego miejsca. Obejrzyj wynik w przeglądarce.

Grafika

Grafikę na stronie internetowej możemy umieścić za pomocą znacznika `<img>`, który nie wymaga znacznika zamykającego. W atrybucie `src` podajemy ścieżkę dostępu do pliku graficznego ujętą w cudzysłów, np.:

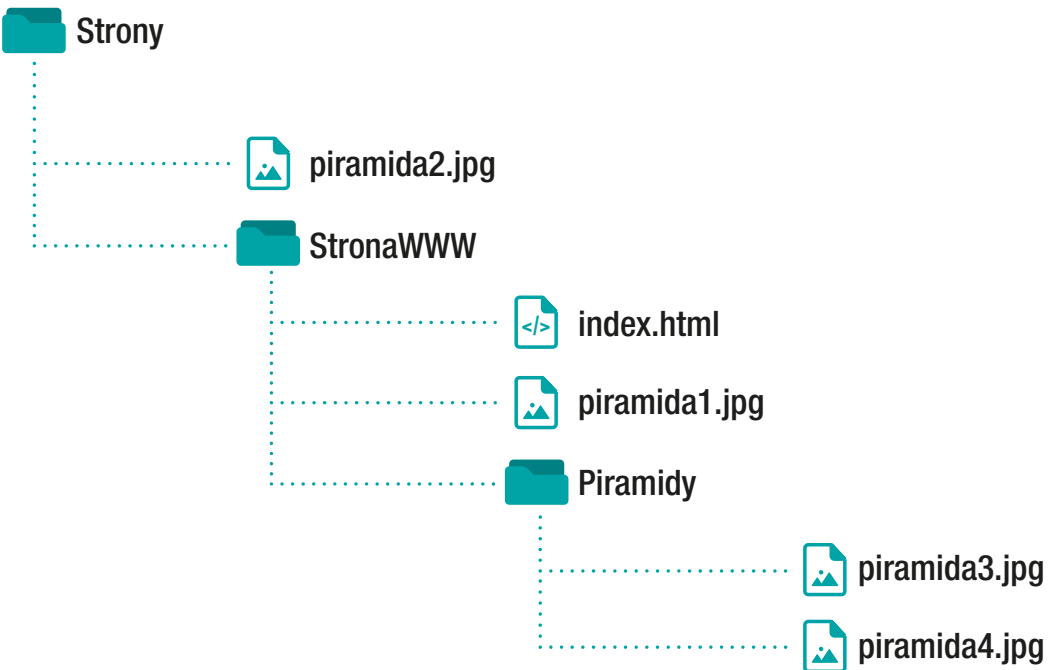
- 1. ``
- 2. ``
- 3. ``

Warto wiedzieć

Nazwa znacznika `<img>` pochodzi od angielskiego słowa *image*, czyli obraz. Natomiast nazwa atrybutu `src` pochodzi od angielskiego słowa *source*, które oznacza źródło.

Ścieżka dostępu do pliku (zwana inaczej adresem) może być względna lub bezwzględna. **Ścieżka względna** kieruje do pliku docelowego od miejsca, w którym znajduje się plik źródłowy. Przykładem jest kod z 2. linii. **Ścieżka bezwzględna** określa pełną ścieżkę od katalogu głównego (np. od litery określającej dysk twardy w komputerze). Przykładem adresowania bezwzględnego jest ścieżka `C:\Strony\StronaWWW\Piramidy\piramida3.jpg`.

Na rysunku 1.6 przedstawiono fragment struktury katalogów, w których umieszczono zdjęcia na stronę internetową. Kod z 1. linii zadziała tylko wtedy, gdy plik *piramida1.jpg* znajduje się na tym samym poziomie (w tym samym katalogu) co plik tworzonej strony (w naszym przypadku *index.html*). Zapis z 2. linii kodu będzie właściwy, jeśli plik *piramida2.jpg* jest umieszczony o jeden poziom wyżej niż plik tworzonej strony. Zapis `..\` oznacza przejście do katalogu położonego wyżej w strukturze. Kod z 3. linii zadziała, gdy plik *piramida3.jpg* znajduje się w katalogu *Piramidy*, który jest na tym samym poziomie co plik tworzonej strony.



Rys. 1.6. Przykład struktury katalogów

Znacznik `<img>` ma wiele atrybutów rozszerzających jego działanie. Może się zdarzyć, że z jakichś powodów przeglądarka nie wyświetli zdjęcia, np. gdy w atrybucie `src` podano adres URL pliku graficznego, a po pewnym czasie zmieni się nazwa pliku. Wtedy pomocny okaże się atrybut `alt`. Wartością tego atrybutu jest tekst alternatywny, który przeglądarka wyświetli zamiast grafiki. Dodatkowo wartość tego atrybutu jest odczytywana przez aplikacje pomagające osobom niewidomym zapoznać się z zawartością strony. Jest on też wykorzystywany przez roboty indeksujące zawartość strony na potrzeby wyszukiwarek internetowych.

Dobra rada

W kodzie najlepiej podawać względne ścieżki dostępu do plików. Dzięki temu, jeśli przeniesiesz stronę internetową w inne miejsce wraz z plikami i zachowasz strukturę katalogów, nie będzie trzeba aktualizować ścieżek.

Dobra rada

W nazwach plików unikaj stosowania polskich znaków, spacji i znaków specjalnych.

Warto wiedzieć

Nazwa atrybutu `alt` pochodzi od angielskiego słowa *alternative*, oznaczającego alternatywę.

Rozmiar obrazu na stronie internetowej ustawiamy za pomocą atrybutów `width` i `height`. Ich wartości podajemy w pikselach:

```

```

Dobra rada
Zawsze staraj się używać atrybutu `alt` podczas umieszczania grafiki na stronie internetowej.

Aby wyświetlane obrazy dopasowywały się do urządzenia, na którym przeglądamy strony WWW, w języku HTML wprowadzono znacznik `<picture>`. Wewnątrz tego znacznika (między elementami rozpoczynającym a kończącym) ustalamy, który z obrazów przeglądarka ma wyświetlać zależnie od szerokości ekranu. Służy do tego znacznik `<source>`. Za pomocą atrybutu `media` podajemy minimalną szerokość okna przeglądarki, dla której dany obrazek zostanie wyświetlony. Źródło obrazka wskazujemy, używając atrybutu `srcset`. Przeglądarka wyświetli ten obrazek z listy, który jako pierwszy spełnia kryterium szerokości.

```
1. <picture>
2.   <source media="(min-width: 1000px)"
   srcset="piramida1.jpg">
3.   <source media="(min-width: 600px)"
   srcset="piramida2.jpg">
4.   
5. </picture>
```

Na końcu listy zawsze należy umieścić znacznik `<img>` ze ścieżką dostępu do obrazka, który zostanie wyświetlony, jeżeli przeglądarka nie obsługuje znacznika `<picture>` lub jeśli żadne z kryteriów szerokości znacznika `<source>` nie jest spełnione.

W kodzie zapisano, że w przeglądarce o szerokości okna co najmniej 1000 pikseli albo co najmniej 600 pikseli wyświetli się odpowiednio *piramida1.jpg* albo *piramida2.jpg*. W przeglądarkach o szerokości okna mniejszej niż 600 pikseli lub nieobsługujących znacznika `<picture>` wyświetli się *piramida3.jpg*. Działanie powyższego kodu pokazano na rys. 1.7.



Rys. 1.7. Różne zdjęcia wyświetlane zależnie od wielkości ekranu

Dobra rada
Możesz zobaczyć, jak strona będzie się zachowywała na ekranach o różnych wielkościach, korzystając z narzędzi administracyjnych dostępnych w przeglądarce.

Zastosowanie znacznika `<picture>` jest przykładem zwiększenia **responsywności** strony WWW.

Strona responsywna
→ s. 27

Ćwiczenie 5
W katalogu *StronaWWW* utwórz katalog o nazwie *Piramidy*. Umieść w nim trzy zdjęcia przedstawiające piramidy w Gizie. Wstaw fotografie na swoją stronę za pomocą znacznika `<img>`.

Hiperłącza

Przemieszczanie się między stronami internetowymi jest łatwe dzięki hiperłączom zwanym również linkami lub odnośnikami.

Aby utworzyć hiperłącze do innej strony, korzystamy ze znaczników `<a>` i `</a>`. W atrybucie `href` znacznika otwierającego zapisujemy adres strony, do której ma prowadzić link. Adres powinien być zapisany w pełnej formie, tzn. musi zawierać informację o protokole (HTTP lub HTTPS). Między znacznikami wpisujemy treść, która będzie widoczna jako link w przeglądarce, np.:

```
<a href="https://www.oi.edu.pl/" target="_blank">
Strona organizatora Olimpiady Informatycznej </a>
```

Przeglądarka internetowa wyświetli w tym przypadku tekst „Strona organizatora Olimpiady Informatycznej”, zwykle domyślnie w kolorze niebieskim i podkreślony. Atrybut `target` z wartością `_blank` spowoduje, że strona, do której prowadzi hiperłącze, otworzy się w nowej karcie przeglądarki internetowej. Do znacznika `<a>` można też dodać atrybut `title`. Jego wartość zostanie wyświetlona po najechaniu na link kursorem myszy.

Warto wiedzieć
Nazwa atrybutu `href` pochodzi od angielskiego wyrażenia *hypertext reference*, które oznacza odnośnik hipertekstowy, czyli link.

A to ciekawe

Czy mogę tego użyć?

Twórcy witryn internetowych używają różnych technologii, nawet w obrębie jednej strony. Aby uniknąć niezgodności zastosowanej technologii z różnymi wersjami przeglądarek, mogą oni skorzystać z serwisu caniuse.com. Wystarczy w polu wyszukiwarki tego serwisu wpisać nazwę wybranej technologii, aby uzyskać informację o jej zgodności z poszczególnymi przeglądarkami. Na stronie caniuse.com twórcy stron mogą też znaleźć inne przydatne dla nich informacje, np. o popularności przeglądarek używanych przez internautów.



Ćwiczenie 6

Wyszukaj artykuł o piramidach w Gizie, np. w serwisie biura podróży. Umieść hiperłącze do tego artykułu na swojej stronie. Link powinien się otwierać w nowej karcie.

Dobra rada

Tworzenie kodu wielu podobnych elementów strony internetowej możesz znacznie przyspieszyć. Wystarczy, że skopiujesz i zmodyfikujesz już zapisane fragmenty kodu.

Poniżej opisano inne możliwości wykorzystania hiperłączy w języku HTML.

- ▶ Wstawienie pliku graficznego, którego kliknięcie będzie przenosić do innej strony WWW:

```
<a href="https://www.oi.edu.pl/"></a>
```

Między znacznikami <a> i wstawiono obraz z wykorzystaniem znacznika .

- ▶ Przenoszenie do wybranego fragmentu strony WWW:

```
<h2 id="Kontakt">Dane adresowe</h2>
<a href="#Kontakt">Idź do danych adresowych</a>
```

Dobra rada

Nazwa, którą nadajesz za pomocą atrybutu id, musi być niepowtarzalna. W przeciwnym razie nie można jednoznacznie określić, do którego fragmentu ma prowadzić link.

W powyższym przykładzie użyto atrybutu id do nazwania wybranego fragmentu strony (w tym przypadku – nagłówek drugiego stopnia). Do fragmentu odwołujemy się z wykorzystaniem znacznika <a> i atrybutu href. Jako wartość podajemy nazwę identyfikatora poprzedzoną znakiem #. Oba znaczniki powinny się pojawić w różnych, odległych od siebie miejscach. Wówczas będzie można zaobserwować przenoszenie do wskazanego fragmentu strony.

- ▶ Linkowanie do adresu e-mail:

```
<a href="mailto:olimpiada@oi.edu.pl">Wyślij wiadomość</a>
```

Wartość atrybutu href rozpoczyna się od zapisu mailto i znaku dwukropka, które oznaczają, że po nich zapisano adres e-mail. Po kliknięciu tak utworzonego linku uruchomi się domyślna aplikacja obsługująca pocztę elektroniczną na naszym urządzeniu. Jeśli mailto zastąpimy tel i wpiszęmy numer telefonu, uruchomi się aplikacja telefonu z wybranym numerem.

Tabele

Język HTML pozwala również na wyświetlanie danych w tabeli. Służą do tego znaczniki <table> i </table>. Wewnątrz tabeli wiersze definiujemy za pomocą znaczników <tr> i </tr>, a komórki w wierszach – przy użyciu znaczników <td> i </td>. Za pomocą znaczników <th> i </th> możemy wyróżnić nagłówek (pierwszy wiersz) tabeli. Nagłówek ten domyślnie będzie pogrubiony i wyśrodkowany.

Poniżej pokazano kod tabeli składającej się z nagłówka, trzech wierszy oraz dwóch kolumn.

```
1. <table border="1">
2.   <tr>
3.     <th>Nazwa piramidy</th>
4.     <th>Wysokość</th>
5.   </tr>
6.   <tr>
7.     <td>Cheopsa</td>
8.     <td>147 metrów</td>
9.   </tr>
10.  <tr>
11.    <td>Chefrena</td>
12.    <td>137 metrów</td>
13.  </tr>
14.  <tr>
15.    <td>Mykerinosa</td>
16.    <td>65 metrów</td>
17.  </tr>
18. </table>
```

Rysunek 1.8 przedstawia realizację tego kodu.

Nazwa piramidy	Wysokość
Cheopsa	147 metrów
Chefrena	137 metrów
Mykerinosa	65 metrów

Rys. 1.8. Tabela na stronie WWW

W zapisie użyto atrybutu border o wartości 1, odpowiadającego za grubość linii obramowania tabeli. Gdybyśmy nie użyli tego atrybutu, linie tabeli byłyby niewidoczne. Czasami korzysta się też z atrybutów cellpadding i cellspacing. Pierwszy z nich określa margines w komórce tabeli, czyli odległość od obramowania komórki do jej zawartości. Drugim możemy ustawić odległość między komórkami. W liniach kodu od 2. do 5. wprowadzono pierwszy wiersz tabeli. Jest to wiersz nagłówkowy, ponieważ użyto znaczników <th> i </th> w liniach 3. i 4.

W komórkach tabeli można umieszczać dane dowolnego typu. Oprócz tekstu mogą to być np. zdjęcia, listy, a nawet inne tabele.

Ćwiczenie 7

Wstaw na swojej stronie tabelę z informacjami o wysokościach piramid w Gizie.

Warto wiedzieć

Nazwa znacznika <tr> pochodzi od angielskiego wyrażenia table row, które oznacza wiersz tabeli.

Warto wiedzieć

Nazwa znacznika <th> pochodzi od angielskiego wyrażenia table heading, które oznacza nagłówek tabeli.

Warto wiedzieć

Nazwa znacznika <td> pochodzi od angielskiego wyrażenia table data, które oznacza dane tabeli.

Listy
→ s. 21

Struktura tabeli może być bardziej rozbudowana. Sąsiednie komórki można **scalać** zarówno w poziomie, jak i w pionie. Służą do tego atrybuty **colspan** (scalanie w poziomie) i **rowspan** (scalanie w pionie), których wartość wyraża liczbę scalonych sąsiednich komórek. Wróćmy do tabeli z piramidami. Załóżmy, że chcemy scalać komórki nagłówka tabeli i w scalonej komórce wpisać nazwę tabeli. Aby to zrobić, należy w pierwszym wierszu tabeli umieścić kod:

```
<th colspan="2">Piramidy w Gizie</th>
```

Dobra rada
Kodu tabeli w języku HTML nie czytaj liniowo (linia po linii). Postaraj się najpierw odszukać znaczniki tabeli `<table>` i `</table>`, a następnie znaczniki kolejnych wierszy oraz znaczniki kolejnych komórek w tych wierszach.

Wartość atrybutu **colspan** określa liczbę komórek scalanych w poziomie. Pełny zapis kodu HTML wygląda następująco:

```
1. <table border="1">
2.   <tr>
3.     <th colspan="2">Piramidy w Gizie</th>
4.   <tr>
5.     <th>Nazwa piramidy</th>
6.     <th>Wysokość</th>
7.   </tr>
8.   <tr>
9.     <td>Cheopsa</td>
10.    <td>147 metrów</td>
11.  </tr>
12.  <tr>
13.    <td>Chefrena</td>
14.    <td>137 metrów</td>
15.  </tr>
16.  <tr>
17.    <td>Mykerinosa</td>
18.    <td>65 metrów</td>
19.  </tr>
20. </table>
```

Warto wiedzieć
Duże serwisy internetowe składają się z wielu różnych plików, m.in. z rozszerzeniami *html*, *css*, *php*, *js*.

Rysunek 1.9 przedstawia działanie powyższego kodu.

Piramidy w Gizie	
Nazwa piramidy	Wysokość
Cheopsa	147 metrów
Chefrena	137 metrów
Mykerinosa	65 metrów

Rys. 1.9. Tabela ze scalonymi komórkami w pierwszym wierszu

Atrybuty **colspan** i **rowspan** możemy ze sobą łączyć, dzięki czemu tabele mogą być bardziej złożone.

Listy

W języku HTML, podobnie jak w edytorach tekstu, możemy skorzystać z listy uporządkowanej (numerowania) lub listy nieuporządkowanej (wypunktowania). Aby otrzymać listę numerowaną, używamy znacznika `<ol>`, natomiast listę wypunktowaną tworzymy za pomocą znacznika `<ul>`. W obydwu przypadkach za kolejne elementy listy odpowiada znacznik `<li>`.

Do znaczników `<ol>` i `<ul>` możemy dodać atrybut **type**. W przypadku list uporządkowanych określa on rodzaj numeracji (np. **I** – wielkie liczby rzymskie, **A** – wielkie litery). Dla list nieuporządkowanych ustawimy nim rodzaj punktora (np. **square** – kwadraty, **circle** – koła).

Poniższy kod przedstawia przykład listy uporządkowanej wraz z poprzedzającym ją nagłówkiem.

```
1. <h1>Ciekawe miejsca do odwiedzenia</h1>
2. <ol>
3.   <li>Wieża Eiffla w Paryżu</li>
4.   <li>Krzywa Wieża w Pizie</li>
5.   <li>Koloseum w Rzymie</li>
6.   <li>Łazienki Królewskie w Warszawie</li>
7.   <li>Machu Picchu w Peru</li>
8.   <li>Wielki Kanion w USA</li>
9. </ol>
```

Rysunek 1.10 przedstawia działanie powyższego kodu.

Ciekawe miejsca do odwiedzenia

- Wieża Eiffla w Paryżu
- Krzywa Wieża w Pizie
- Koloseum w Rzymie
- Łazienki Królewskie w Warszawie
- Machu Picchu w Peru
- Wielki Kanion w USA

Rys. 1.10. Lista uporządkowana

Ćwiczenie 8

Dodaj do swojej strony ciekawe miejsca, które twoim zdaniem warto odwiedzić. Użyj do tego listy.

Zapamiętaj

Stronę internetową tworzymy za pomocą znaczników języka HTML. Znaczniki mogą zawierać atrybuty z wartościami, które rozszerzają ich możliwości.

Warto wiedzieć
Nazwa znacznika `<ol>` pochodzi od angielskiego wyrażenia *ordered list*, które oznacza listę uporządkowaną.

Warto wiedzieć
Nazwa znacznika `<ul>` pochodzi od angielskiego wyrażenia *unordered list*, które oznacza listę nieuporządkowaną.

Warto wiedzieć
Nazwa znacznika `<li>` pochodzi od angielskiego wyrażenia *list item*, które oznacza element listy.

Dobra rada
Aby opisać fragment kodu komentarzem, którego nie będzie widać na stronie, zapisz jego treść między znacznikami `<!--` oraz `-->`.

1.4. Pozycjonowanie stron WWW

Niektóre znaczniki oraz atrybuty mają wpływ nie tylko na wygląd strony w przeglądarce, lecz także na wyniki prezentowane przez wyszukiwarki internetowe. Wyszukiwarki korzystają ze specjalnego oprogramowania, które przeszukuje strony internetowe m.in. po to, aby określić charakter strony, jakość jej treści oraz bezpieczeństwo.

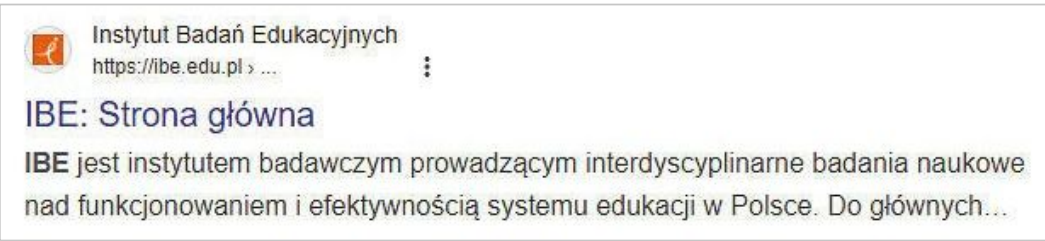
Indeksowanie • Proces analizowania treści strony nazywamy **indeksowaniem**. Dzięki zaindeksowaniu wielu stron internetowych wyszukiwarka w znacznie krótszym czasie może zwrócić wyniki wyszukiwania, szeregując adresy stron od najatrakcyjniejszych.

Wszelkie działania prowadzące do tego, aby zdobyć i utrzymać możliwie jak najwyższe miejsce w wynikach wyszukiwania, nazywamy

Pozycjonowanie • **pozycjonowaniem**.

Na kolejność w wynikach wyszukiwania pozytywnie wpływa wykorzystanie poniższych elementów:

- ▶ nagłówków od `<h1>` do `<h6>` – pomagają one w indeksowaniu struktury strony,
- ▶ znacznika `<title>` określającego wprost zawartość strony,
- ▶ znacznika `<meta>` z atrybutem `description` – wartość tego atrybutu pojawi się w wynikach wyszukiwania pod tytułem strony (rys. 1.11).



Rys. 1.11. Wynik wyszukiwania frazy „Instytut Badań Edukacyjnych”

Warto wiedzieć

Tekst alternatywny zapisany w atrybucie `alt` jest wykorzystywany przez specjalistyczne programy, które umożliwiają korzystanie ze stron internetowych osobom niewidomym i niedowidzącym.

Na pozycję w wynikach wyszukiwania obrazów ma wpływ również atrybut `alt` znacznika `<img>`. Wyszukiwarki wyżej oceniają te strony internetowe, które zawierają opisy grafik. Dzięki atrybutowi `alt` jest możliwe skuteczniejsze wyszukiwanie elementów graficznych w sieci. Warto więc go stosować i odpowiednio opisywać obrazy.

Podczas tworzenia stron internetowych należy także zwrócić uwagę na to, do jakich stron zewnętrznych prowadzą linki, które zamieszczamy. Jeżeli są to serwisy, które naruszają czyjeś prawa autorskie, to takie linki mogą obniżyć pozycję strony w wynikach wyszukiwania.

1.5. Kolory na stronach WWW

Model RGB • W języku HTML za pomocą kolorów możemy zmieniać wygląd wielu elementów. Do opisu koloru wykorzystuje się zazwyczaj **model RGB**. Każdy kolor w tym modelu jest mieszanką trzech barw podstawowych –

czerwonej (ang. *red*), zielonej (ang. *green*) i niebieskiej (ang. *blue*). Natężenie barw głównych reprezentują liczby w zakresie 0–255.

Kolor w modelu RGB możemy opisać na dwa sposoby:

- ▶ w **trybie RGB**, np. `rgb(124, 252, 0)`,
- ▶ w **trybie szesnastkowym**, np. `#7cfc00`.

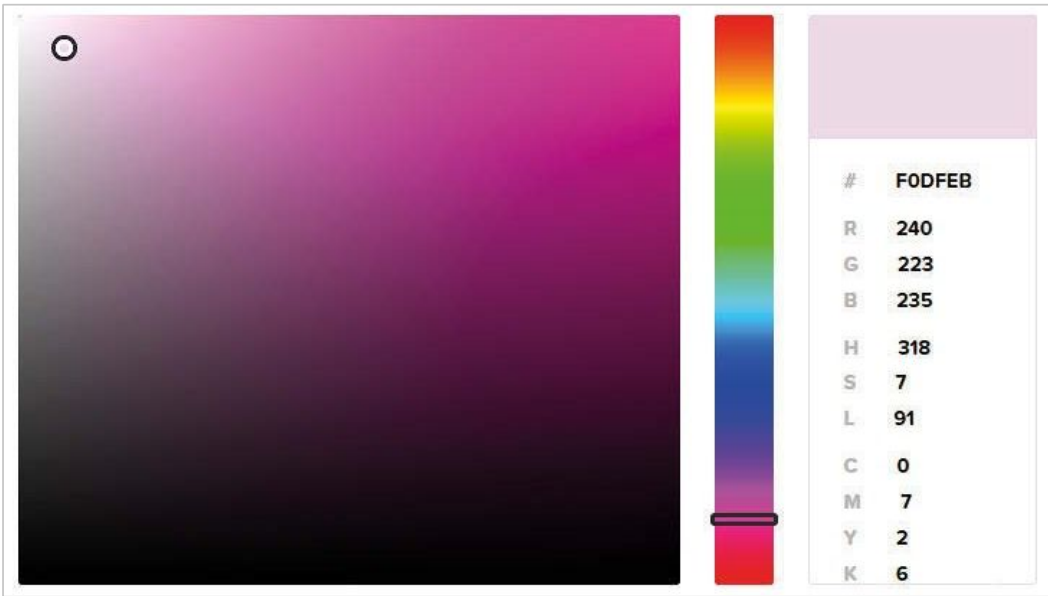
Oba tryby pozwalają zapisać dokładnie tę samą liczbę kolorów. Poniżej podano przykład zastosowania wartości kolorów w kodzie HTML. Wykorzystano w nim atrybut `style` znacznika `<p>`:

```
1. <p style="color:#7cfc00;">Szesnastkowy zapis koloru</p>
2. <p style="color:rgb(124, 252, 0);">Zapis koloru w RGB</p>
```

Tryb RGB
Tryb szesnastkowy to inaczej tryb heksadecymalny. Popularny jest również skrót HEX, od angielskiego słowa *hexadecimal*.

W praktyce często korzysta się z serwisów, które podają wprost wartości koloru w wybranym trybie oraz pozwalają na budowanie schematów kolorystycznych. Jednym z takich serwisów jest <https://htmlcolorcodes.com> (rys. 1.12).

Warto wiedzieć
Podstawowe kolory można określić w języku HTML również za pomocą ich nazw w języku angielskim, np. *black* dla czarnego lub *yellow* dla żółtego.



Rys. 1.12. Narzędzie do wybierania kolorów

1.6. Kaskadowe arkusze stylów

Do tej pory omawialiśmy znaczniki i ich atrybuty w oderwaniu od formy graficznej. Domyślne wartości znaczników i sposób ich działania możemy zmieniać za pomocą stylów. Język HTML dopuszcza różne możliwości określania stylu elementów strony.

Jednym z najpopularniejszych rozwiązań jest zastosowanie zewnętrznego pliku, w którym opiszemy wszystkie interesujące nas cechy elementów na stronie. Będziemy go nazywać kaskadowym arkuszem stylów, a językiem opisu będzie **język CSS** (ang. *Cascading Style Sheets* – kaskadowe arkusze stylów).

Warto wiedzieć
Style można określać również wprost w pliku HTML za pomocą atrybutu `style` wewnątrz znacznika, który chcemy zmodyfikować.

Język CSS

STYLE NA STRONIE INTERNETOWEJ

Dzięki stylom CSS możemy opisać wygląd strony WWW w jednym pliku i zastosować go do wszystkich stron naszej witryny. Taki plik zapisujemy z rozszerzeniem `css`.

Plik `index.html` opisuje strukturę i zawartość strony

```
1. <!DOCTYPE html>
2. <html lang="pl">
3. <head>
4. <meta charset="UTF-8">
5. <link rel="stylesheet" type="text/css" href="style.css">
6. <title>Siedem cudów świata</title>
7. </head>
8. </body>
9. <h1 class="tytul">Zwiedzamy świat</h1>
10. <hr>
11. <h2 class="tytul">Piramidy w Gizie</h2>
12. <p>Piramidy w Gizie to zespół historycznych budowli obejmujący dwie
    największe piramidy zbudowane w starożytnym Egipcie (piramidę Cheopsa
    i piramidę Chefrena) oraz mniejszą piramidę Mykerinosa i towarzyszące
    im obiekty.</p>
13. </body>
14. </html>
```

W sekcji nagłówkowej
umieszczamy znacznik
`link` łączący arkusz stylów
z dokumentem HTML.

`index.html`

Tak wygląda strona bez skorzystania
z arkusza stylów CSS.

Efekt połączenia obu plików obserwujemy poniżej na dwóch ekranach
o różnych rozmiarach.

Strona naturalnie dostosowuje się do rozmiaru ekranu, na którym
jest wyświetlana.

Plik `style.css` opisuje wygląd poszczególnych elementów strony

```
1. body {
2.     width: 70%;
3.     margin-left: 15%;
4.     background-image: url("tlo.jpg");
5.     background-size: 100%;
6.     font-family: Tahoma;
7.     color: #000000;
8. }
9. .title {
10.    text-align: center;
11.    color: #99680A;
12. }
13. p {
14.    text-align: justify;
15. }
```

`index.html` + `style.css`



Ćwiczenie 9

W folderze *StronyWWW* utwórz plik *style.css*. Dodaj do niego łącze w pliku *index.html*. Uzupełnij plik opisem stylów. Możesz skorzystać z infografiki na poprzednich stronach.

Zapamiętaj

Korzystanie z zewnętrznego arkusza stylów sprawia, że jakiegokolwiek zmiana w wyglądzie elementów strony WWW wymaga edycji tylko w jednym miejscu. Każda podstrona musi zawierać odniesienie do zewnętrznego arkusza stylów.

Generowanie stron internetowych z wykorzystaniem sztucznej inteligencji

Twórców stron internetowych wspomaga również rozwijająca się bardzo szybko sztuczna inteligencja (SI). Oferuje ona narzędzia i algorytmy, które znacznie usprawniają proces tworzenia. Dzięki SI jest możliwe automatyczne generowanie treści na stronach WWW, obrazów oraz gotowych kodów HTML i całych **witryn internetowych**. Sztuczna inteligencja może też optymalizować strony WWW, czyli poprawiać ich widoczność w wynikach wyszukiwania (SEO). Pomaga także analizować dane użytkowników oraz dostosować treść i układ strony do ich preferencji czy konkretnych działań.

Na rysunku 1.13 znajduje się przykładowa prośba (prompt) do systemu Microsoft Copilot o wygenerowanie kodu HTML strony o polskich lasach oraz fragment kodu wygenerowanego przez tę usługę. System umieścił w pliku HTML style, ale możemy go poprosić o przeniesienie ich do oddzielnego pliku CSS oraz o dołączenie odpowiedniego linku w dokumencie HTML.



Rys. 1.13. Prośba do systemu SI o wygenerowanie kodu HTML oraz fragment wygenerowanego kodu

1.7. Responsywność i dostępność stron WWW

Strony internetowe można wyświetlać na różnych urządzeniach – komputerach, tabletach, smartfonach. Trudno sobie wyobrazić, aby na każdym z nich wyglądały identycznie, skoro ekrany mają zupełnie inną wielkość. Wymusza to na twórcach stron WWW dostosowywanie ich do różnego rodzaju urządzeń.

Strony internetowe, które zoptymalizowano pod kątem wyświetlania na różnych urządzeniach, nazywamy **responsywnymi**. Na infografice ze s. 24–25 jest widoczny przykład tej samej strony wyświetlanej na dwóch urządzeniach wyposażonych w ekrany odmiennej wielkości. Możemy zaobserwować, jak zawartość strony dostosowuje się do wielkości ekranu.

- Zapewnianie stronie responsywności może polegać na:
- ▶ skalowaniu zdjęć w zależności od szerokości ekranu,
 - ▶ zastępowaniu zdjęć innymi zdjęciami wraz ze zmianą szerokości ekranu (rys. 1.7, s. 16),
 - ▶ zastępowaniu zdjęć tekstem je opisującym w sytuacji, gdy ekran jest zbyt mały, aby odpowiednio wyświetlić zdjęcie,
 - ▶ zmianie układu tekstu,
 - ▶ zmianie rozmiaru czcionki,
 - ▶ ustaleniu takiej wielkości linków, przycisków i menu, aby można było z nich korzystać, używając zarówno myszy, jak i palca na ekranach dotykowych.

Skalowanie obrazu dla ekranów o różnej wielkości można zrealizować z wykorzystaniem znacznika `<img>` i atrybutu `style` z wartością `max-width:100%` oraz automatycznie ustaloną wysokością (`height:auto`). Wówczas zdjęcie będzie mogło być tylko pomniejszone.

```

```

Innym sposobem jest użycie znacznika `<picture>` wraz z opracowanymi wcześniej zdjęciami w różnych rozmiarach, które mają się pojawić w zależności od wielkości ekranu.

Dobra rada

Jeśli chcesz sprawdzić, czy strona WWW jest responsywna, możesz skorzystać z linku <https://www.websiteplanet.com/pl/webtools/responsive-checker/>.

Witryna internetowa → s. 8

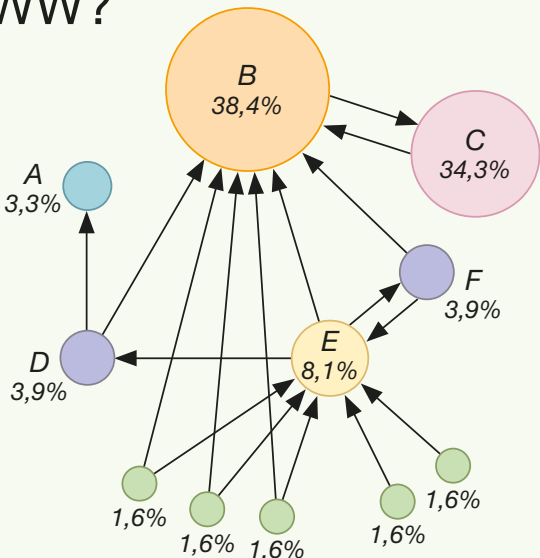
Dobra rada

Narzędzia takie jak Microsoft Copilot mogą ci pomóc w znalezieniu i poprawieniu błędów w kodzie twojej strony. Wystarczy, że wkleisz kod w oknie zapytania, a sztuczna inteligencja go przeanalizuje, wskaże miejsca do poprawienia i poprawi wadliwy kod.

A to ciekawe

Jak zmierzyć atrakcyjność strony WWW?

Wyszukiwarki internetowe kryją w sobie bardzo skomplikowane i częściowo niejawnie algorytmy klasyfikowania stron pod względem ich atrakcyjności. Jednym z najważniejszych algorytmów był algorytm PageRank, opracowany w 1996 r. przez Larry'ego Page'a i Sergeya Brina, twórców Google. Wykorzystywał on m.in. matematyczne twierdzenie Perrona-Frobeniusa z dziedziny algebry liniowej, udowodnione w 1912 r. Algorytm nadawał każdej stronie internetowej liczbową wartość, która reprezentowała jej ważność i atrakcyjność. Do dzisiaj jest on jedną z najpilniej strzeżonych tajemnic firmy Google.



Znacznik `<picture>` → s. 16

Aby automatycznie zmieniać rozmiar czcionki, wystarczy skorzystać z atrybutu `style`:

```
<h1 style="font-size:8vw">Stolice świata</h1>
```

Wielkość czcionki określono za pomocą specjalnej jednostki – `vw`, która oznacza procent szerokości okna przeglądarki. Drugą jednostką tego typu jest `vh`, czyli procent wysokości okna przeglądarki.

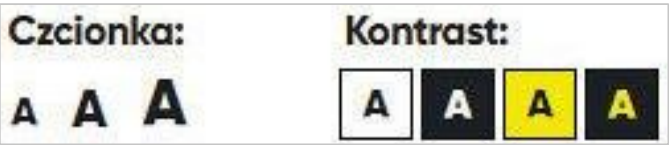
Ćwiczenie 10

Uzupełnij znaczniki użyte do wyświetlania akapitów na stronie o piramidach o atrybut `style`. Ustaw w nim skalowanie czcionki do 8 procent wysokości okna przeglądarki.

- Framework
- W tworzeniu stron responsywnych bardzo pomocne są **frameworki**, czyli zestawy gotowych komponentów do budowania stron internetowych. Przykładem takiego frameworka jest Bootstrap – zawiera on gotowe zestawy plików CSS, które twórca strony może modyfikować według własnych potrzeb. Szablon startowy Bootstrapa opiera się na
- Siatka
- siatce** (ang. *grid*), która płynnie dostosowuje się do wielkości ekranu, na którym jest wyświetlana strona internetowa.

Dostępność stron WWW

- Dostępność cyfrowa
- Standardem staje się dbanie o **dostępność cyfrową** stron internetowych, czyli ich dostosowywanie do możliwości różnych osób, w tym tych z zaburzeniami wzroku, słuchu czy ruchu. Instytucje publiczne są nawet do tego zobowiązane. Twórcy witryn budują je zgodnie z wytycznymi WCAG 2.1, które określają, jakie standardy powinna spełniać dostosowana strona.
- Witrynę, która realizuje wytyczne WCAG 2.1, można nazwać dostępną cyfrowo. Często spotykanym przykładem dostosowania są opcje zmiany wielkości tekstu na stronie oraz wyświetlania jej w kontrastowych kolorach (rys. 1.14).



Rys. 1.14. Opcje zmiany wielkości tekstu strony oraz jej kolorów

Dla zwiększenia dostępności strony powstały też dodatkowe znaczniki w języku HTML. Jednym z nich jest `<strong>`, który wizualnie działa tak jak znacznik `<b>`, czyli pogrubia tekst. Jego dodatkową rolą jest jednak nadanie znaczenia wyróżnionemu tekstowi – podkreśla on jego ważność i jest w ten sposób interpretowany przez urządzenia takie jak czytniki ekranu.

Warto wiedzieć

Innym przykładem znacznika wyróżniającego tekst jest znacznik `<em>`, który, podobnie jak znacznik `<i>`, pochyla tekst. Wskazuje również czytelnikom ekranu na ważność wyróżnionego w ten sposób tekstu.

Zadania

- 1 Odszukaj w Internecie tabelę prezentującą aktualne kursy walut. Utwórz plik *tabela.html* i sporządź w nim podobną tabelę dla pięciu wybranych walut.
- 2 Utwórz stronę internetową, która odwzoruje układ treści z pliku wskazanego przez nauczyciela (np. *przyroda.jpg*).
- 3 Utwórz plik *lista.html*. Wypisz w nim uporządkowaną listę linków do pięciu stron z gramii *online* oraz listę nieuporządkowaną z adresami trzech kanałów wideo.
- 4 Wykonaj stronę internetową, na której opiszesz swoje osiągnięcia. Każde z nich zapisz za pomocą nagłówka `<h3>` i krótko opisz, korzystając ze znacznika akapitu `<p>`.
- 5 Utwórz stronę WWW składającą się z trzech plików: *index.html*, *omnie.html* oraz *kontakt.html*. W każdym z tych plików napisz kilka zdań i wstaw listę nieuporządkowaną, w której aktywne będą linki tylko do pozostałych podstron.
- 6 Utwórz zewnętrzny szablon stylów i zapisz w nim właściwości dla części BODY: ciemnozielony kolor tła, biały kolor tekstu. Dla akapitu i nagłówka `<h4>` ustaw wyrównanie tekstu do środka, a dla elementu tworzącego tabelę – ramki w kolorze białym.
- 7 Zapisz kod tabeli o rozmiarach 5 × 5 bez widocznych krawędzi. Przygotuj obrazki o tej samej wielkości i umieść je w katalogu *Images*. W każdej komórce wstaw jeden obraz. Skorzystaj z atrybutów `width` i `height`.
- 8 Utwórz stronę *galeria.html* i umieść na niej pięć zdjęć stanowiących tematyczną galerię. Wszystkie zdjęcia powinny wyświetlać się w formie miniatur. Po kliknięciu w miniaturkę zdjęcie powinno otworzyć się w pełnym rozmiarze na nowej karcie przeglądarki.
- 9 Wykonaj stronę internetową, na której umieścisz ciekawe informacje o miejscu twojego zamieszkania. Dodaj również zdjęcia.
- 10 Stwórz stronę WWW z kilkoma przepisami na swoje ulubione potrawy. Każdy przepis powinien zawierać opis, zdjęcie, listę produktów oraz kolejne kroki do wykonania.
- 11 Za pomocą zewnętrznego arkusza stylów sformatuj plik, który otrzymasz od nauczyciela (np. *biuro_podrozy.html*), tak aby wyglądał w przeglądarce podobnie do rysunku, również przekanego przez nauczyciela (np. *biuro.jpg*).

2 Wizualne tworzenie stron internetowych

Znajomość języka HTML i kaskadowych arkuszy stylów CSS jest niezwykle istotna dla twórców stron internetowych oraz wszystkich osób, które chcą rozumieć, jak te strony działają. Czasami jednak potrzebujemy szybko stworzyć efektowną stronę i równie szybko umieścić ją w Internecie, ale nie chcemy szczegółowo znać jej działania. Z pomocą przychodzą wówczas narzędzia do wizualnego tworzenia stron internetowych.

Ważne w tym temacie

- Wizualne tworzenie treści
- Systemy CMS
- Blokowe układanie treści na stronie
- Publikowanie strony w Internecie

2.1. Na czym polega wizualne tworzenie treści?

Aby szybko utworzyć stronę internetową, można wykorzystać narzędzie do **wizualnego tworzenia treści**. Za jego pomocą buduje się strony z gotowych elementów, które można modyfikować według własnych potrzeb (rys. 2.1). Do pracy w takich narzędziach nie jest konieczna znajomość języków HTML i CSS.

Warto wiedzieć

Metodę pracy w narzędziach do wizualnego tworzenia treści określamy akronimem WYSIWYG (ang. *what you see is what you get* – to, co widzisz, jest tym, co dostajesz). Oznacza to, że strona, którą widzimy w edytorze, będzie tak samo wyglądała w przeglądarce.



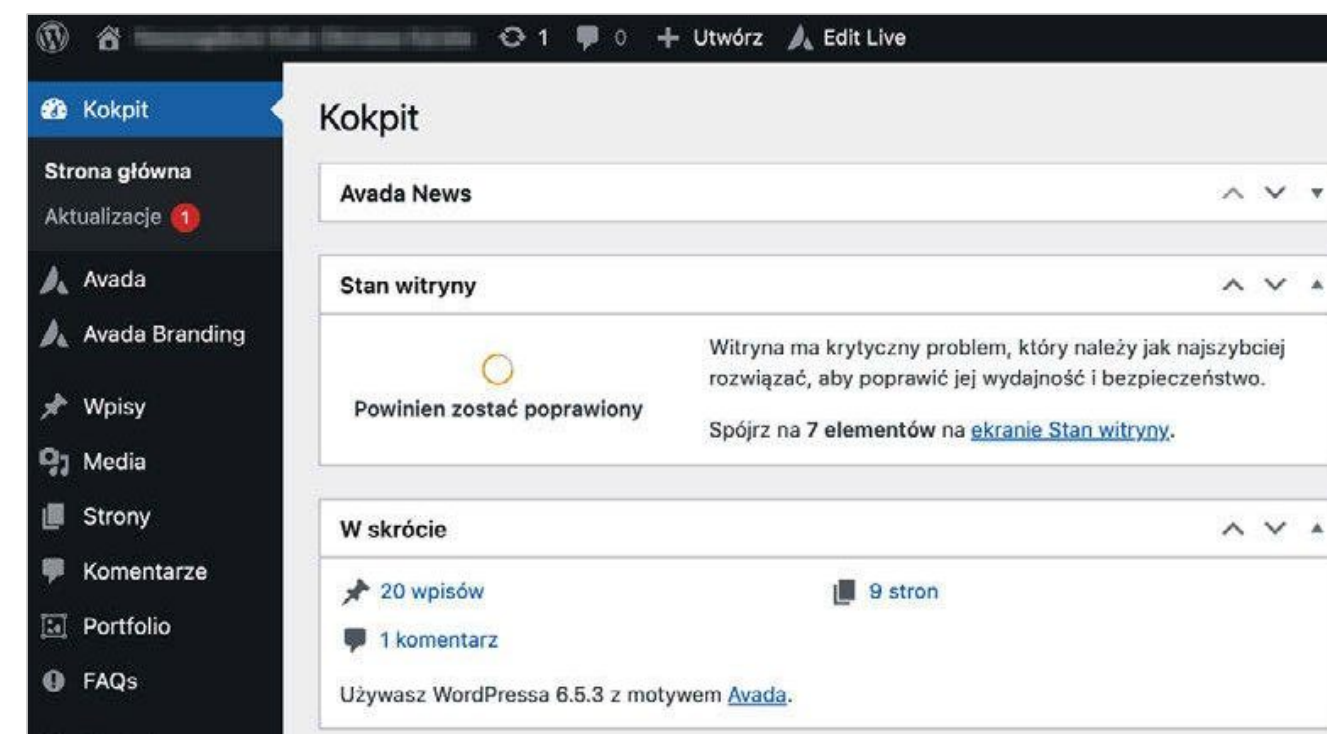
Rys. 2.1. Wizualne tworzenie treści z gotowych elementów

CMS (system zarządzania treścią)

Popularnymi narzędziami tego typu są systemy **CMS** (ang. *content management system*), czyli systemy zarządzania treścią. Ich cechą charakterystyczną jest oddzielenie zawartości strony od jej wyglądu, podobnie jak się to dzieje w przypadku języka HTML i kaskadowych arkuszy stylów w **języku CSS**. Ważną cechą jest również możliwość zarządzania tym, co znajduje się na stronie. Zawartość strony, czyli jej treść, jest zapisywana i przechowywana w bazie danych. Z kolei

Język CSS
→ s. 23

o wyglądzie decyduje szablon. Często można go wybrać spośród wielu udostępnionych przez twórców i dostosować do własnych wymagań. Zarówno treścią, jak i wyglądem strony możemy zarządzać z poziomu wygodnego graficznego interfejsu użytkownika. Do popularnych systemów CMS należą Joomla, Drupal oraz WordPress (rys. 2.2).



Rys. 2.2. Interfejs systemu CMS WordPress

Ćwiczenie 1

Odszukaj stronę internetową jednego z popularnych systemów CMS (np. WordPress). Przejrzyj dostępne szablony. Który z nich warto, według ciebie, wykorzystać do stworzenia strony internetowej prezentującej własne zainteresowania?

Zapamiętaj

Wizualne tworzenie treści polega na budowaniu strony internetowej z gotowych elementów. Można do tego wykorzystać systemy zarządzania treścią CMS oparte na bazach danych i szablonach połączonych za pomocą interfejsu użytkownika.

2.2. Blokowe układanie treści na stronie

W systemach zarządzania treścią wygląd strony projektuje się, korzystając z gotowych elementów i wybierając ich układ oraz rozmieszczenie. W dalszej części tematu omówimy pracę w darmowym programie Mobirise.

Dobra rada

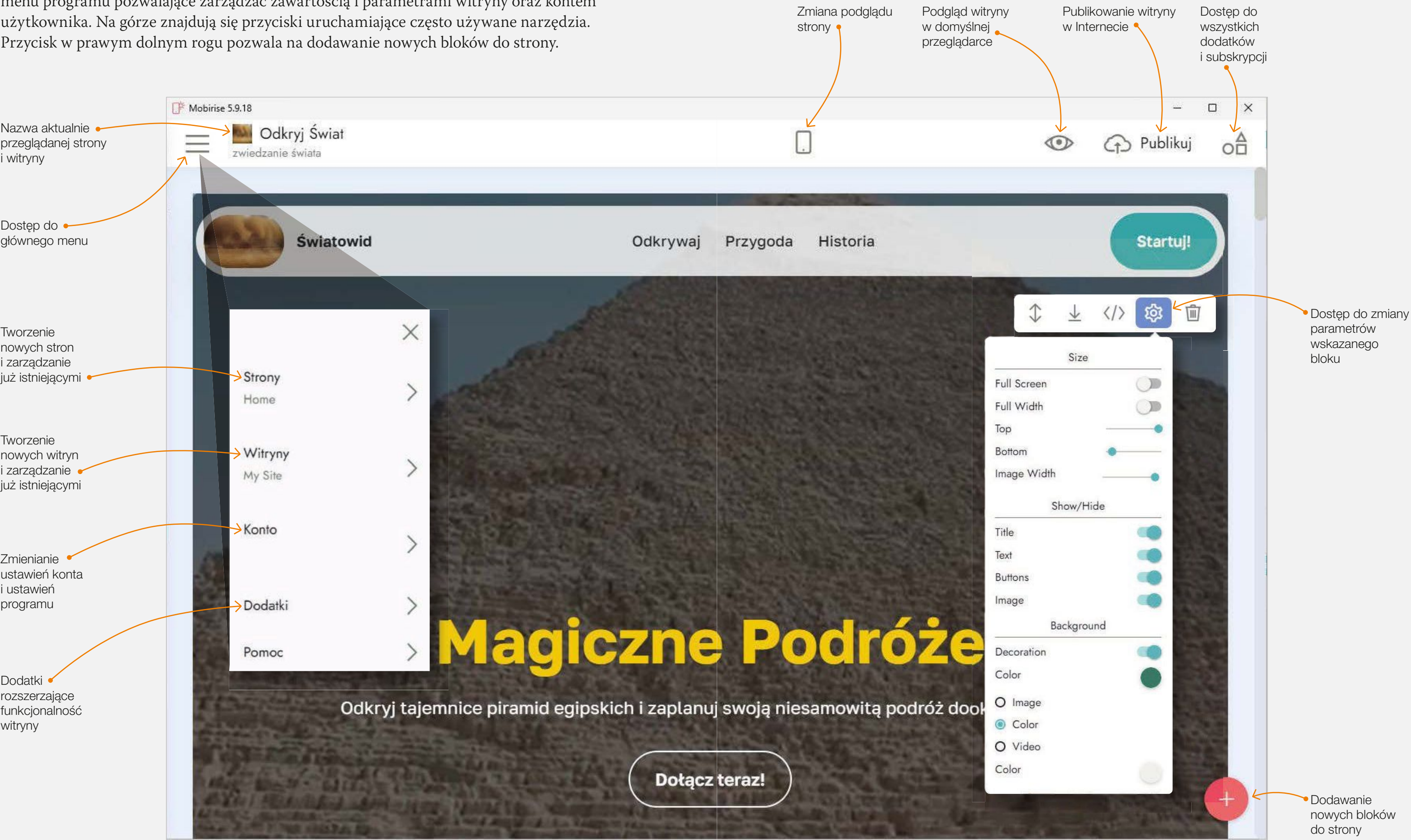
Jeśli myślisz o zostaniu twórcą stron internetowych, warto poznać popularne systemy CMS (w szczególności WordPress) i nauczyć się nimi zarządzać. Z pewnością spotkasz się z nimi w przyszłej pracy.

Warto wiedzieć

Mobirise nie jest systemem CMS w ścisłym znaczeniu, ponieważ nie ma jednej z głównych cech takiego systemu – możliwości przechowywania treści i ustawień strony w bazie danych.

OKNO PROGRAMU MOBIRISE

Główną część okna programu zajmuje projektowana strona. Po lewej można wyświetlić menu programu pozwalające zarządzać zawartością i parametrami witryny oraz kontem użytkownika. Na górze znajdują się przyciski uruchamiające często używane narzędzia. Przycisk w prawym dolnym rogu pozwala na dodawanie nowych bloków do strony.



Warto wiedzieć

Mobirise dostępny jest na komputery z system Windows oraz macOS.

Warto wiedzieć

Nie wszystkie nazwy narzędzi programu są spolszczone. Często opatrzone są jednak ikonami, które pomagają zrozumieć ich działanie.

Instalacja i pierwsze uruchomienie programu Mobirise

Aby zainstalować program na swoim komputerze, najpierw pobieramy instalator ze strony <https://mobirise.com>.

Po zainstalowaniu programu, by móc z niego korzystać, trzeba utworzyć konto za pomocą adresu e-mail. Można też skorzystać z innych możliwości autoryzacji: za pomocą konta Google lub Facebook. Podczas pierwszego uruchomienia programu możemy wybrać język, w którym będą wyświetlane opcje dostępne w programie (rys. 2.3).



Rys. 2.3. Logowanie do programu Mobirise i wybór języka interfejsu

Ćwiczenie 2

Zainstaluj na komputerze program Mobirise. Uruchom go, zaloguj się i zmień język na polski.

Tworzenie strony internetowej w programie Mobirise

W programie Mobirise stroną internetową nazywamy pojedynczy plik HTML. Kilka takich stron tworzy **witrynę WWW**. Po pierwszym uruchomieniu programu zobaczymy w jego głównym oknie przykładową stronę, a po kliknięciu przycisku w lewym górnym rogu uzyskamy dostęp do menu.

Witryna WWW
→ s. 8

A to ciekawe

Zasięg systemu WordPress

Najpopularniejszym systemem do zarządzania treścią stron internetowych jest obecnie WordPress. Szacuje się, że jest wykorzystywany w ponad 60% wszystkich witryn internetowych opartych na CMS oraz ponad 40% wszystkich stron internetowych. Korzystają z niego zarówno małe firmy, jak i duże globalne marki (np. Disney). Sam WordPress podaje, że użytkownicy tego systemu tworzą około 70 milionów postów każdego miesiąca.



Tworzenie strony internetowej w programie Mobirise krok po kroku: Tworzenie strony internetowej w programie Mobirise

1. Z głównego menu wybierz opcję **Strony**, a następnie **Utwórz nową stronę**.
2. Wybierz opcję **Dodaj pustą stronę (Add Blank Page)** i nadaj jej nazwę.
3. Dodaj wybrane segmenty do swojej strony.

W menu głównym mamy dostęp do pojedynczych stron utworzonych w programie oraz do całych witryn. Klikając **Witryny**, możemy też rozpocząć tworzenie nowej strony lub wspomóc się **generatorem wykorzystującym sztuczną inteligencję** (rys. 2.4).



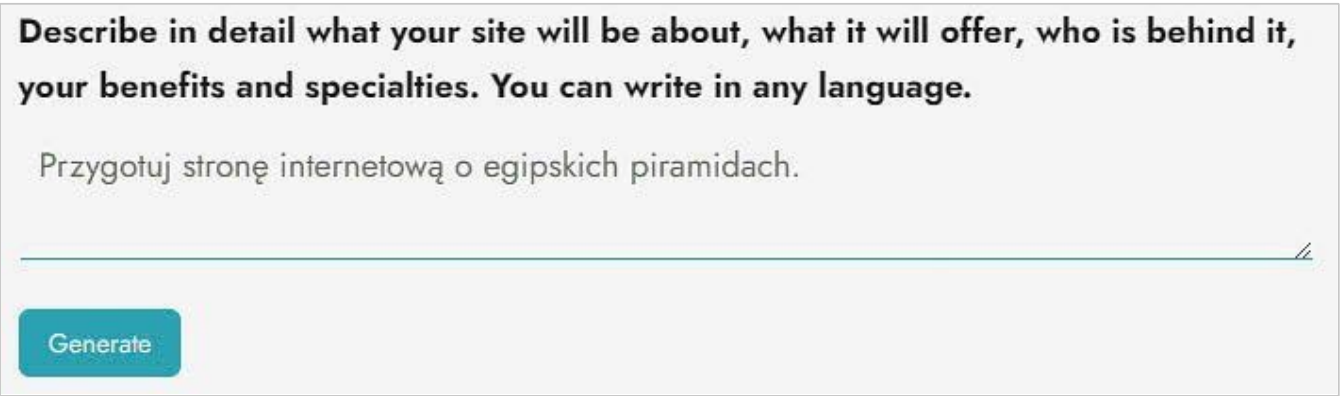
Rys. 2.4. Opcje dostępne w menu Witryny

Generowanie strony z wykorzystaniem sztucznej inteligencji

Po uruchomieniu generatora wykorzystującego sztuczną inteligencję w formularzu wpisujemy opis witryny, którą chcemy otrzymać. Formularz jest opisany po angielsku, jednak można w nim pisać również po polsku (rys. 2.5). Im dokładniej opiszemy stronę, którą chcemy uzyskać, tym bardziej zadowalający rezultat możemy otrzymać. Warto więc poświęcić trochę czasu na sformułowanie szczegółowego opisu wymagań.

Dobra rada

Pamiętaj, żeby zawsze dokładnie sprawdzić treść wygenerowaną przez sztuczną inteligencję. Może ona bowiem zawierać różnego rodzaju błędy.



Rys. 2.5. Generowanie strony za pomocą sztucznej inteligencji

Po wypełnieniu formularza i kliknięciu **Generate** otrzymamy w odpowiedzi witrynę według naszego opisu otwartą w Mobirise. Możemy od razu przejść do jej edycji (rys. 2.6).

 **Dobra rada**

Po wygenerowaniu strony przez sztuczną inteligencję efekt może nie być zadowalający. Możesz jednak dowolnie zmienić wygląd, układ i zawartość wszystkich bloków widocznych na stronie.

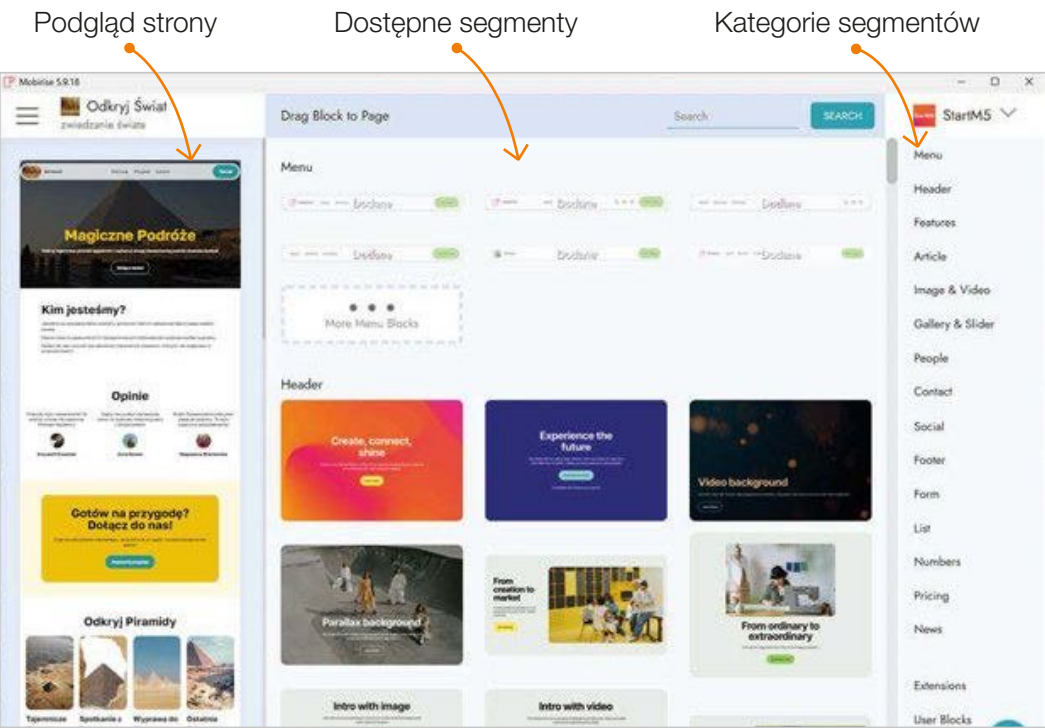


Rys. 2.6. Witryna o podróżach wygenerowana przez sztuczną inteligencję

W prawej dolnej części okna znajduje się przycisk służący do dodawania **segmentów**, czyli elementów, z których buduje się stronę. Kiedy klikniemy ten przycisk, okno aplikacji podzieli się na trzy części. W środku znajdują się segmenty, które można przeciągać do lewej części prezentującej podgląd strony. Segmenty można wyszukiwać, korzystając z kategorii w prawej części okna aplikacji (rys. 2.7).

 **Dobra rada**

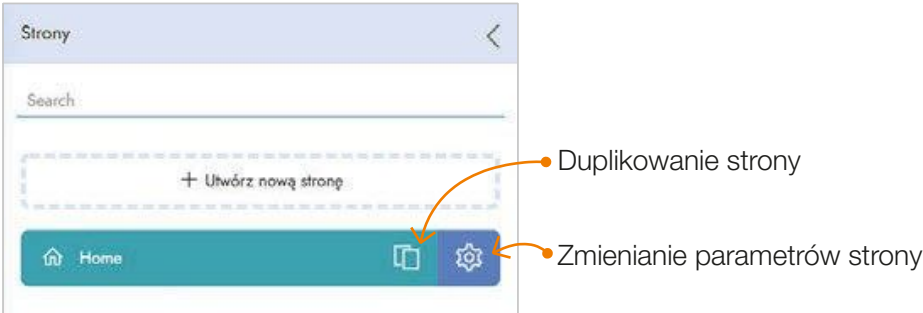
Jeśli na swojej witrynie planujesz wielokrotnie korzystać z opracowanego przez siebie segmentu, warto dodać go do **Segmentów Użytkownika**. W tym celu kliknij ikonę strzałki skierowanej w dół w prawym górnym rogu wybranego segmentu.



Rys. 2.7. Dodawanie segmentu do strony

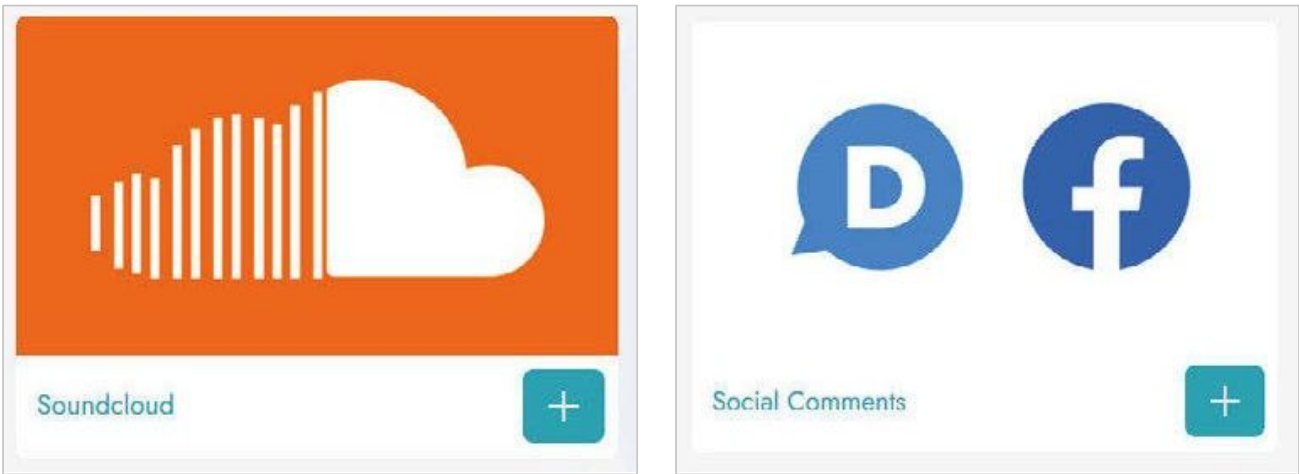
Po dodaniu wybranych segmentów do strony możemy zamknąć ten widok przyciskiem **Zamknij** znajdującym się w prawym dolnym rogu lub kliknąć podgląd z lewej strony.

Utworzona strona znajdzie się na liście w menu **Strony**. Możemy ją powielić jako wzorzec dla kolejnych stron oraz ustawić jej podstawowe parametry ważne np. z punktu widzenia pozycjonowania, m.in. tytuł, opis, nazwę pliku HTML (rys. 2.8).



Rys. 2.8. Duplikowanie strony i zmienianie jej parametrów

Z głównego menu, po kliknięciu opcji **Konto**, uzyskujemy również dostęp do ustawień programu. Możemy też zobaczyć, jakie dodatki rozszerzające możliwości tworzonych stron internetowych oferuje nam program. Większość z nich jest płatna, ale są również darmowe (rys. 2.9).



Rys. 2.9. Przykładowe dodatki dostępne w programie Mobirise

Ćwiczenie 3

Zbuduj nową stronę w systemie do wizualnego tworzenia stron. Nadaj jej nazwę. Dodaj trzy wybrane przez siebie bloki.

 **Zapamiętaj**

Budowanie strony w serwisie do wizualnego tworzenia treści polega na układaniu jej z gotowych bloków, które można następnie edytować.

 **Warto wiedzieć**

Gdy klikniemy opcję zmieniania parametrów strony, możemy zobaczyć, jak będzie wyglądała nasza strona w wynikach wyszukiwania Google.

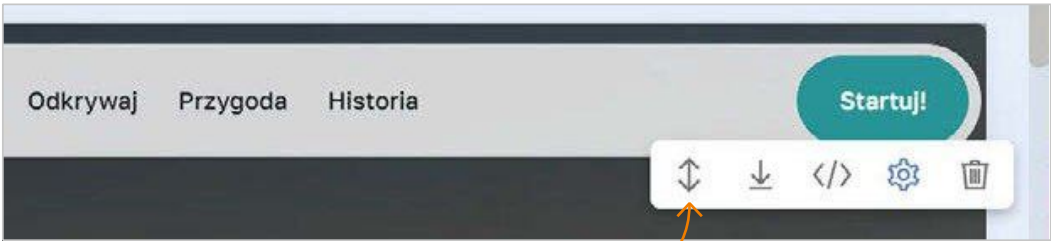
 **Warto wiedzieć**

W programie Mobirise można wykupić roczną subskrypcję, która pozwala na niemal nieograniczone wykorzystywanie wszystkich dostępnych dodatków.

2.3. Edycja segmentów strony internetowej

Dobra rada
Program Mobirise jest połączony z wyszukiwarką zdjęć, z których możesz skorzystać w swojej witrynie. Dzięki temu nie musisz ich szukać w innych programach i bardzo łatwo dodasz je do swojego projektu.

Gdy dodajemy do strony nowe segmenty, układają się one jeden pod drugim. Jeśli chcemy zmienić ich kolejność, musimy w oknie edycji strony użyć przycisku **Przesuń segment w górę/dół** – pojawi się on po najechaniu kursorem myszy na dany segment (rys. 2.10). Ten przycisk nie pojawia się dla segmentów z kategorii **Menu**, ponieważ menu witryny zawsze będzie u góry.



Rys. 2.10. Zmiana kolejności bloku na stronie

Zmiana zawartości segmentu

Zmiana zawartości segmentu krok po kroku:

- 1. Kliknij segment, którego zawartość chcesz edytować.
- 2. Zamień tekst umieszczony w bloku na własny.
- 3. Zmień właściwości segmentu według własnych wymagań.

Dobra rada
Różne segmenty mają wiele cech wspólnych ustawianych w ten sam sposób, dlatego używając ich, bardzo szybko opanujesz działanie większości parametrów.

Pozostałe przyciski ustawień segmentu pozwalają kolejno na zapisanie segmentu jako własnego (do ponownego wykorzystania), zmianę właściwości segmentu (rys. 2.11) oraz usunięcie go ze strony.

Właściwości segmentu mogą być różne w zależności od tego, jaki typ segmentu edytujemy. Możemy ustawić m.in.:

- ▶ rozmiar segmentu,
- ▶ marginesy górne i dolne,
- ▶ wyrównanie zawartości,
- ▶ widoczność elementów tekstowych czy przycisków,
- ▶ tło (zdjęcie, kolor, jednolitą kolorową nakładkę na zdjęcie z efektem przezroczystości),
- ▶ efekt paralaksy (nachodzenia kolejnych segmentów na tło przy przewijaniu strony).

Wszystkie elementy, z których składa się dany segment, można edytować. Możemy dowolnie zmieniać teksty, ich kolor, wyrównanie, styl (jeden ze zdefiniowanych), dodawać linki do zaznaczonych fragmentów, dodawać, włączać lub wyłączać widoczność przycisków czy ustawiać ich styl.



Rys. 2.11. Zmianie właściwości segmentu

Ćwiczenie 4

Zmień zawartość bloków dodanych do strony w ćwiczeniu 3. Wpisz wybrany przez siebie tekst, zmień jego wyrównanie, kolor oraz parametry czcionki.

Dodawanie hiperłączy do wybranych elementów strony

Wiemy z poprzedniego tematu, jak istotnym elementem na stronach internetowych są **hiperłącza** (linki). Pozwalają przenieść się niemal do dowolnego miejsca w Internecie. W programie Mobirise hiperłącza można dodawać do dowolnych elementów segmentu: tekstów, zdjęć, przycisków.

Tworzenie linku krok po kroku:

- 1. Zaznacz tekst (lub kliknij w element), który ma być linkiem.
- 2. Ustaw link poprzez wpisanie adresu zewnętrznego strony WWW lub wybór jednej ze stron witryny albo segmentu aktualnej.
- 3. Zatwierdź wybór.

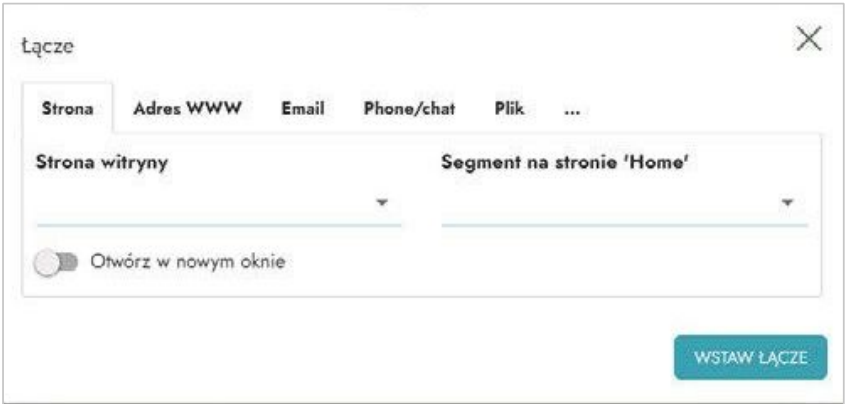
Po zaznaczeniu dowolnego tekstu lub kliknięciu przycisku pojawi się niewielki pasek narzędzi. Wybranie z niego ikony zaznaczonej na rys. 2.12 pozwoli na wstawienie łącza i jego edycję.



Rys. 2.12. Wstawianie łącza na stronie internetowej

Hiperłącza mogą przekierowywać w różne miejsca (rys. 2.13):

- ▶ na strony naszej witryny lub ich wybrane fragmenty (karta **Strona**),
- ▶ w dowolne miejsca w Internecie (karta **Adres WWW**),
- ▶ na adres poczty elektronicznej (karta **E-mail**),
- ▶ na numer telefonu (karta **Phone/chat**),
- ▶ do plików do pobrania (karta **Plik**).



Rys. 2.13. Edycja linku

Hiperłącza
→ s. 17

Tworzenie linku

Dobra rada
Jeśli chcesz, aby obrazek był linkiem, kliknij go i uzupełnij pole **Łącze obrazka** lub kliknij przycisk **Edytuj łącze**.

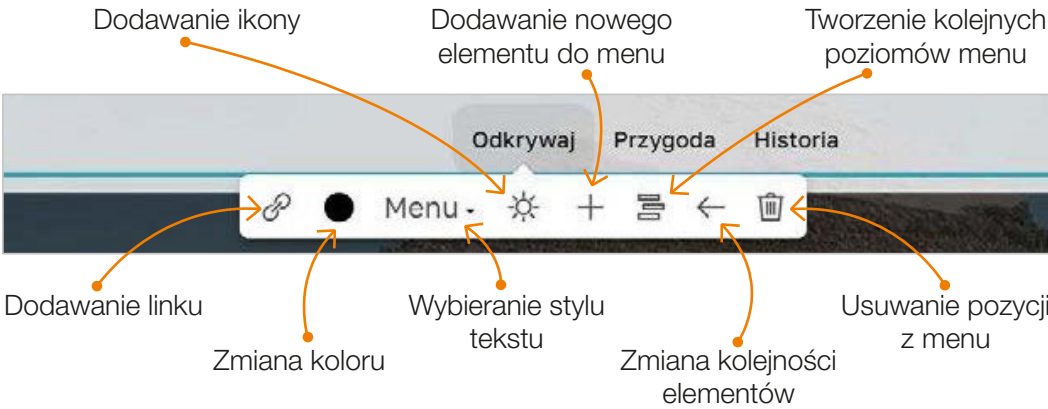
Dobra rada
Hiperłącza do fragmentów strony możesz wykorzystać, tworząc na początku długiego artykułu spis treści. Każdy element tego spisu może być linkiem do fragmentu strony, który opisuje.

Ćwiczenie 5

Zaznacz fragment tekstu na stronie z ćwiczenia 4. i dodaj do niego link do wybranej strony internetowej.

Edycja menu strony

Segmenty z kategorii **Menu** mają nieco inne właściwości niż pozostałe. Są umieszczane na górze strony i powtarzają się na każdej z nich. To oznacza, że gdy dodamy taki segment do pierwszej strony, automatycznie będzie on dodawany do kolejnych w obrębie witryny (rys. 2.14).



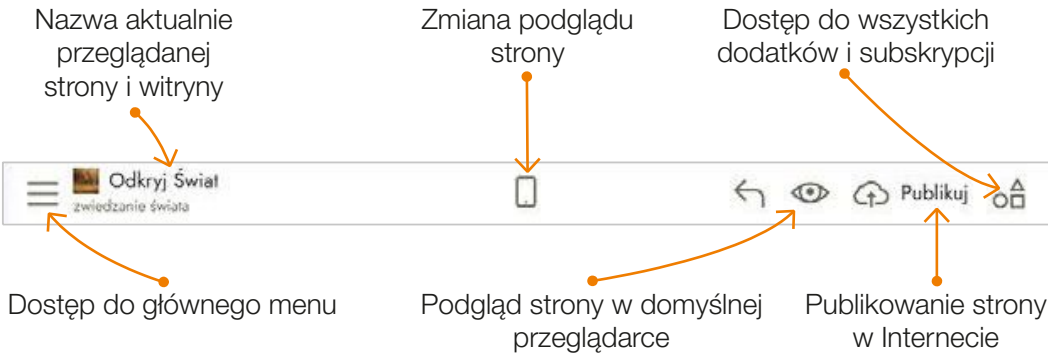
Rys. 2.14. Edycja menu strony

Ćwiczenie 6

Zmień opcje dostępne w menu swojej witryny tak, aby odpowiadały zawartości, którą planujesz w niej umieścić.

Górny pasek narzędzi

W środkowej części górnego paska mamy możliwość przełączenia widoku projektu pomiędzy smartfonem, tabletem o orientacji pionowej lub poziomej a monitorem. Dzięki temu od razu w trakcie tworzenia stron WWW możemy zobaczyć, jak wyświetlają je różne urządzenia. Z prawej strony górnego menu, za pomocą ikony oka możemy wyświetlić projekt w domyślnej przeglądarce internetowej. Przycisk **Publikuj** służy do udostępniania projektu w Internecie (rys. 2.15).



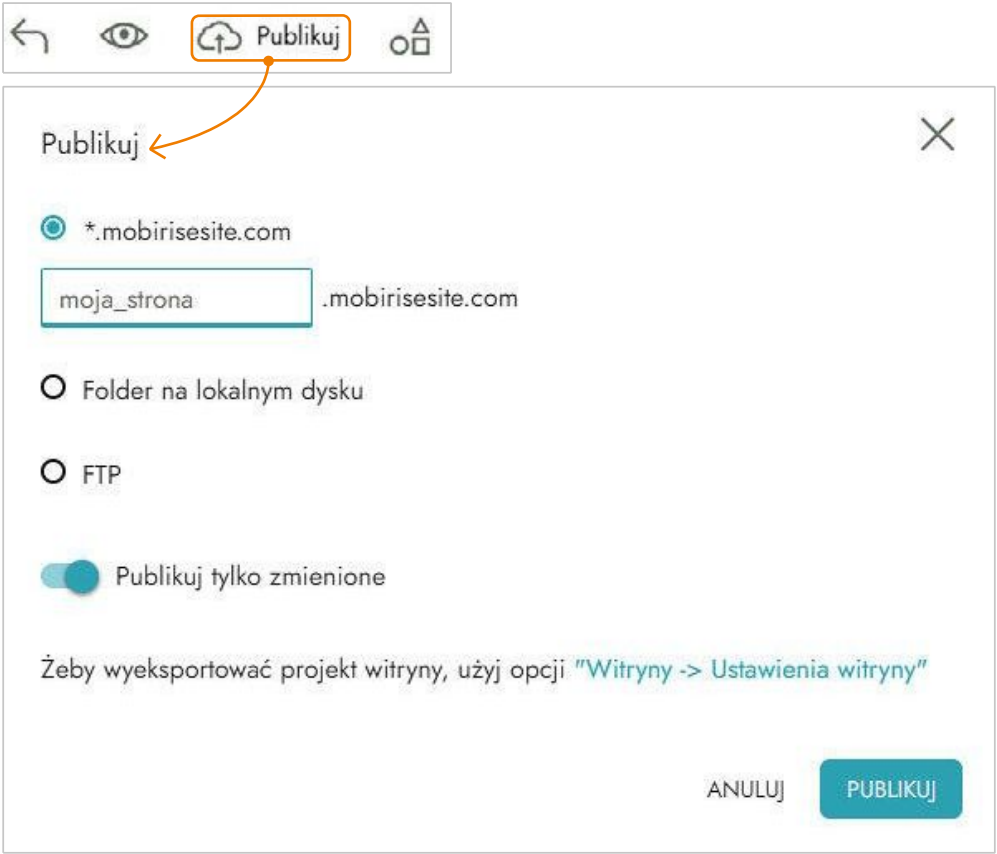
Rys. 2.15. Górny pasek narzędzi programu Mobirise

2.4. Publikowanie witryny WWW

Aby wyświetlić stronę internetową w przeglądarce, zwykle trzeba podać jej **adres URL**. Przeglądarka łączy się z serwerem, na którym znajduje się strona, pobiera tę stronę, a następnie wyświetla ją w swoim oknie.

Witryny tworzone w programie Mobirise również możemy szybko umieścić na takim serwerze, czyli je **opublikować**. W tym celu klikamy przycisk **Publikuj** widoczny w prawym górnym rogu okna.

Zobaczymy okno, w którym zaznaczamy opcję ***.mobirisesite.com**, wpisujemy wybraną przez siebie nazwę (będzie ona częścią adresu) i klikamy przycisk **Publikuj** (rys. 2.16).



Rys. 2.16. Publikowanie strony w Internecie

Kiedy strona zostanie opublikowana, otrzymamy łącze URL. Możemy je przekazać znajomym, którym chcemy pokazać naszą stronę.

Ćwiczenie 7

Opublikuj swoją witrynę. Przekaż koleżankom i kolegom z klasy jej adres.

Zapamiętaj

Publikowanie strony polega na umieszczeniu jej na serwerze, który najczęściej należy do firmy oferującej taką usługę.

Adres URL
podręcznik *NOWA Informatyka na czasie 1, Zakres podstawowy* s. 38

Publikowanie

Warto wiedzieć
Jeśli ktoś już opublikował witrynę pod wybranym przez nas adresem, program poprosi o wybranie innej nazwy.

Warto wiedzieć
Publikowanie strony zajmie trochę czasu. Ten czas jest zależny od wielkości witryny i prędkości łącza internetowego.

Zadania

- * **1** Korzystając z systemu do wizualnego tworzenia treści, utwórz stronę o twojej szkole. Zobacz, jak strona wygląda w przeglądarce internetowej.

- * **2** Przygotuj stronę internetową w systemie do wizualnego tworzenia treści. Wykorzystaj jeden z proponowanych tematów:
 - a. Wybrane zwierzę domowe – opieka, karmienie, ciekawostki.
 - b. Ulubiona książka lub ulubiony film – opis fabuły, recenzja, ciekawostki.
 - c. Ulubiona gra komputerowa lub gra planszowa.
 - d. Wybrana dyscyplina sportowa lub drużyna.

- * **3** Dodaj do swojej strony wielopoziomowe menu.

- ** **4** Utwórz witrynę WWW na wybrany temat, składającą się z kilku stron. Wykorzystaj bloki programu Mobirise, w których dostosujesz treści i obrazy. Zadbaj o poprawność działania łączy pomiędzy stronami. Opublikuj stronę na lokalnym dysku.

- ** **5** Wykonaj projekt strony WWW, na której opiszesz 10 ciekawych miejsc w Polsce. Dla każdej z opisywanych lokalizacji umieść mapę, galerię zdjęć i opisy (wykorzystaj odpowiednie bloki).

- ** **6** Wykonaj stronę, na której opiszesz swoje ulubione zajęcia i umiejętności, które masz. Potraktuj tę stronę jako swoją wizytówkę. Jeśli chcesz, udostępnij ją w Internecie, a link przekaż znajomym.

- ** **7** Wykonaj w programie Mobirise witrynę składającą się z jednej strony. Postaraj się wykorzystać jak największą liczbę bloków dostępnych w programie.

- *** **8** Obejrzyj strony internetowe różnych agencji marketingowych. Następnie wykonaj witrynę WWW dla hipotetycznej agencji marketingowej. Opisz na stronach witryny usługi agencji i w ciekawy sposób zaprezentuj jej portfolio. Opublikuj stronę tak, aby była widoczna w Internecie.

- *** **9** Strona <https://playground.wordpress.net> oferuje możliwość testowania w przeglądarce systemu CMS WordPress zarówno od strony panelu administratora, jak i wyglądu witryny. Korzystając z niej, dokonaj edycji przykładowej, stworzonej przez ciebie strony.

- *** **10** Utwórz w programie Mobirise witrynę, która będzie stanowiła atlas grzybów. Podziel grzyby na jadalne i trujące, opisz je i zamieść ich zdjęcia.

3

Sztuka publikowania w sieci

Nasze działania w różnych dziedzinach życia coraz częściej mają miejsce w sieci, a czasem muszą wręcz ograniczyć się tylko do niej. Dlatego szczególnie ważna staje się umiejętność szeroko rozumianego publikowania w Internecie – od komunikacji z przyjaciółmi po przygotowywanie materiałów na potrzeby nauki, pracy czy hobby. Z tego tematu dowiesz się, jak tworzyć zrozumiałe, atrakcyjne i profesjonalne treści, które mają szansę zaistnieć w sieci.

Ważne w tym temacie

- Opracowywanie treści interesujących dla odbiorców
- Narzędzia pomagające w przygotowaniu publikacji wzbogaconych o grafikę, nagrania audio lub filmy
- Wyjaśnienie pojęć: webcast, webinarium, screencast i podcast
- Przygotowanie do wystąpienia przed kamerą i mikrofonem
- Narzędzia pozwalające monitorować i analizować ruch w Internecie

Twórcy internetowi najczęściej wybierają jedną główną formę publikowania, np. teksty na blogu, filmy w serwisie wideo lub audycje tylko do słuchania. Jednak niezależnie od tego, jaką podstawową formę przekazu wybierzemy, przyda się umiejętność tworzenia pozostałych. Publikacje filmowe wymagają bowiem np. przygotowania tekstów zapowiadających ich premierę, a także komunikatów (tekstu lub grafiki) do zamieszczenia w sieciach społecznościowych.

Ćwiczenie 1

Zastanówcie się, co przyciąga waszą uwagę w sieci. Omówcie to w grupie i wymieńcie najważniejsze czynniki. Następnie zastanówcie się, co sprawia, że:

- a. wracacie do tego samego miejsca w sieci – witryny lub kanału,
- b. przestajecie odwiedzać daną witrynę lub dany kanał.

3.1. Jak zainteresować odbiorców swoimi publikacjami?

Gdy przeglądamy informacje w sieci, pewne czynniki decydują, że przy jednych treściach zatrzymujemy się na dłużej, by poznać szczegóły. Z kolei w wypadku innych treści jest przeciwnie – pomijamy je i przechodzimy dalej. Podobnie jest z całym serwisami internetowymi.

Zastanówmy się, które elementy sprawiają, że publikowane w Internecie treści są atrakcyjne i przyciągają uwagę odbiorców.

Warto wiedzieć

Nad zróżnicowaniem formy przekazu warto się zastanowić w przypadku serii publikacji, które będą się pojawiać cyklicznie. Wówczas formę można dopasować do aktualnego tematu.

Warto wiedzieć

Tworzenie artykułów skierowanych do czytelników z niższym wykształceniem wymaga odpowiedniego dostosowania tekstu. Kiedyś redakcje zatrudniały osoby o takim profilu w roli testerów, którzy oznaczali niezrozumiałe dla nich fragmenty tekstu. Z czasem jednak redaktorzy uczyli się pisać wystarczająco przystępnie, a testerzy tracili pracę.

Treści dopasowane do odbiorców

Przygotowanie treści dopasowanych do odbiorcy, jego potrzeb i oczekiwań jest w dzisiejszych czasach koniecznością – bez tego nie uda się wzbudzić szerszego zainteresowania.

Inaczej należy przygotować np. zaproszenie do obejrzenia filmu dla rówieśników, a inaczej dla nauczycieli czy specjalistów w danej dziedzinie. Trzeba odpowiednio dobrać konstrukcję samej publikacji, język przekazu oraz warstwę graficzną. Nie bez znaczenia jest wiek i wykształcenie odbiorcy.

Ćwiczenie 2

Opisz krótko swoją ulubioną grę, książkę lub film w sposób dostosowany do odbiorców w wieku 10 lat oraz 70 lat.

Co przyciąga uwagę użytkowników?

Oto wybrane elementy, które przyciągają uwagę odbiorców podczas przeglądania informacji w Internecie.

- Lid** ▶ Tytuły (nagłówki) oraz **lidy** (ang. *lead*), czyli krótkie zapowiedzi treści. Wzbudzające emocje i intrygujące tytuły zachęcają do przeczytania, obejrzenia lub odsłuchania całości materiału.
- ▶ Dobrze skonstruowane teksty informacyjne – wprowadzenia, streszczenia, zaproszenia oraz zapowiedzi, np. w mediach społecznościowych.
- ▶ Ilustracje – wszelkiego rodzaju grafiki związane z danym tematem, włącznie z banerami, oraz odgrywające rolę informacyjną, np. odpowiedniki plakatów filmowych.
- ▶ Właściwy dobór słów, które przybliżą użytkownikowi zagadnienie, ale również będą odpowiadały zapytaniom w wyszukiwarkach internetowych. W ten sposób można zwiększać szansę na to, że naszą publikację znajdą nowi użytkownicy, bo pojawi się wysoko w **organicznych wynikach wyszukiwania** (ang. *organic search*), czyli takich, które nie są wynikiem opłaconych kampanii.
- Hashtag** ▶ **Hashtagi**, czyli krótkie teksty poprzedzone znakiem #, które pełnią funkcję znacznika treści. Hashtagi pozwalają uporządkować pokrewne publikacje, a niekiedy pomagają różnym inicjatywom zyskać rozgłos.

Ważnym elementem, który może zwrócić uwagę internautów na publikację, są również reklamy. Najczęściej mają one postać graficzną lub tekstową. Warto pamiętać o dwóch ważnych zasadach. Po pierwsze – reklama nie może być zbyt nachalna. Nikt nie lubi często wyskakujących banerów. Po drugie – reklama nie może oszukiwać.

Tworzenie tytułów zachęcających do kliknięcia, ale mylących co do zawartości, jest niedopuszczalne. Takie działanie nazywane jest często

Clickbait ▶ **clickbaitem** (od ang. *to click* – klikać, *bait* – przynęta) i jest bardzo negatywnie odbierane przez użytkowników.

Warto wiedzieć

Tytuł publikacji, który ma przyciągnąć uwagę, nie musi ściśle odpowiadać jej tematowi. Na przykład artykuł o inteligencji szczurów może mieć tytuł: *Dlaczego tak trudno pozbyć się szczurów?*

Organiczne wyniki wyszukiwania

Jak utrzymać uwagę odbiorców?

Zastosowanie kilku wymienionych poniżej zasad może sprawić, że odbiorcy nie tylko zapoznają się z przygotowanym przez nas materiałem, lecz także będą wracać do naszego serwisu.

- ▶ Twórz interesujące i użyteczne treści, dostosowane do potrzeb i oczekiwań odbiorcy pod względem zawartości i języka przekazu.
- ▶ Aktualizuj treści – dbaj o to, aby nawet starsze materiały zawierały aktualne odnośniki. Sprawdzaj, czy wszystko działa prawidłowo, oznaczaj nieaktualne treści jako archiwalne lub usuwaj je z serwisu.
- ▶ Bierz odpowiedzialność za słowo – publikuj informacje sprawdzone w rzetelnych źródłach, konsekwentnie realizuj zapowiadane działania, zrezygnuj z publikowania treści, których nie rozumiesz.
- ▶ Publikuj systematycznie, zwłaszcza na blogach i w serwisach z poradami. Jest to obowiązek, jeśli tworzysz coś w formie serii złożonej z wielu odcinków (części).
- ▶ Monitoruj ruch w sieci i **poznawaj własnych odbiorców** – pozwoli to lepiej dopasować treści do potrzeb odbiorców.
- ▶ Komunikuj się z odbiorcami – odpowiadaj na ich pytania i uwagi, zachęcaj do wykonania określonych działań (np. udostępniania lub zakupu), badaj preferencje użytkowników (np. za pomocą ankiet) umożliwiaj dyskusję, organizuj konkursy.

Dobra rada

Ponieważ filmy trudniej aktualizować niż teksty, możesz w zamian przygotować odpowiednie adnotacje, np. w opisie filmu.

Jak poznać (lepiej) swojego odbiorcę?
→ s. 62

Storytelling
podręcznik NOWA Informatyka na czasie 1. Zakres podstawowy
→ s. 60

Ćwiczenie 3

Przypomnij sobie, na czym polega storytelling. Zastanów się, czy i jak można go zastosować do treści publikowanych w Internecie.

Zapamiętaj

Publikacja powinna uwzględniać potrzeby i zainteresowania odbiorców. Należy zadbać o to, by materiał był interesujący, przejrzysty, kompletny oraz treściwy, a zamieszczone w nim informacje były sprawdzone w rzetelnych źródłach.

A to ciekawe

Czym jest content marketing?

Content marketing to jedna z popularnych i efektywnych form marketingu, którą powszechnie wykorzystuje się w Internecie. Publikowanie interesujących i przydatnych treści (np. artykułów, webinarów, infografik) przyciąga klienta do firmy i potwierdza, że pracują w niej eksperci w danej dziedzinie. Dzięki temu łatwiej jest przekonać użytkownika np. do zakupów.



JAK PRZYGOTOWAĆ TREŚĆ PUBLIKACJI?

Aby sprawnie przygotować publikację, która zainteresuje wiele osób, warto odpowiednio zorganizować sobie pracę.

1

Zrób plan

Znajdź pomysł na publikację i zaplanuj jej zawartość.

Publikacja tekstowa

- konspekt z punktami do opracowania
- opis grafik ilustrujących tekst

Nagranie wideo lub występ na żywo

- scenariusz lub scenorys z opisem scen i ujęć
- tekst lektora
- opis potrzebnych grafik

2

Podдай plan krytycznej analizie

Przygotowany plan warto przeanalizować i odpowiedzieć sobie na kilka pytań:

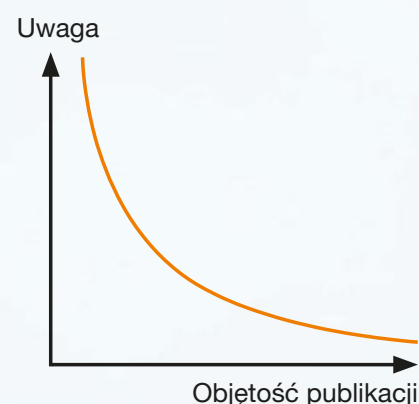
1. Czy to jest dla kogoś ważne? ✓
2. Czy to jest ciekawe? ✓
3. Czy to jest przejrzyste? ✓
4. Czy to jest kompletne? ✓
5. Czy to jest treściwe? ✓
6. Czy to jest poprawne? ✓

Odpowiedzi na te pytania powinny skłonić do wprowadzenia niezbędnych poprawek.

3

Ustal objętość publikacji

- Im dłuższy materiał, tym więcej czasu zajmie jego przygotowanie.
- Pamiętaj, że wraz ze wzrostem objętości publikacji spada uwaga odbiorców. Publikacja powinna omawiać wszystkie istotne zagadnienia, ale nie powinna być zbyt obszerna.



4

Wymyśl tytuł

- Dobry tytuł powinien przyciągać uwagę i zachęcać do zapoznania się z treścią.
- Sformułuj tytuł tak, by łatwo było go używać w różnych celach, m.in. w zapowiedziach, na banerach, plakatach, gadżetach, planszach początkowych i końcowych.
- Tytuł powinien brzmieć tak samo, niezależnie od miejsca użycia – nawet drobne odstępstwa mogą dezorientować odbiorców.

5

Wykonaj plan

- Podczas tworzenia publikacji trzymaj się przygotowanego konspektu lub scenariusza.
- Wprowadź modyfikacje, jeśli poprawią one jakość materiału i sprawią, że będzie lepiej odpowiadał potrzebom odbiorców.
- Opublikuj materiał zgodnie z zapowiedziami. Możesz skorzystać z systemów, które automatycznie publikują treści o określonej porze.

Unikaj błędów

- Przed publikacją sprawdź: imiona i nazwiska, nazwy, adresy, daty, dane teleadresowe itp.
- Nie używaj wyrażen typu *dziś*, *w zeszłym tygodniu*, bo nie wiesz, kiedy odbiorca zapozna się z publikacją.

Śledź reakcje i wprowadzaj poprawki

- Czytaj komentarze i utrzymuj kontakt z odbiorcami.
- Staraj się zrozumieć grono swoich odbiorców – pamiętaj, że może się ono zmieniać.
- Analizuj trendy w sieci.
- Dostosowuj nowe publikacje do oczekiwań odbiorców i aktualizuj starsze treści.

3.2. Teksty informacyjne

Tekst informacyjny

Teksty opracowuje się nie tylko na potrzeby publikacji ściśle tekstowych, lecz także podczas tworzenia filmów, podcastów czy webinarów. Najczęściej są to **teksty informacyjne**, czyli takie, których głównym zadaniem jest skuteczne przekazanie informacji. Z tego powodu zazwyczaj są pozbawione wszelkich środków artystycznych, cechujących np. teksty literackie.

Przygotowanie do pisania

Przed rozpoczęciem pisania należy się zastanowić, jakiego rodzaju tekst informacyjny będzie nam potrzebny. Może to być m.in.:

Podcast
→ s. 56

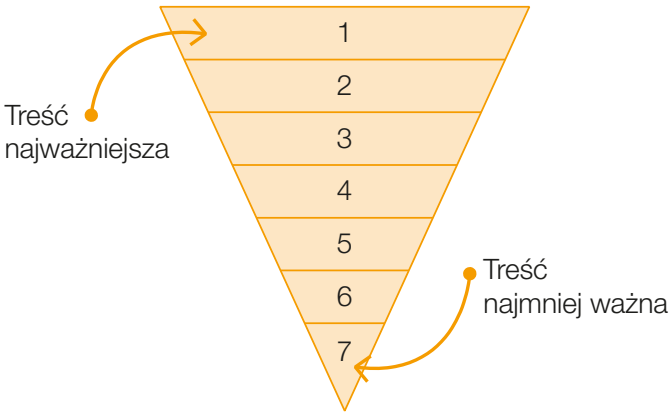
- zaprośenie, np. do obejrzenia filmu lub wysłuchania **podcastu**,
- zapowiedź wydarzenia,
- opis audycji,
- relacja z wydarzenia,
- tekst do działu „O nas” na stronę internetową lub blog.

Każdy z wymienionych typów ma inną charakterystykę i zazwyczaj zawiera nieco inne informacje. Zanim zaczniemy pisać, musimy ustalić, jaka jest najważniejsza informacja do przekazania, oraz przemyśleć konstrukcję tekstu.

Konspekt

Nawet w wypadku krótkiego tekstu warto przygotować **konspekt**, czyli zarys przedstawiony w punktach. Powinny się w nim znaleźć zagadnienia, które chcemy uwzględnić. Dzięki temu podczas pisania o niczym nie zapomnimy.

Punkty w konspekcie warto na bieżąco ustawiać w formie odwróconej piramidy (rys. 3.1). Najważniejsze treści, które opiszemy najszerzej, umieszczamy na początku, a najmniej ważne, którym poświęcimy mniej miejsca – na końcu.



Rys. 3.1. Konstrukcja konspektu w formie odwróconej piramidy

Dzięki zapisaniu haseł z konspektu w formie odwróconej piramidy już w trakcie tworzenia tekstu zorientujemy się, jaka będzie jego objętość i z jakich treści możemy zrezygnować, aby nadal przekazać wszystkie najważniejsze informacje.

Dobra rada

Stworzenie konspektu da ci jeszcze jedną korzyść – liczba pozostałych do opracowania punktów będzie przypominała o ilości pracy do wykonania oraz skłaniała do zwięzłego opisywania punktów.

Tworzenie komunikatu tekstowego

W napisaniu tekstu może nam pomóc **zasada 5W**. Jej nazwa pochodzi od pięciu pytań w języku angielskim zaczynających się na literę „w”. Te pytania należy sobie postawić w odniesieniu do informacji, którą chcemy przekazać:

1. *Who?* – Kto?
2. *What?* – Co?
3. *Where?* – Gdzie?
4. *When?* – Kiedy?
5. *Why?* – Dlaczego (po co)?

Jeśli odpowiemy w tekście na powyższe pytania, w wielu przypadkach to wystarczy, aby zaintrygować czytelnika i przekazać mu wszystko, co najważniejsze. Jednak konieczne jest także uwzględnienie tych informacji, które będą szczególnie istotne dla naszego odbiorcy. Pomoże w tym również przygotowanie **konspektu** w formie odwróconej piramidy.

Jeśli tekst zaprasza do obejrzenia filmu, w którym występuje znana i ceniona przez użytkowników osoba, to już w pierwszych zdaniach warto podać jej nazwisko i dodatkowo wyróżnić je w tekście.

Jeśli jednak dopiero budujemy swoją markę, na pierwszym miejscu umieścimy interesujący tytuł oraz opis, który przyciągnie odbiorców. Własne nazwisko należy podać w dalszej części. Jeśli publikacja się spodoba, nazwisko (pseudonim lub logo) autora zacznie być prędkiej czy później rozpoznawalne i z czasem może samo w sobie stać się zachętą dla użytkowników.

Ćwiczenie 4

Zastosuj zasadę 5W, aby:

- a. przygotować tekst zaproszenia na zbliżające się webinarium z udziałem eksperta,
- b. zmodyfikować przygotowany tekst tak, aby zapraszał do obejrzenia nagrania webinarium, które już się odbyło.

Tworzenie treści a narzędzia sztucznej inteligencji

Tworząc treści, możemy korzystać z narzędzi sztucznej inteligencji. Mogą one pomóc np.:

- z głębić wybrany temat,
- znaleźć przydatne informacje, materiały oraz inspiracje,
- opracować konspekt tekstu, scenariusz nagrania lub wystąpienia,
- przygotować streszczenia artykułów,
- opracować wstępny plan prezentacji,
- przetłumaczyć teksty, artykuły, a także napisy do filmów z języków obcych.

Warto wiedzieć

Zasada 5W jest powszechnie stosowana podczas przygotowywania informacji dziennikarskiej.

Konspekt
→ s. 48

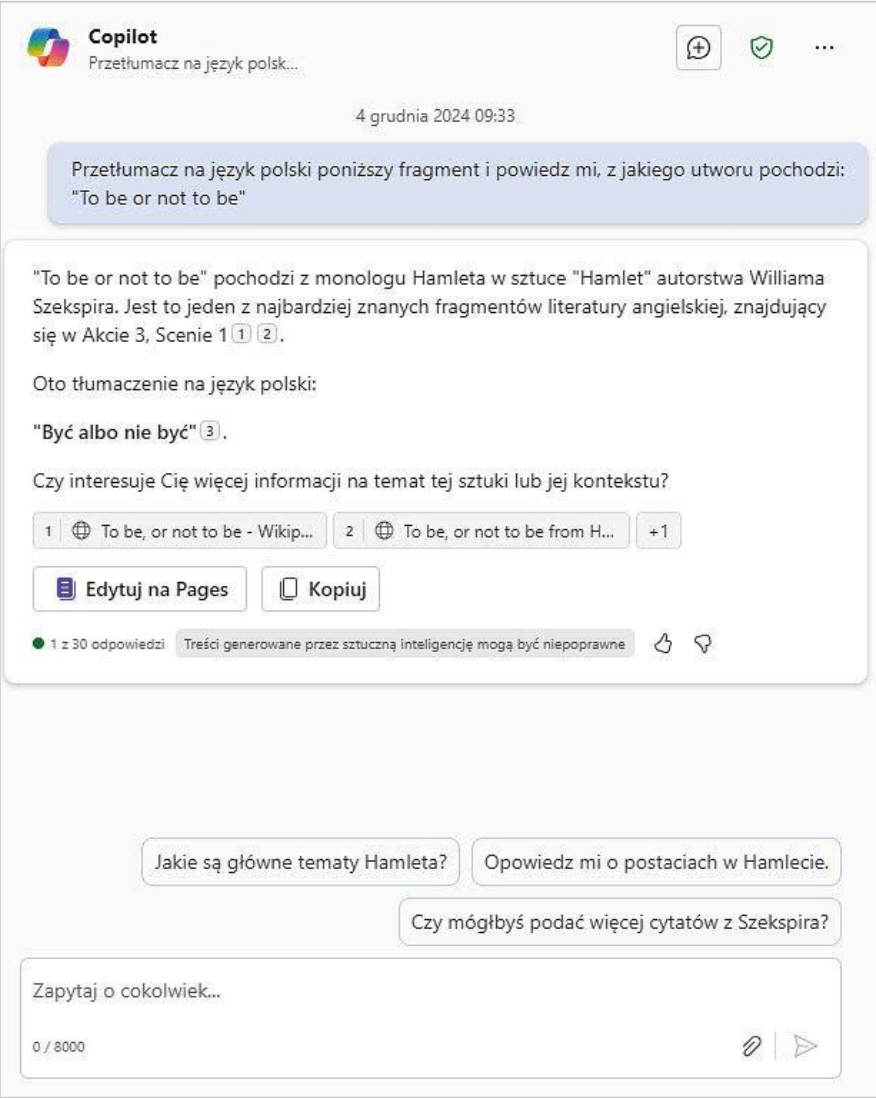
Dobra rada

Wykorzystując narzędzia SI, takie jak Copilot, Gemini czy ChatGPT, możesz w zapytaniu określić, do kogo ma być skierowany tekst (np. wiek, wykształcenie, zainteresowania), oraz podać oczekiwaną długość tekstu.

Dobra rada

Przed wykorzystaniem narzędzi SI zastanów się, czy ranga problemu wymaga ich użycia. Czasami szybciej i wygodniej znajdziesz potrzebne informacje, korzystając z tradycyjnej wyszukiwarki.

Narzędzia sztucznej inteligencji mogą się przydać także podczas pracy z grafiką, dźwiękiem czy filmem. Warto jednak pamiętać, że materiały przygotowane z użyciem takich narzędzi nie mogą być traktowane jako gotowe do opublikowania. Mimo że odpowiedzi sztucznej inteligencji często są precyzyjne (rys. 3.2), to dla pewności warto je sprawdzać w innych rzetelnych źródłach.



Rys. 3.2. Odpowiedź wygenerowana przez Microsoft Copilot

A to ciekawe

Klucz do sukcesu

O tym, jak duże znaczenie ma tytuł oraz użyte w nim słownictwo, przekonała się kiedyś firma Microsoft. Ten sam artykuł z poradami dotyczącymi edytora prezentacji został opublikowany pod dwoma bardzo podobnymi tytułami: *10 smart tips for using PowerPoint* oraz *10 key tips for using PowerPoint*. Okazało się, że artykuł opatrzone drugim tytułem przeczytało kilka razy więcej osób.



3.3. Grafika i identyfikacja wizualna

Przygotowanie grafiki, podobnie jak opracowanie tekstów informacyjnych, towarzyszy każdej formie publikacji sieciowej. Do elementów graficznych możemy zaliczyć:

- ▶ ilustracje w tekście – zdjęcia, **grafiki informacyjne (infografiki)**, wykresy, schematy,
- ▶ plansze tytułowe i końcowe w filmach lub wystąpieniach na żywo,
- ▶ podpisy osób w filmie lub w czasie wystąpień na żywo,
- ▶ plakaty i banery informacyjne zapraszające na wydarzenia oraz zachęcające do zapoznania się z publikacjami,
- ▶ logotypy,
- ▶ materiały prezentacyjne i animacje wykorzystywane w filmach lub podczas wystąpień na żywo.

Język wizualny stał się bardzo popularnym sposobem komunikacji w Internecie, ponieważ jest prosty i atrakcyjny oraz przyciąga uwagę odbiorców. Warto poświęcić czas na przemyślenie i przygotowanie odpowiednich grafik towarzyszących publikacji sieciowej.

Oprogramowanie i serwisy ułatwiające przygotowanie grafik

Istnieje wiele aplikacji i serwisów, dzięki którym można przygotować atrakcyjną grafikę. Niektóre z nich omówiliśmy w poprzedniej części podręcznika. Warto jednak kilka aplikacji i serwisów wyróżnić. Przedstawia je tabela 3.1.

Program lub serwis	Przeznaczenie
Canva	Serwis oferuje wiele narzędzi, szablonów oraz grafik, które można wykorzystać w projektach do stworzenia m.in. plakatów, banerów, prezentacji i plansz tytułowych.
Edytor prezentacji (np. PowerPoint, Impress)	Aplikacje do tworzenia prezentacji mają wbudowane narzędzia graficzne. Dzięki nim można wygodnie przygotować plakat, baner lub planszę tytułową.
Narzędzie Wycinanie (Windows)	To narzędzie pozwala na łatwe wykonywanie i przetwarzanie zrzutów ekranowych oraz nagrywanie ekranu, co może być przydatne podczas tworzenia samouczków. Można je uruchomić skrótem klawiszowym Windows + Shift + S .
Pixabay, Pixels, Unsplash	Te serwisy internetowe udostępniają zdjęcia, grafiki, filmy i nagrania audio na otwartych licencjach (często do użytku komercyjnego). Przed użyciem zdjęcia trzeba sprawdzić warunki licencyjne, ponieważ mogą one narzucać sposób oznaczania autora zdjęcia.
Programy graficzne (np. Inkscape, GIMP)	Program do grafiki wektorowej (np. Inkscape) można wykorzystać np. do stworzenia logo lub baneru. Programy do grafiki rastrowej pomogą m.in. odpowiednio wykadrować zdjęcia.

Tabela 3.1. Programy i serwisy ułatwiające przygotowanie grafik do publikacji

Grafiki informacyjne
podręcznik *NOWA Informatyka na czasie 1. Zakres podstawowy*
s. 73–78

Dobra rada

Wskazówki dotyczące grafiki możesz znaleźć w tematach *Sztuka prezentacji* oraz *Grafika rastrowa i Grafika wektorowa* z podręcznika *NOWA Informatyka na czasie 1. Zakres podstawowy*.

Warto wiedzieć

W Internecie można znaleźć również serwisy oferujące darmowe zasoby muzyczne, np. Free Music Archive lub Bensound. Istnieje także wiele witryn umożliwiających generowanie własnej muzyki z wykorzystaniem algorytmów sztucznej inteligencji, np. Udio.

Warto wiedzieć

Canva oferuje narzędzia do generowania grafik z wykorzystaniem sztucznej inteligencji.

Tworzymy zaproszenie na webinar, korzystając z serwisu Canva

Jednym z serwisów, który może pomóc nam w przygotowaniu różnorodnych treści do publikacji w Internecie, jest Canva. Oferuje ona bogatą bibliotekę narzędzi, gotowych szablonów oraz grafik, zdjęć, filmów i dźwięków.

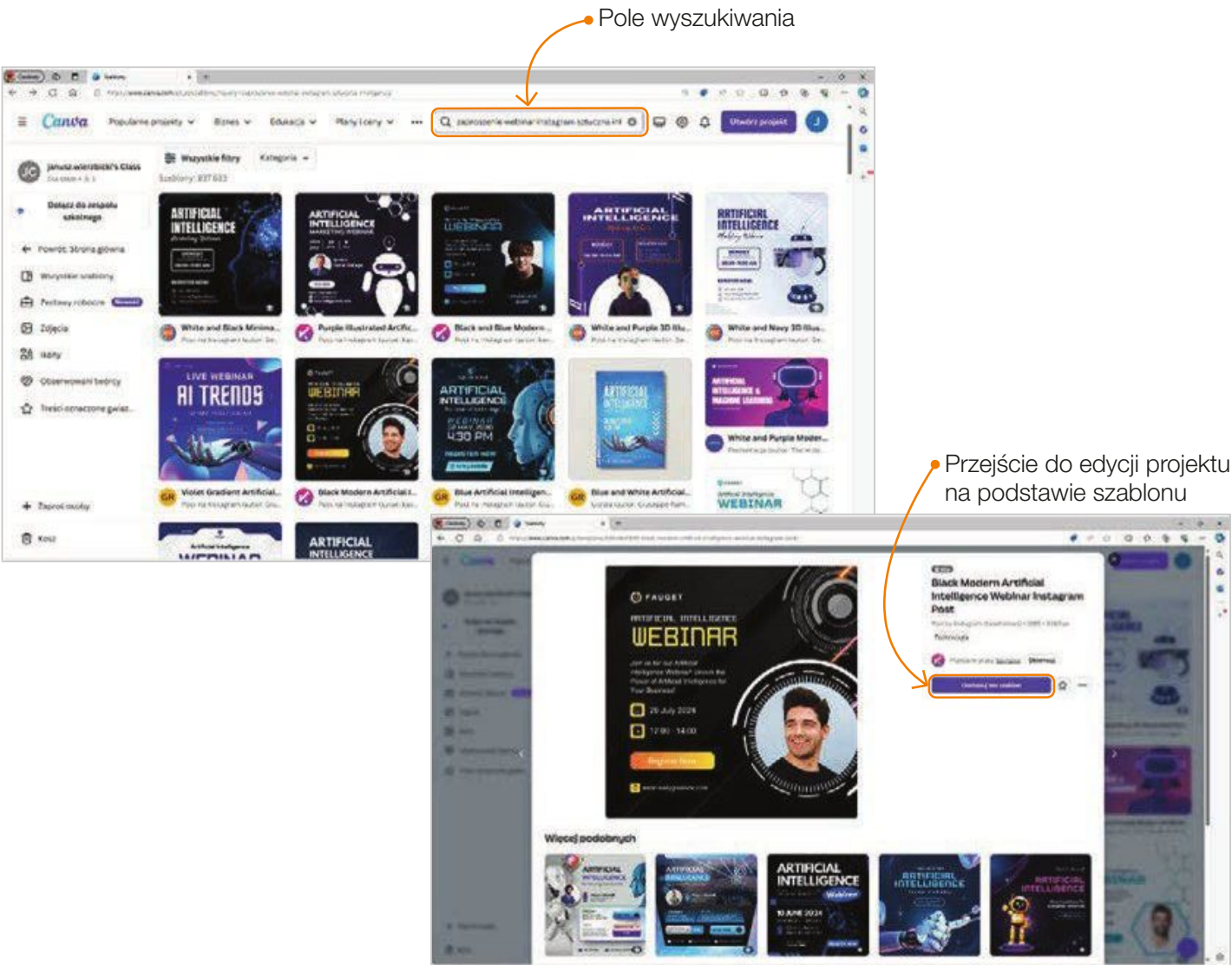
Zakładamy, że chcemy stworzyć zaproszenie na webinar dotyczący możliwości, ograniczeń, szans i zagrożeń zastosowania sztucznej inteligencji w edukacji.

Tworzenie zaproszenia w serwisie Canva

Tworzenie zaproszenia w serwisie Canva krok po kroku:

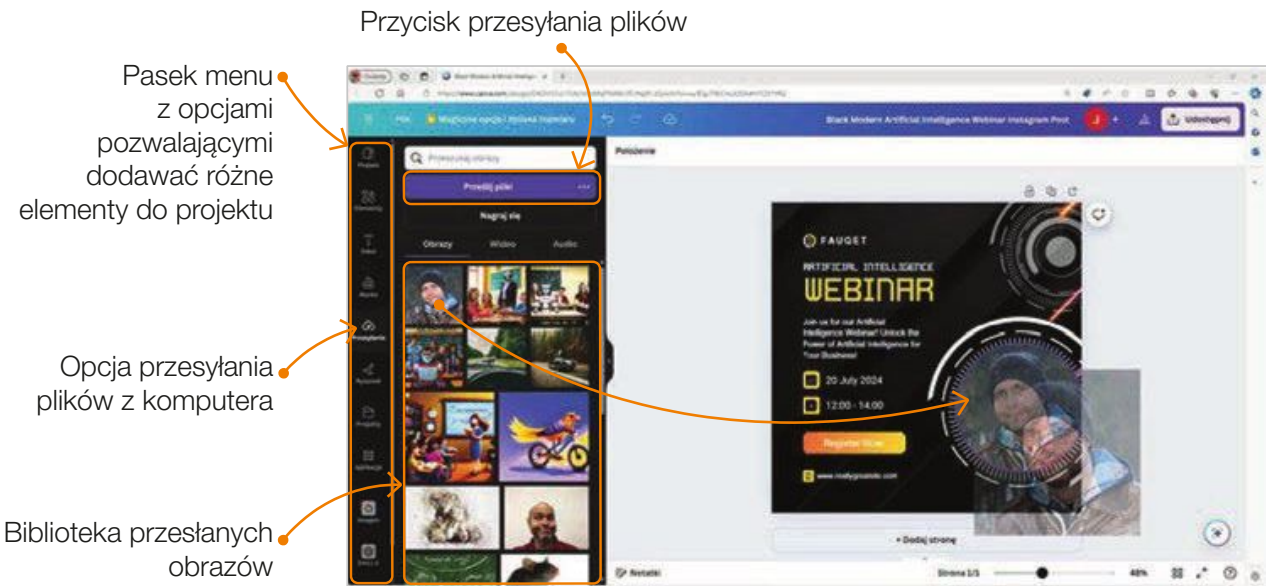
1. Zaloguj się do serwisu Canva i, korzystając z pola wyszukiwania, znajdź i wybierz szablon pasujący do tematu.
2. Wymień zdjęcia i grafiki w szablonie na własne oraz wpisz swoje teksty.
3. Usuń niepotrzebne elementy i w razie potrzeby dodaj nowe.
4. Pobierz utworzone zaproszenie na swój komputer.

Po zalogowaniu się w serwisie opisujemy w polu wyszukiwania, jaką grafikę chcemy utworzyć. Po znalezieniu interesującego nas szablonu otwieramy go do edycji (rys. 3.3).



Rys. 3.3. Wyszukiwanie szablonu projektu

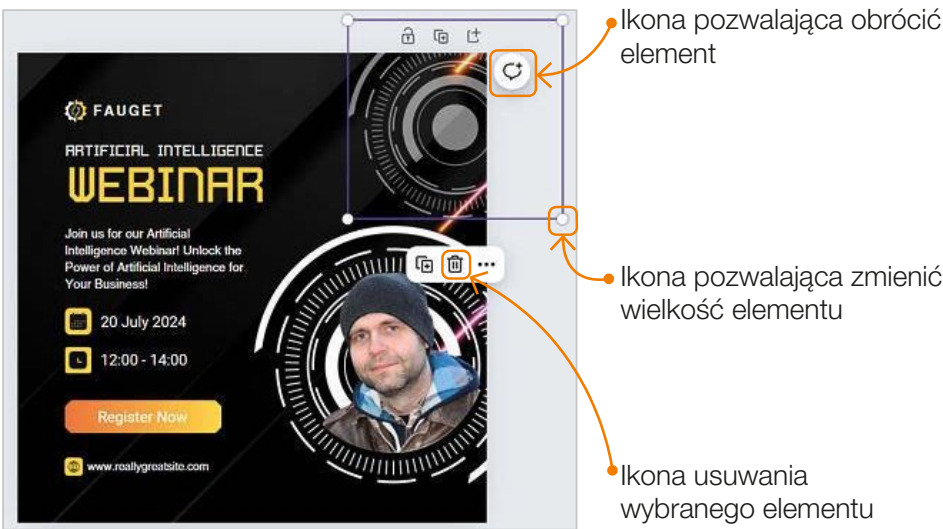
Zdjęcie zamieszczone w szablonie możemy zamienić na swoje. W tym celu z bocznego paska wybieramy opcję **Przesyłanie**, a następnie **Prześlij pliki**. Wskazujemy na dysku zdjęcia, które chcemy dodać do projektu. Pojawia się one w panelu po lewej stronie, skąd możemy je przenieść do okna z projektowaną grafiką (rys. 3.4). Po dwukrotnym kliknięciu na dodanej fotografii możemy ją jeszcze wykadrować lub dodać jakiś efekt.



Rys. 3.4. Zamiana zdjęcia w projekcie

Z paska bocznego możemy też wybierać elementy do dodania. W tym celu wybieramy kategorię obiektu, który chcemy dodać, i przeciągamy do okna projektu wybrany element.

Możemy usuwać elementy. W tym celu zaznaczamy kliknięciem te, których chcemy się pozbyć, i naciskamy przycisk **Delete** na klawiaturze lub klikamy ikonę kosza, która pojawia się obok zaznaczonego elementu. Z tego miejsca możemy też obrócić wybrany element lub zmienić jego rozmiar (rys. 3.5).



Rys. 3.5. Operacje na wybranym elemencie projektu (obracanie, zmiana wielkości, usuwanie)

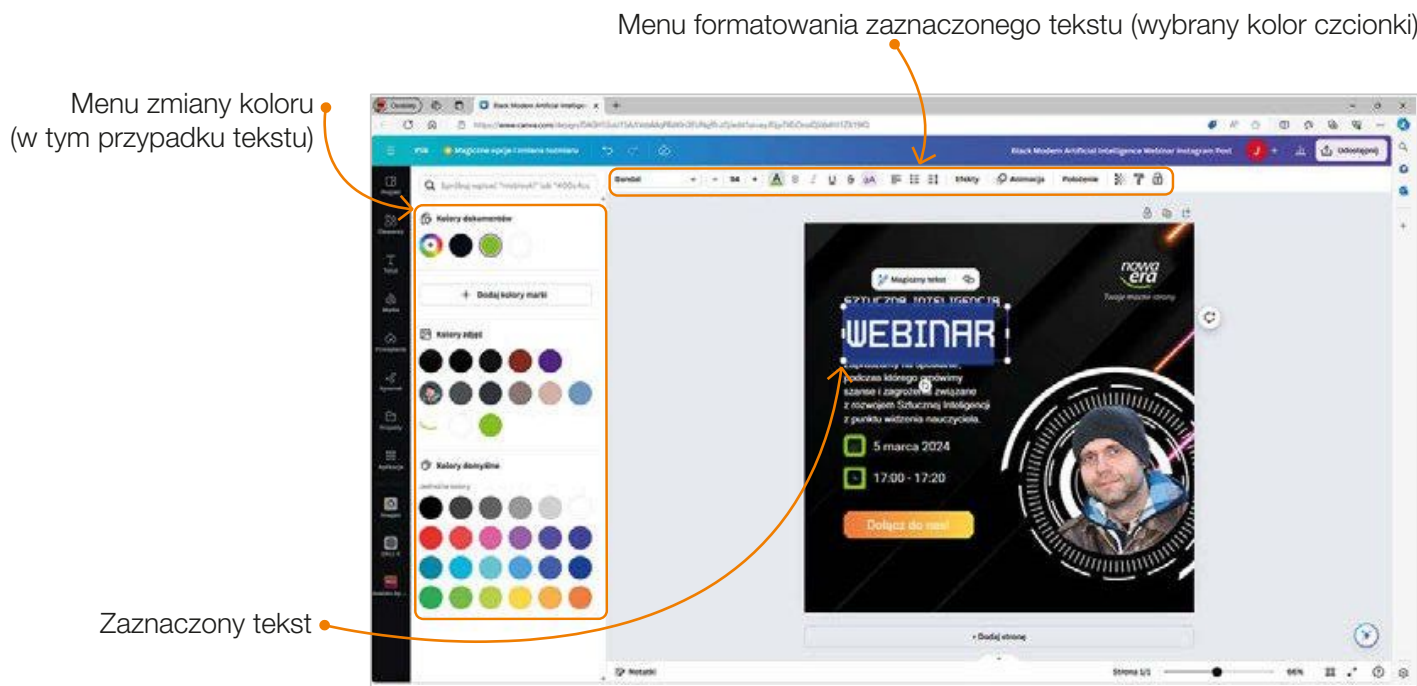
Dobra rada

Kiedy opracowujesz prezentację multimedialną lub film, możesz dodać do projektu filmy i dźwięki w taki sam sposób jak zdjęcia.

Dobra rada

Jeśli chcesz usunąć kilka elementów jednocześnie, zaznaczaj je kolejno, trzymając klawisz **Shift**.

Po dwukrotnym kliknięciu tekstu możemy rozpocząć jego edycję. W górnej części okna widoczne będą przyciski służące do formatowania tekstu (rys. 3.6).

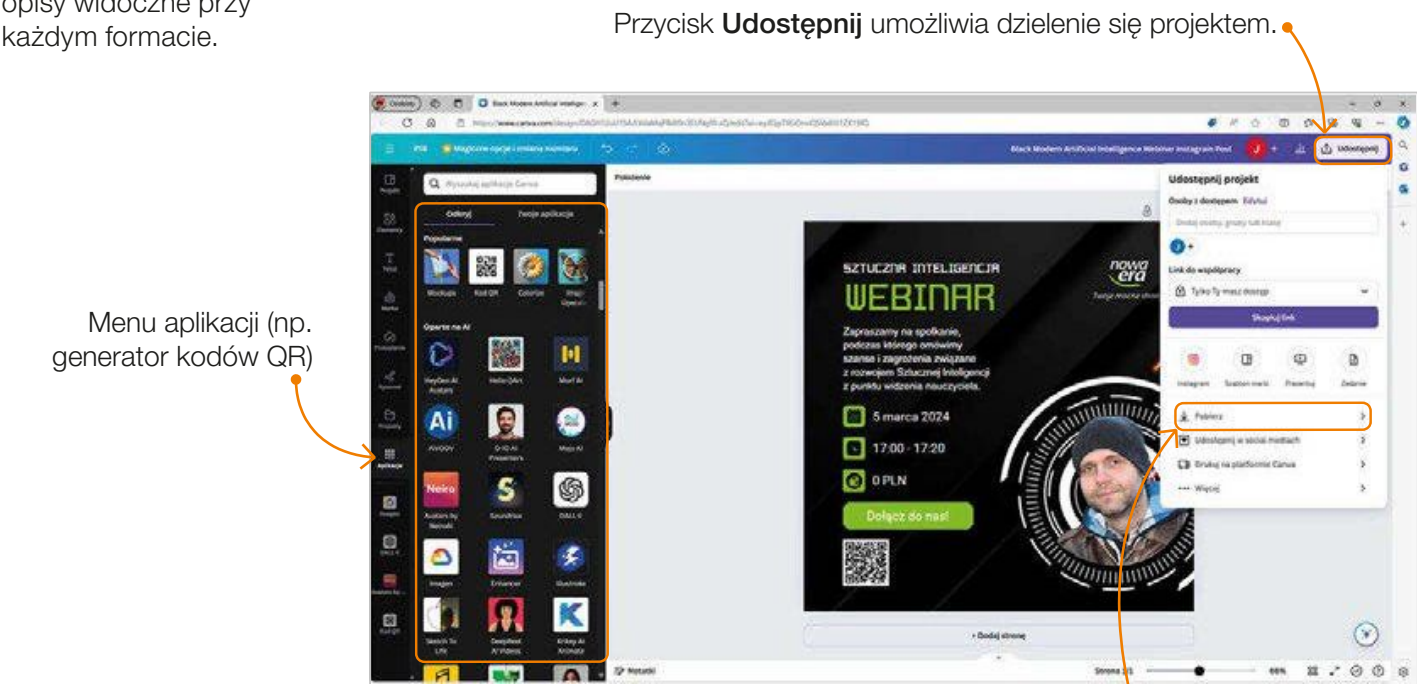


Rys. 3.6. Edycja i formatowanie tekstu

Dobra rada

Przed pobraniem grafiki możesz wybrać format zapisanego obrazu. Dobierz format do swoich potrzeb, wykorzystując opisy widoczne przy każdym formacie.

Na koniec pracy nad projektem warto zadbać o detale. Elementy dobrze jest rozmieścić tak, żeby tworzyły spójną kompozycję. Kiedy projekt jest gotowy, klikamy przycisk **Udostępnij** i z rozwiniętego menu wybieramy opcję **Pobierz**, aby pobrać grafikę na dysk naszego komputera (rys. 3.7).



Opcja **Pobierz** umożliwia zapisanie projektu na komputerze w wybranym formacie.

Rys. 3.7. Dodatkowe aplikacje oraz pobieranie gotowego projektu

Identyfikacja wizualna

Twórcy, którzy publikują treści w Internecie, dbają o odpowiednią **identyfikację wizualną**, czyli rozpoznawalność przygotowywanych przez siebie materiałów. Mogą ją zapewnić stałe, powtarzane w każdej publikacji elementy graficzne, nazywane czasem motywem lub szablonem. Należą do nich:

- czcionki,
- tła,
- kolorystyka,
- zdjęcie,
- układ treści.

Stale elementy pozwalają szybko skojarzyć treści z daną firmą, serią publikacji lub autorem. Warto także zadbać o ich spójność nie tylko w samych publikacjach, lecz także na stronie WWW, blogu, w mediach społecznościowych i we wszelkich zapowiedziach.

Jednym z elementów graficznych ściśle związanych z identyfikacją wizualną jest **logo**. Może się ono składać z samego symbolu graficznego, zwanego **sygnetem**, z sygnetu i **logotypu** (nazwy) lub samego logotypu. Czasem zawiera też dodatkowy tekst, np. slogan.

Zazwyczaj dąży się do tego, by logotyp był oryginalny, ale równocześnie prosty i łatwy do użycia w różnych publikacjach. Przykładowe logo składające się z sygnetu i logotypu przedstawiono na rysunku 3.8.



Rys. 3.8. Logo Olimpiady Informatycznej składające się z sygnetu i logotypu

Ćwiczenie 5

Wskaż wizualne i (o ile to możliwe) dźwiękowe elementy identyfikujące:

- wybrany serwis VOD (wideo na życzenie),
- Wielką Orkiestrę Świątecznej Pomocy,
- miasta: Rybnik, Katowice, Melbourne,
- wybranego blogera lub kanał internetowy.

Warto wiedzieć

W publikacjach multimedialnych stałymi elementami są intra oraz dżingle.

Logo

Sygnet, logotyp

Warto wiedzieć

Oryginalne rozwiązania przyciągają uwagę odbiorców, ale nie można z nimi przesadzić. Gdy jedna z firm technologicznych wprowadziła na swoich stronach przewijanie treści lewo–prawo, szybko musiała z pomysłu zrezygnować, bo użytkownicy nie potrafili z tego korzystać.

3.4. Nagrania audio i wideo oraz wystąpienia na żywo

Wiele treści w Internecie ma postać różnego rodzaju filmów, audycji lub relacji na żywo. Tabela 3.2 przedstawia najpopularniejsze rodzaje publikacji związane z wideo i dźwiękiem.

Webinarium (webinar) ▶

Webcast ▶

Podcast ▶

Screencast ▶

Rodzaj publikacji	Opis
Webinarium (webinar)	Rodzaj seminarium organizowanego na żywo w Internecie. Może być prowadzone przez jedną osobę lub wiele osób znajdujących się w oddalonych miejscach. Podczas webinarium wykorzystuje się transmisję audio-wideo z kamer, udostępnianie prezentacji lub zawartości ekranu komputera. Możliwa jest interakcja z publicznością, np. dyskusja z uczestnikami lub odpowiedzi na pytania (zadane np. na czacie). Webinaria wykorzystuje się głównie jako narzędzia szkoleniowe i marketingowe.
Webcast	Audycja internetowa w formie wideo (przypominająca audycję telewizyjną) najczęściej realizowana na żywo, choć może być również odtwarzana z przygotowanego wcześniej materiału. Zazwyczaj nie wykorzystuje możliwości bezpośredniej komunikacji między odbiorcami a prowadzącymi. Nie oznacza to jednak braku interakcji – widzowie mogą np. głosować w przeprowadzanych podczas programu ankietach.
Podcast	Rodzaj audycji internetowej (najczęściej audio) publikowanej regularnie jako seria odcinków.
Screencast	Film będący zapisem czynności wykonywanych na ekranie komputera uzupełniony o komentarz lektora. Wykorzystuje się go jako materiał szkoleniowy lub demonstracyjny. Nagrywanie i publikowanie tego typu filmu określane jest często jako screencasting, a film nazywany jest tutorialiem.

Tabela 3.2. Popularne typy internetowych publikacji audiowizualnych

Znaczenie przedstawionych wyżej pojęć może się zmieniać wraz z rozwojem technologii. Wszystkie te działania wymagają jednak odpowiedniej pracy z kamerą oraz mikrofonem.

Praca z kamerą

W zastosowaniach amatorskich i półprofesjonalnych do rejestrowania filmów wykorzystuje się nie tylko kamery, lecz także smartfony. Na przykład w czasie nadawania na żywo użycie smartfona pozwoli uzyskać o wiele lepszy efekt niż korzystanie z kamery wbudowanej w laptop.

Większość telefonów po zainstalowaniu dodatkowej aplikacji pozwala używać aparatu jak zewnętrznej kamery USB lub kamery bezprzewodowej (z wykorzystaniem wi-fi). Dzięki temu otrzymujemy lepszy obraz i więcej możliwości kadrowania.

W czasie przygotowań do nagrania warto pamiętać o kilku prostych zasadach, które pomogą uzyskać materiał lepszej jakości.

Warto wiedzieć

Wiele aplikacji pozwala ustabilizować zapisany film, czyli pozbawić go drgań. Warto z tego korzystać, ale trzeba zwrócić uwagę, czy po ich użyciu film nie traci na rozdzielczości lub jakości.

Zachowanie przed kamerą

Aby dobrze wypaść przed kamerą, warto stosować zasady podobne do tych, które obowiązują podczas wystąpień publicznych.

Utrzymuj kontakt wzrokowy



Podczas wywiadu patrz na osobę, z którą rozmawiasz, a nie w obiektyw kamery. W innych okolicznościach zapytaj operatora kamery, jaka pozycja będzie bardziej naturalna, lub zrób próbę i oceń to samodzielnie.

Dobierz odpowiedni strój



Kamera lubi szarości i błękity. Unikaj drobnych wzorów, pasków i zieleni – zwłaszcza gdy występujesz w studiu telewizyjnym. Załóż dłuższą spódnicę lub dłuższe spodnie, aby w pozycji siedzącej niczego niepotrzebnie nie odsłonić.

Przyjmij właściwą postawę



Wyprostuj się, stań pewnie na lekko rozstawionych nogach ze stopami skierowanymi na zewnątrz. Jeśli siedzisz, siedź prosto, ale swobodnie. Nie rozstawiaj nóg i nie zapadaj się w głębokich fotelach. Unikaj kręcenia i bujania się na krześle.

Gestykuluj z umiarem



Wyznacz przestrzeń, w której możesz się poruszać podczas nagrania, by nie wyjść z kadru, gdy kamera jest ustawiona nieruchomo (lub poproś operatora o wyznaczenie tej przestrzeni).

Unikaj biżuterii i daj się pomalować



Mieniąca się biżuteria może powodować refleksy, a duże ozdoby mogą odwracać uwagę od twojej wypowiedzi. Dzięki makijażowi oświetlenie nie będzie wywoływać refleksów na twojej skórze.

Przekazuj właściwe emocje



Uśmiech i pozytywne nastawienie prelegenta udzielają się słuchaczom. Jednak nie okazuj radości, gdy mówisz o rzeczach poważnych lub smutnych – nie będzie to naturalne ani wiarygodne.

Zanim wystąpi się przed kamerą (np. w studiu telewizyjnym lub w sondzie ulicznej), warto wcześniej porozmawiać zarówno z osobą prowadzącą rozmowę, jak i z operatorem kamery. To, jak wypadniemy na nagraniu, będzie zależało również od pracy operatora.

O zasadach zachowania przed kamerą warto pamiętać również podczas przygotowywania np. poradnika kucharskiego czy pokazu tresury ulubionego zwierzęcia. Wtedy oprócz kucharza czy tresera ważne są potrawa lub zwierzę. To one będą głównymi obiektami nagrania – należy zadbać o odpowiednie ich sfilmowanie, a samemu pozostać nieco w tle.

Na planie filmowym mogą się zdarzyć nieprzewidziane sytuacje. Ujęcia z wielu kamer (np. kilku smartfonów lub tabletów) ułatwią pozbycie się niepotrzebnych ujęć na etapie montażu, a także pozwolą zwiększyć dynamikę ostatecznej wersji filmu.

Ćwiczenie 6

Przeanalizuj filmy, które dotyczą wybranego zagadnienia, i które mają dużą oglądalność w Internecie. Zastanów się:

- a. w jaki sposób zostały nagrane i jakie techniki w nich zastosowano,
- b. jak zachowują się osoby występujące na nagraniach.

Dobra rada

Jeśli nagrywasz film z kilku kamer, poproś kogoś o pomoc. Możesz także ustawić kilka urządzeń na stałe, a samodzielnie nagrywać ujęcia z ręki.

PRACA Z KAMERĄ

Podczas nagrywania materiału wideo warto pamiętać o wskazówkach, które pomogą wykorzystać możliwości dostępnych urządzeń i podnieść jakość nagrań.

Oświetlenie

Zadbaj o to, by nagrywany obiekt lub osoba były odpowiednio oświetlone. Uważaj na cień rzucany przez czapkę lub kapelusz.

Naturalne światło

Najlepsze ujęcia powstaną o poranku, gdy słońce jeszcze jest nisko, albo po południu, gdy światło nie jest już tak ostre.



Mikrofon

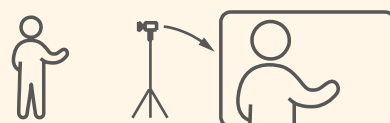
Korzystaj z mikrofonu krawatowego lub ręcznego. Zamontuj też osłonę chroniącą przed szumem.

Nagranie na zewnątrz

Stabilizacja kamery

Przy ujęciach w planie sprawdź się gimbal, czyli urządzenie, które stabilizuje kamerę.

Kadrowanie



Jeśli dużo gestykulujesz lub korzystasz z rekwizytów, lepiej wypadniesz w dalszym planie, tzw. planie średnim. Nie musisz też być w centrum kadru.

Sztuczne oświetlenie

Światło powinno być rozproszone, miękkie i równomierne. Pozwoli to filmowanej osobie uniknąć cieni pod oczami i refleksów na twarzy.



Mikrofon

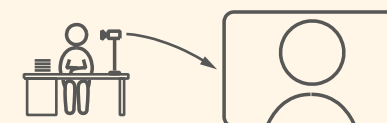
Korzystaj z dodatkowego, zewnętrznego mikrofonu. Poprawi on jakość nagrania.

Tylna kamera

Staraj się korzystać z tylnej kamery smartfona. Zazwyczaj ma ona lepsze parametry od kamery przedniej.

Nagranie z wielu kamer

Jeśli używasz jednocześnie wielu kamer, włącz w każdej z nich nagrywanie obrazu oraz dźwięku. Ułatwisz sobie synchronizację fragmentów.



Przy ujęciach z bliska kamera powinna być na wysokości twoich oczu. Uważaj z gestykulacją, aby dłonie nie pojawiły się w kadrze.

Zadbaj o przestrzeń wokół siebie. Usuń wszystkie niepotrzebne rzeczy, które mogłyby odwracać uwagę od tego, co najważniejsze.

Jeśli nie masz wpływu na swoje otoczenie, ustaw wirtualne tło. Pamiętaj, żeby postać na tle była dobrze oświetlona.



Unikaj kadrowania z funkcją zoomu, ponieważ może to pogorszyć jakość nagrania. Wykorzystaj optyczne możliwości kamery lub przestaw ją tak, aby dopasować kadr.

Podczas nagrań na żywo zabezpiecz przestrzeń przed niepowołanymi osobami. Poproś kogoś, by informował przechodniów o trwającym nagraniu i prosił ich o omijanie filmowanego obszaru.

Praca z mikrofonem

Tworzenie publikacji dźwiękowych oraz filmowych wymaga odpowiedniej pracy z mikrofonem. Aby uzyskać dobre efekty, warto stosować się do kilku podstawowych zasad.

Warto wiedzieć

Mikrofony rejestrują nawet dźwięki ludzkiego oddechu. Takie dźwięki mogą istotnie zakłócać odbiór nagrania.

- ▶ **Przećwicz pracę z mikrofonem.** Wypróbuj różne ustawienia parametrów mikrofonu oraz różne jego położenie. Wiele mikrofonów pozwala wybrać sposób nagrywania (np. dźwięk dookolny lub kierunkowy) oraz czułość. Część parametrów można ustawić w programie do rejestrowania dźwięku. Zazwyczaj powinno się zachować odległość kilkunastu centymetrów między twarzą a mikrofonem. Unikaj zbyt głośnego dźwięku powodującego zniekształcenie nagrania. W razie potrzeby zmniejsz czułość mikrofonu.
- ▶ **Zadbaj o wyciszenie pomieszczenia.** Większość mikrofonów rejestruje **szum tła** (ang. *background noise*), czyli dźwięki pochodzące z otoczenia. Sprawdź, czy tam, gdzie planujesz nagrywać, występują takie dźwięki. Zapewnij sobie spokojne miejsce: zamknij okno, drzwi i poinformuj innych o nagraniu (np. przyklej na drzwiach kartkę z odpowiednim komunikatem). Unikaj echa.
- ▶ **Przygotuj skrypt wypowiedzi.** Nie musi to być dokładny zapis tego, co chcesz powiedzieć, lecz ogólny plan, który pozwoli uniknąć pomyłek i przestojów. Wypowiedź będzie żywsza niż w przypadku odczytania przygotowanego tekstu.
- ▶ **Zadbaj o dykcję i natężenie głosu.** Mów do mikrofonu tak, jak do osoby siedzącej naprzeciwko ciebie przy stole, w głośnym otoczeniu.
- ▶ **Odpowiednio zareaguj na pomyłkę.** Zrób chwilę przerwy i dopiero potem wznów nagranie. Przerwa ułatwi później montaż.

Szum tła

Warto wiedzieć

W telewizji często używa się telepromptera, czyli ekranu, który wyświetla tekst do odczytania przez osobę występującą przed kamerą. Widz odnosi wtedy wrażenie, że tekst jest wygłaszany z pamięci.

Jak mówić do mikrofonu?

Niezależnie od tego, czy wypowiadamy się na filmie czy w materiale audio, pomóc może wyobrażenie sobie osób, które będą nas słuchać. Oprócz tego warto zastosować kilka prostych zasad.

Mów naturalnie



Mów podobnie jak podczas rozmowy: zmieniaj tempo, natężenie i brzmienie głosu. Nie używaj stylu urzędowego i zbyt oficjalnego.

Mów wyraźnie



Lepiej mówić wolniej i ciszej niż zbyt szybko i zbyt głośno. Unikaj zjadania końcówek i wydawania zbędnych dźwięków.

Mów prosto



Dopasuj język do odbiorcy. Jeśli nie jest on sprecyzowany, używaj prostego języka, aby każdy mógł cię zrozumieć.

Mów konkretnie



Używaj krótkich zdań oraz unikaj dygresji. Korzystaj z przygotowanego skryptu.

Nagrywanie i udostępnianie ekranu

Podczas nagrywania operacji wykonywanych na ekranie komputera również warto stosować się do kilku zasad.

Przygotuj ekran komputera



Usuń z pulpitu, paska zadań oraz zasobnika systemowego zbędne dokumenty i ikony. Pozamykaj niepotrzebne okna aplikacji oraz zakładki w przeglądarce internetowej. Ustaw jako tapetę tło w jednolitym, stonowanym kolorze.

Zablokuj powiadomienia



Ustaw tryb pracy komputera tak, aby zablokowane były wszystkie powiadomienia (np. od programu antywirusowego). Zamknij komunikatory, wyłącz automatyczne aktualizacje. Zablokuj wyskakujące okna w przeglądarce internetowej.

Stosuj zasady zachowania się przed kamerą



Jeśli chcesz w materiale uwzględnić siebie, zastosuj się do wskazówek dotyczących wystąpień przed kamerą oraz pamiętaj, aby czasem spojrzeć w obiektyw. Dobrze to zrobić wtedy, kiedy zapowiadasz, co za chwilę zaprezentujesz.

Skup się na wskaźniku myszy



Poruszaj kursorem myszy precyzyjnie – wolnymi, płynnymi ruchami wskazuj wykonywane operacje. Skorzystaj z narzędzi do wygładzania ruchów kursora w programie rejestrującym ekran komputera.

Podziel materiał na mniejsze fragmenty



Materiał zarejestruj jako serię krótkich nagrań. Zaczynaj nagrywać kilka sekund przed właściwą częścią filmu i zakończ kilka sekund po niej. W kolejnych nagraniach staraj się zachować te same ustawienia ekranu i mikrofonu.

Nie spiesz się



Zarezerwuj sobie odpowiednio dużo czasu, zwłaszcza jeśli masz małe doświadczenie w nagrywaniu. Stworzenie jednogodzinowego samouczka może zająć nawet ponad godzinę.

Montaż materiału

Przed opublikowaniem materiału trzeba wykonać jego korektę i montaż. Warto wyregulować dźwięk, usunąć szum i ustabilizować obraz. Do wideo można dodać plansze, napisy początkowe i końcowe, a do audio – zapowiedzi, dżingle i zakończenie.

Do montażu prostych projektów filmowych można wykorzystać oprogramowanie dostarczone wraz z systemem, np. aplikację Clipchamp w Windows, iMovie w macOS czy iPadOS. Przy bardziej złożonych projektach warto sięgać po programy oferujące więcej opcji, np. używany w produkcjach filmowych DaVinci Resolve lub popularny wśród twórców internetowych Capcut. Do pracy nad dźwiękiem może się przydać program Audacity, bezpłatny i dający duże możliwości.

Do nagrywania ekranu oraz podstawowego montażu **screencastu** można wykorzystać usługę Stream (wymagane jest szkolne konto Microsoft 365). Wbudowana w nią funkcja **Nagrywanie ekranu** pozwala nagrać czynności wykonywane na pulpicie lub w wybranej aplikacji, narrację oraz obraz z kamery. Czas nagrania nie może jednak w tym wypadku przekroczyć 15 minut. Opcję nagrywania ekranu mają także aplikacja Clipchamp oraz **narzędzie Wycinanie**.

Dobra rada

Stosuj z umiarem efekty dodatkowe – animacje napisów i przejść między ujęciami oraz dżingle.

Dobra rada

Na urządzeniach mobilnych możesz użyć aplikacji Hokusai Audio Editor (iOS/iPadOS) lub Super Sound (Android).

Screencast

→ s. 56

Warto wiedzieć

W systemach macOS i iPadOS również znajdują się wbudowane narzędzia do nagrywania ekranu.

Innym sposobem jest wykorzystanie wbudowanej w system operacyjny funkcjonalności ukrytej pod skrótem klawiszowym **Windows + G**.

Zapamiętaj

Zapewnienie odbiorcom ciekawej rozrywki lub wartościowej wiedzy może przysłonić niedoskonałości techniczne przygotowanego materiału. Dlatego przed opracowaniem publikacji należy się przede wszystkim zastanowić, jaki jest jej cel.

Dobra rada

Do narzędzi analitycznych możesz sięgnąć również na etapie planowania publikacji. Pozwolą one m.in. odnaleźć słowa często wyszukiwane przez internautów.

3.5. Jak poznać (lepiej) swojego odbiorcę?

Przed przygotowaniem publikacji internetowej przyjmujemy wiele założeń na jej temat. Jednym z najważniejszych jest pytanie o głównego odbiorcę. Nie zawsze nasze założenia i wynikające z nich działania odniosą zamierzony skutek. Dlatego warto śledzić efekty i wyciągać wnioski – poprawiać zamieszczone treści i uwzględniać nową wiedzę podczas przygotowywania kolejnych materiałów.

Do zbadania tego, w jaki sposób nasza publikacja oddziałuje na odbiorców, możemy wykorzystać gotowe narzędzia. Przedstawia je tabela 3.3.

Narzędzie	Przeznaczenie
Google Trends	Pozwala sprawdzać, które hasła są najczęściej wyszukiwane w sieci, oraz badać zmiany ich popularności w czasie. Na podstawie tych wyników można zidentyfikować poszukiwane przez internautów słowa i używać ich we własnych tekstach, np. w opisach, tytułach, lidach czy w głównej treści publikacji.
Serwisy analityczne (np. Google Analytics, stat.4u.pl)	Służą do zbierania informacji o osobach odwiedzających dane serwisy. Można się z nich dowiedzieć więcej o swoich odbiorcach, m.in. poznać ich wiek, narodowość, miejsce, z którego łączą się z serwisem, sposób dotarcia do naszych treści (przez wyszukiwanie, bezpośrednio wpisanie adresu czy link z innego serwisu), czas spędzony w serwisie, odwiedzane podstrony.
Narzędzia badające Internet (np. Mediapanel)	Warto korzystać z informacji i raportów przekazywanych przez firmy badające Internet, przepływy odbiorców oraz trendy. Informacje te są skierowane głównie do reklamodawców, ale mogą być również przydatne przy rozwijaniu własnego serwisu.

Tabela 3.3. Narzędzia analizujące zasoby Internetu

Łączenie informacji z różnych źródeł oraz śledzenie trendów wyszukiwania pomoże przygotowywać treści nie tylko lepiej odpowiadające oczekiwaniom odbiorców, lecz także łatwiejsze do znalezienia w gąszczu informacji.

Zadania

- * 1 Poszukaj serwisów oferujących gotowe dżingle, które można wykorzystać we własnych nagraniach.
- * 2 Sprawdź, czy i w jaki sposób możesz spowodować, by publikowane przez ciebie nagrania w różnych serwisach wideo (np. YouTube, Vimeo) były odtwarzane bez reklam.
- * 3 Znajdź informacje o tym, jakie tematy i hasła były najczęściej wyszukiwane w ciągu ostatnich 12 miesięcy w Polsce w kategorii: edukacja. Zastanów się, jak możesz wykorzystać tego typu analizy do przygotowania własnych treści na określony temat.
- ** 4 Przygotuj logotyp oraz przewodnik dotyczący identyfikacji wizualnej dla strony internetowej lub bloga na wybrany przez siebie temat. Przewodnik poza logiem powinien obejmować m.in. opis zalecanych czcionek (wraz z ich przykładami), opis kolorów i ich zastosowań (tła, teksty) oraz innych elementów. Podczas pracy wykorzystaj narzędzia sztucznej inteligencji.
- ** 5 Przygotuj grafikę do wykorzystania w sieciach społecznościowych reklamującą serię webinarów lub blog na interesujący cię temat. Załóż, że grafika będzie kierowana do twoich kolegów.
- ** 6 Przygotuj tekst, który będzie relacją z wybranego przez ciebie webinarium lub podcastu. Tekst powinien zachęcić twoich rówieśników do odsłuchania nagrania.
- ** 7 Poszukaj i użyj bezpłatnego narzędzia (serwisu) wykorzystującego sztuczną inteligencję i pozwalającego wygenerować podkład muzyczny (bez słów), który można by było wykorzystać w samouczku wideo (screencaście). Podkład powinien być dopasowany dynamiką do samouczka, nie zagłuszać ewentualnego głosu lektora, nie być zbyt monotony i nie wywoływać zdenerwowania odbiorcy.
- *** 8 Przygotuj podcast na wybrany temat. Opracuj towarzyszące mu materiały promocyjne: zapowiedź tekstową i grafiki do zamieszczenia w sieciach społecznościowych lub na stronach WWW.
- *** 9 Przygotuj webinarium na wybrany temat związany z edukacją. Opracuj towarzyszące publikacji materiały promocyjne (zapowiedź tekstową, grafiki do zamieszczenia w sieciach społecznościowych lub na stronach WWW).
- *** 10 Opracuj samouczek wideo (screencast) na temat użycia wybranego przez ciebie narzędzia wykorzystującego sztuczną inteligencję. Samouczek powinien przedstawiać problematykę w sposób zrozumiały dla starszych osób i do nich być skierowany.

Tworzenie treści internetowych

Wyobraź sobie, że...

...postanawiasz zostać front-end developerem. Liczne ogłoszenia o pracę zachęciły cię, by zająć się zawodowo programowaniem stron internetowych. Okazuje się, że pierwszy krok masz już za sobą. Spodobały ci się lekcje z **języka HTML** i **arkuszy CSS**, a znajomość **systemów CMS** jest przez przyszłych pracodawców mile widziana. Wiesz, że to odpowiednia dla ciebie ścieżka zawodowa. Jeśli będziesz doskonalić swoje umiejętności w tym zakresie i poznasz więcej technologii tworzenia stron WWW, masz spore szanse na ciekawą, rozwijającą i dobrze płatną pracę.



...pracujesz w dziale promocji miasta. Kierujesz zespołem przygotowującym kampanię, która ma przyciągnąć turystów oraz zachęcić mieszkańców do odkrywania miasta na nowo. Z pomocą grafików opracowujecie **identyfikację wizualną**, m.in. rozpoznawalne **logo**, które od tej pory będzie widnieć na wszystkich materiałach informacyjnych. Kręcicie także spot promocyjny do mediów społecznościowych i wymyślcie oryginalne hasło, które rozpowszechnicie w formie **hashtagu**.



...postanawiasz założyć kanał w popularnym serwisie wideo, żeby dzielić się wiedzą na temat kultur różnych krajów. Przygotowujesz **konspekty** filmów, zachęcające **teksty informacyjne** i nagrywasz filmy w formie **podcastów** wideo. Szybko zdobywasz duże grono oglądających, a dzięki ich wsparciu – fundusze na wymarzoną podróż dookoła świata, w czasie której nagrasz jeszcze więcej wartościowego materiału.



...otwierasz własny gabinet stomatologiczny dla dzieci i zamierzasz zareklamować go w Internecie. Nie czujesz się na siłach, aby napisać całą stronę od początku, i postanawiasz wykorzystać narzędzie do **wizualnego tworzenia treści**. Z łatwością budujesz stronę z gotowych **segmentów**, w których umieszczasz zdjęcia gabinetu, informacje kontaktowe, mapę i cennik usług. Na koniec przygotowujesz stronę kontaktową i umieszczasz na niej mapkę z dojazdem oraz numer telefonu i adres e-mailowy.



2. Relacyjne bazy danych

4 Projektowanie bazy danych

Portale społecznościowe i serwisy internetowe przetwarzają olbrzymie ilości informacji. Trzeba je gdzieś przechowywać i katalogować, a także zdecydować, które z nich wyświetlać użytkownikowi i w jakiej kolejności. Bez bazy danych i zaawansowanych algorytmów byłoby to niemal niemożliwe. W profesjonalnych zastosowaniach baz danych dominuje model relacyjny – w tym temacie będzie mowa o tym, na czym on polega i jak zaprojektować własną relacyjną bazę danych.

Ważne w tym temacie

- Definicja bazy danych
- Zastosowania relacyjnych baz danych
- Przechowywanie danych w tabelach
- Relacje między tabelami
- Przygotowanie projektu bazy danych

4.1. Zastosowania baz danych

Baza danych ◉ **Bazą danych** będziemy nazywać zbiór uporządkowanych i powiązanych ze sobą informacji dotyczących pewnej dziedziny życia, zorganizowanych w taki sposób, aby łatwo można było je przeszukiwać, wyciągać z nich wnioski i na tej podstawie podejmować decyzje. Tabela 4.1 przedstawia typowe obszary zastosowania baz danych.

Obszar zastosowania	Przykłady wykorzystania
Biznes	▶ przechowywanie informacji o klientach, zamówieniach, produktach i transakcjach w firmach ▶ zarządzanie danymi kadrowymi, płacowymi i administracyjnymi w organizacjach ▶ utrzymywanie danych dotyczących zapasów, dostaw, logistyki i zarządzania łańcuchem dostaw
Aplikacje internetowe	▶ przechowywanie informacji o użytkownikach, treściach, komentarzach itp. na stronach i portalach internetowych ▶ zarządzanie produktami, zamówieniami, płatnościami i informacjami o klientach w systemach <i>e-commerce</i>
Systemy informatyczne w instytucjach edukacyjnych	▶ utrzymywanie danych o uczniach, ocenach, planach zajęć i nauczycielach ▶ przechowywanie informacji o dostępnych książkach, ich lokalizacji i historii wypożyczeń w systemach bibliotecznych
Opieka zdrowotna	▶ przechowywanie danych medycznych pacjentów, historii chorób, wyników badań i informacji o lekach ▶ planowanie zasobów, zarządzanie personelem i monitorowanie stanu pacjentów w systemach zarządzania szpitalami
Aplikacje naukowe i badawcze	▶ przechowywanie danych eksperymentalnych, wyników badań i analiz naukowych

Tabela 4.1. Przykładowe zastosowania baz danych

4.2. Tabele i relacje w bazie danych

Załóżmy, że prowadzimy w formie bloga serwis internetowy o nowoczesnych technologiach. Aby utrzymać stałych czytelników oraz pozyskać nowych, powinniśmy regularnie publikować ciekawe i dobrze opracowane artykuły.

Utworzenie bazy danych w tym wypadku pozwoliłoby:

- ▶ gromadzić dane dotyczące opublikowanych i planowanych artykułów,
- ▶ kontrolować terminowe przygotowanie artykułów,
- ▶ przechowywać dane kontaktowe autorów,
- ▶ sprawdzać, czy długość artykułów zgadza się z ustaleniami.

Aby to wszystko było możliwe, trzeba się zastanowić, które informacje należałoby zebrać w bazie danych. Mogą to być m.in. tytuł, liczba stron i termin dostarczenia artykułu, nazwa działu, w którym ukaże się artykuł, imię i nazwisko autora oraz kontakt do niego, a także informacja, czy tekst został dostarczony.

Tabela, w której gromadzilibyśmy wymienione dane, mogłaby wyglądać tak jak na rysunku 4.1. Zostały w niej umieszczone informacje, które wydają się niezbędne, aby skutecznie zarządzać pracą nad tekstami na blog.

Lp.	Tytuł	Strony	Termin	Dział	Dostarczono	Imię	Nazwisko	Telefon	E-mail
1	Sztuczna inteligencja	6	11.10.2025	Nowe technologie	Tak	Jan	Kowalski	991123123	jkow@mail.com
2	Instagram jako narzędzie marki osobistej	4	13.10.2025	Samouczki	Nie	Eliza	Makulska	991231231	emaku@mail.com
3	Autonomiczne pojazdy – analiza SWOT	3	12.10.2025	Rynek	Nie	Jan	Kowalski	991123122	jkow@mail.com

Rys. 4.1. Tabela z danymi dotyczącymi bloga

Jednak gdy uzupełniamy tak zaprojektowaną tabelę, możemy zauważyć, że w rzeczywistości przechowujemy w niej informacje o dwóch rodzajach obiektów: artykułach oraz autorach. Łączenie informacji o nich w taki sposób, jak na rysunku 4.1, stwarza kilka problemów.

Problem 1. Dodawanie kolejnych artykułów jednego autora wymaga wprowadzania za każdym razem tych samych informacji. Zajmuje to czas oraz miejsce, a poza tym łatwo popełnić błąd (np. w numerze telefonu Jana Kowalskiego, jak na rysunku 4.1).

Problem 2. Jeśli wprowadzimy błędne dane dotyczące tego samego autora, trudno potem rozstrzygnąć, które są właściwe.

Problem 3. Jeśli autor zmieni swoje dane (np. adres e-mail), trzeba będzie tę zmianę wprowadzić przy wszystkich jego artykułach.

Warto wiedzieć

Tematyka serwisu internetowego i związanej z nim bazy danych nie ma większego znaczenia. Projekt bazy po niewielkich modyfikacjach może również posłużyć do prowadzenia np. gazetki szkolnej, bloga modowego lub profilu szkoły w mediach społecznościowych.

Warto wiedzieć

Ten sam tekst napisany na komputerze może mieć różną liczbę stron, zależnie od szerokości marginesów, wybranej czcionki, jej wielkości itp. Dlatego do rozliczeń wykorzystuje się znormalizowaną stronę maszynopisu. Zwyczajowo zawiera ona 1800 zzs (znaków ze spacjami).

Projektując bazę danych, powinniśmy więc unikać możliwości powielania informacji o tym samym obiekcie. Dlatego w naszej sytuacji należy oddzielić dane dotyczące autorów od tych dotyczących artykułów. Można w tym celu utworzyć dwie tabele, które są ze sobą powiązane, np. takie jak na rysunkach 4.2 i 4.3.

ID	Tytuł	Strony	Termin	Dział	Dostarczono	Autor
1	Sztuczna inteligencja	6	11.10.2025	Nowe technologie	Tak	1
2	Instagram jako narzędzie marki osobistej	4	13.10.2025	Samouczki	Nie	2
3	Autonomiczne pojazdy – analiza SWOT	3	12.10.2025	Rynek	Nie	1

Rys. 4.2. Tabela Artykuły

ID	Imię	Nazwisko	Telefon	E-mail
1	Jan	Kowalski	991123123	jkow@mail.com
2	Eliza	Makulska	991231231	emaku@mail.com

Rys. 4.3. Tabela Autorzy

Dobra rada

W tabelach bazy danych zadbaj o to, aby każdy wiersz dotyczył tylko jednego obiektu oraz aby ten sam obiekt nie występował w dwóch różnych wierszach.

Tabela z rysunku 4.1 została więc podzielona na dwie tabele o nazwach Artykuły i Autorzy. Tabela Artykuły zawiera teraz dane ściśle powiązane z artykułami, w tym informację o autorze. Dane autorów zostały w niej jednak zapisane nie wprost, lecz za pomocą liczb, które są identyfikatorami w tabeli Autorzy. Dlatego też zmieniliśmy nazwę pierwszej kolumny tabeli Autorzy z Lp. na ID (skrót słowa *identyfikator*).

W tabeli Autorzy znajdują się teraz wszystkie dane dotyczące poszczególnych autorów. Nie trzeba, a nawet nie powinno się umieszczać w niej dwa razy jednej osoby (w naszym przypadku: Jana Kowalskiego). Zakładamy przy tym, że jeden artykuł ma jednego autora, a przynajmniej jednego autora głównego, który reprezentuje zespół autorów.

W opisany powyżej sposób zaprojektowaliśmy dwie tabele bazy danych. W **tabeli bazy danych** (ang. *database table*) są gromadzone informacje na temat obiektów jednego, określonego typu. Obiekty te są opisywane za pomocą cech podanych jako wartości w kolumnach tabeli.

Atrybut W bazach danych kolumny tabeli nazywa się **atrybutami** (ang. *attributes*). Nazwy tych atrybutów są nagłówkami kolumn – w naszym przykładzie to m.in. Tytuł, Termin czy Dział.

Każdy wiersz tabeli zawiera informacje o jednym obiekcie i jest nazywany **rekordem** (ang. *data record*). Każdy rekord składa się z wielu **pól rekordu** (ang. *values*), które znajdują się na przecięciu rekordu z wybranym atrybutem.

Rysunek 4.4 przedstawia te pojęcia na przykładzie tabeli Artykuły.

ID	Tytuł	Strony	Termin	Dział	Dostarczono	Autor
1	Sztuczna inteligencja	6	11.10.2025	Nowe technologie	Tak	1
2	Instagram jako narzędzie marki osobistej	4	13.10.2025	Samouczki	Nie	2
3	Autonomiczne pojazdy – analiza SWOT	3	12.10.2025	Rynek	Nie	1

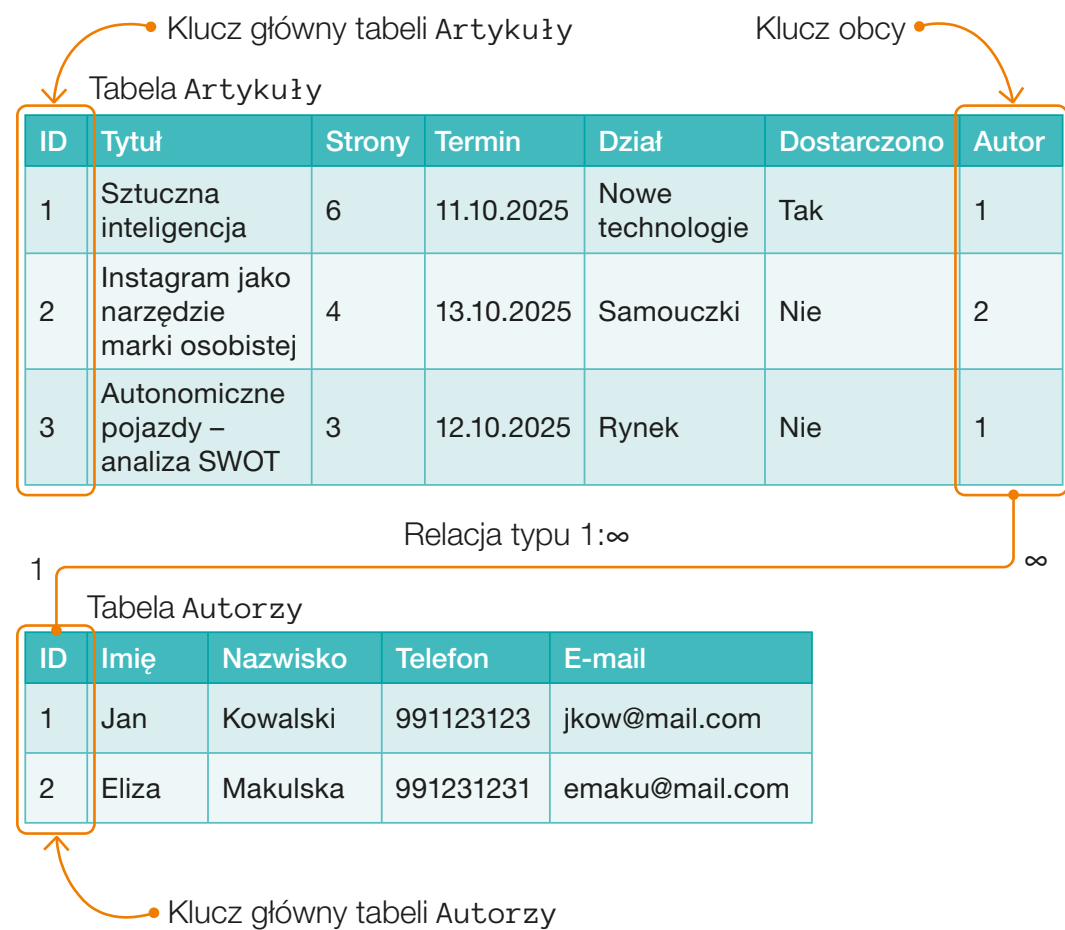
Rys. 4.4. Elementy składowe tabeli bazy danych na przykładzie tabeli Artykuły

Atrybuty o nazwie ID, znajdujące się w obu tabelach, pełnią funkcję **klucza głównego** (ang. *primary key*), nazywanego również **kluczem podstawowym**, czyli niepowtarzalnego identyfikatora, który jednoznacznie wskazuje konkretny rekord danej tabeli. Klucz główny może się składać z jednej lub wielu kolumn i nie musi przechowywać wartości liczbowych. Niepowtarzalność klucza to jego bardzo ważna cecha. W naszym przykładzie daje pewność, że odwołując się do konkretnego numeru ID w tabeli, zawsze mówimy o tym samym rekordzie. Kiedy usuniemy rekord z tabeli, nie powinniśmy dodawać nowego rekordu z numerem ID takim samym jak ten usunięty. Przy rozbudowanych bazach mogłoby to prowadzić do zaburzenia spójności danych, czyli pewności, że są one kompletne i zgodne z przyjętymi założeniami. Gdybyśmy w naszej tabeli Autorzy chcieli usunąć rekord przechowujący informacje o Elizie Makulskiej i dodać nowego autora, powinniśmy nadać mu ID o wartości 3 (rys. 4.5).

ID	Imię	Nazwisko	Telefon	E-mail
1	Jan	Kowalski	991123123	jkow@mail.com
3	Michał	Pawłowski	991231237	mpaw@mail.com

Rys. 4.5. Dodanie nowego rekordu po usunięciu innego

Kolumna Autor tabeli Artykuły pełni funkcję **klucza obcego** (ang. *foreign key*), ponieważ wskazuje na klucz główny innej tabeli, w naszym przypadku kolumnę ID tabeli Autorzy (rys. 4.6, s. 70).



Rys. 4.6. Klucze główne i klucz obcy tabel Artykuły i Autorzy

Takie powiązanie między rekordami w dwóch tabelach nazywamy relacją. **Relacja** (ang. *relationship*) to połączenie logiczne między dwiema tabelami, określające, które atrybuty pól są wspólne dla tych tabel.

- Relacja typu jeden do wielu** W tym przypadku mamy do czynienia z **relacją typu jeden do wielu** (zapisywaną jako 1:∞), ponieważ:
- ▶ jeden autor może napisać wiele artykułów, tzn. rekord z tabeli Autorzy może być powiązany z wieloma rekordami w tabeli Artykuły,
 - ▶ zakładamy, że jeden artykuł ma tylko jednego autora, tzn. jednemu rekordowi z tabeli Artykuły odpowiada tylko jeden rekord w tabeli Autorzy.

A to ciekawe

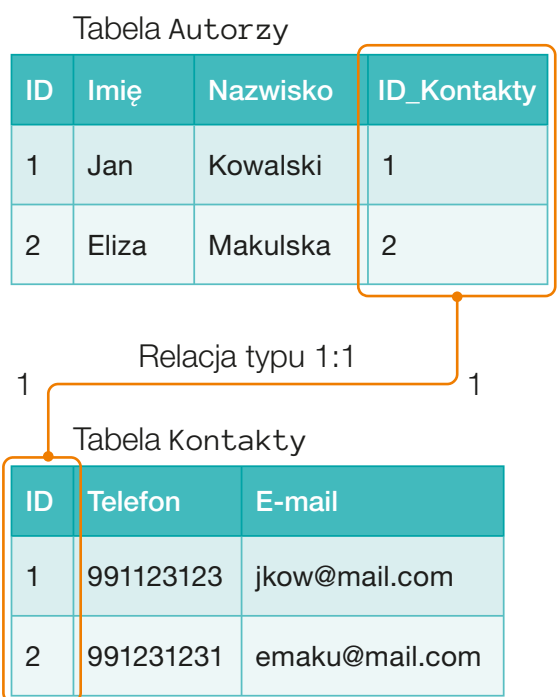
Bazy danych na początku XX w.

Przykładem dawnej bazy danych jest archiwum czeskiego Zakładu Ubezpieczeń Społecznych. Aby dostać się do wyżej położonych dokumentów, urzędnicy korzystali z mechanicznych wind wyposażonych w biurka, krzesła i niezbędne do pracy przybory. Wtedy było to bardzo nowoczesne i praktyczne rozwiązanie, pozwalające zaoszczędzić mnóstwo czasu dzięki wykonywaniu pracy bezpośrednio przy miejscu przechowywania dokumentów i bez przenoszenia ich do oddzielnego biura.



Relacja jeden do wielu jest często występującym typem relacji. Może być zastosowana w różnorodnych sytuacjach, w których pojedyncza informacja z jednej tabeli jest powiązana z wieloma informacjami z innej tabeli, np. jedna klasa jest powiązana z wieloma uczniami, którzy są do niej przypisani.

Oprócz relacji jeden do wielu istnieją także relacje jeden do jednego i wiele do wielu. W **relacji typu jeden do jednego** (1:1) jeden rekord tabeli odpowiada dokładnie jednemu rekordowi innej tabeli. Relację tego typu wprowadza się czasem z powodu ograniczeń technicznych, np. kiedy system zarządzania relacyjną bazą danych nie pozwala zgromadzić wymaganej liczby atrybutów w jednej tabeli. Możemy też wykorzystać taką relację, aby część informacji o zbiorze danych przechowywać w oddzielnej tabeli (rys. 4.7).



Rys. 4.7. Relacja typu jeden do jednego

W **relacji typu wiele do wielu** (∞:∞) wiele rekordów jednej tabeli może odpowiadać wielu rekordom innej tabeli. Ta relacja jest powszechna w bazach danych obsługujących sklepy internetowe. Jeden klient takiego sklepu może zamówić wiele produktów, a jeden rodzaj produktu może być zamówiony przez wielu klientów. Taka relacja sprawdziłaby się również w naszej bazie, jeśli przyjęlibyśmy założenie, że każdy autor może napisać wiele artykułów, a jeden artykuł może być współtworzony przez wielu autorów. Na potrzeby relacji ∞:∞ tworzy się dodatkową tabelę, w której są przechowywane pary kluczów głównych tabel powiązanych ze sobą. Na rysunku 4.8 na następnej stronie pokazano przypisanie dwóch autorów do jednego artykułu. Widać na nim również, że jeden autor ma do napisania kilka artykułów. Dodając kolejne rekordy do tabeli pośredniej, możemy dalej rozszerzać te przypisania.

Relacja typu jeden do jednego

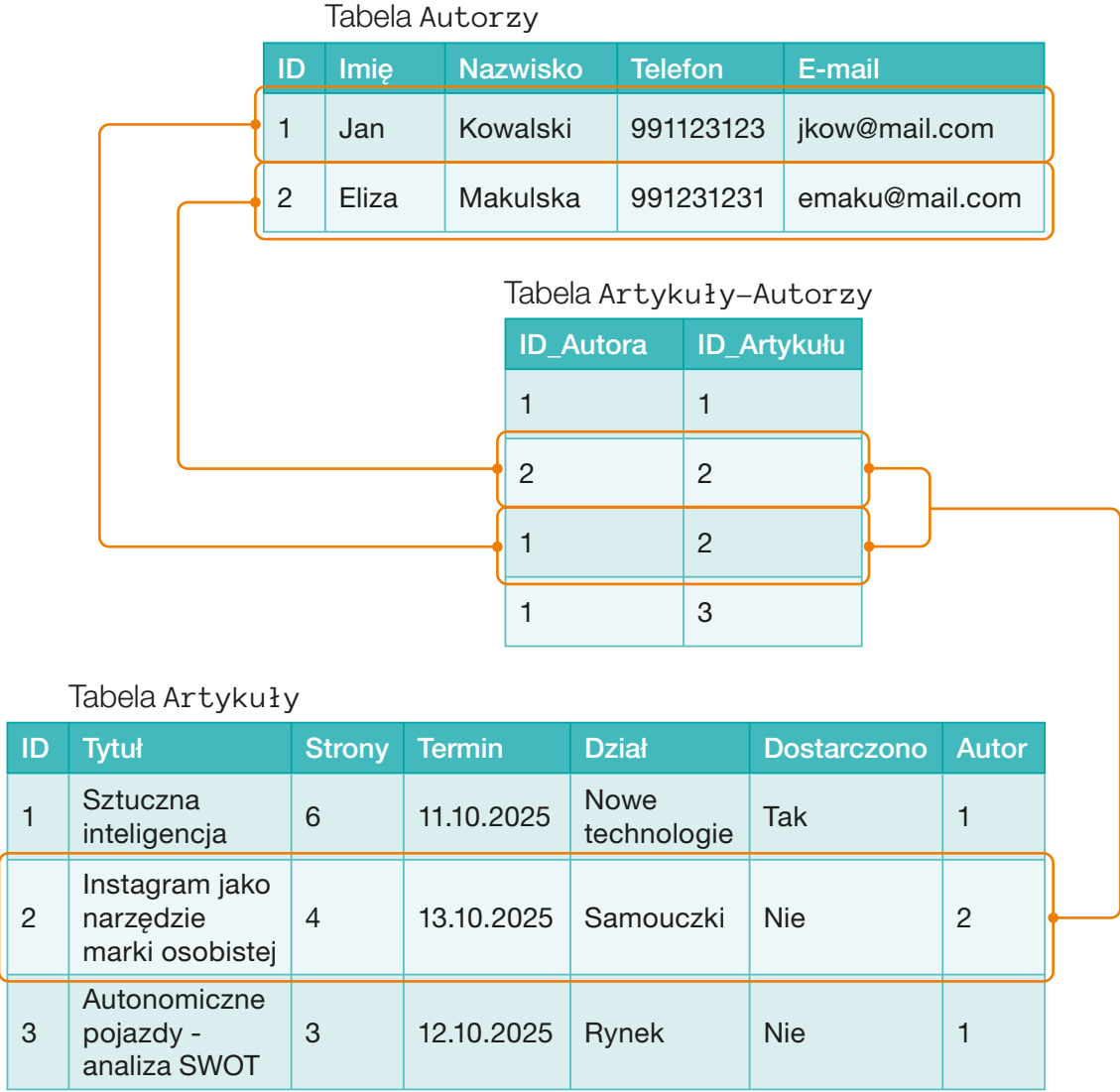
Warto wiedzieć

Nazwy kolumn zawierających wartości klucza obcego często tworzy się z wykorzystaniem skrótu ID oraz nazwy tabeli, do której klucza głównego się odnoszą. Na przykład kolumna o nazwie ID_Autora zawiera klucze obce odnoszące się do rekordów z tabeli Autorzy.

Relacja typu wiele do wielu

Warto wiedzieć

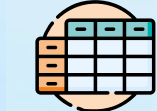
W relacyjnych bazach danych często tabelę nazywa się relacją, ponieważ wiąże ona cechy opisujące określony obiekt.



Rys. 4.8. Relacja typu wiele do wielu – przykład artykułu pisanego przez dwóch autorów

ZASADY PROJEKTOWANIA BAZ DANYCH

Projektując bazę danych, warto pamiętać o kilku zasadach. Dzięki nim zamiana projektu w rzeczywistą bazę odbędzie się bezproblemowo, a korzystanie z danych w niej zawartych będzie znacznie łatwiejsze.



Tworzenie tabel danych

- Zadbaj o to, by informacje o każdym określonym obiekcie przechowywać w oddzielnej tabeli (np. informacje o autorach powinny się znajdować w innej tabeli niż informacje o tworzonych przez nich dziełach).
- Informacje o danym obiekcie (np. autorze) powinny być wprowadzone tylko raz w danej tabeli.
- Projektując tabelę, zadbaj o to, by zestaw przewidzianych w niej atrybutów (pól) pozwalał jednoznacznie zidentyfikować każdy obiekt. Na przykład tabela zawierająca dane osobowe autorów nie powinna składać się tylko z pól Imię oraz Nazwisko, ponieważ istnieje duża szansa, że dwóch różnych autorów będzie miało identyczne dane w tym zakresie.



Nazwy tabel oraz ich atrybutów (pól)

- Stosuj krótkie, ale znaczące nazwy zarówno w odniesieniu do tabel, jak i ich atrybutów (pól).
- Staraj się tworzyć nazwy konsekwentnie. W przypadku kluczy obcych dodawaj np. przedimek ID, a dalej nazwę tabeli, do której się odnosi dany klucz obcy.
- Przy złożonych projektach warto zapisywać nazwy tabel wielkimi literami, a nazwy pól w tabelach – małymi.
- Należy pamiętać, że niektóre systemy mogą nie pozwalać na stosowanie znaków diakrytycznych (np. ą, ę, ć).



Wybór odpowiednich typów danych

Na etapie projektowania poświęć czas i dobierz odpowiedni typ danych dla każdego atrybutu (pola) tabeli do rodzaju przechowywanych w nim danych. Tym typem może być (np. krótki lub długi tekst, liczba, data). Złe określenie typu danych na etapie projektowania może skutkować koniecznością wprowadzenia czasochłonnych, kosztownych poprawek na etapie wdrożenia i testowania bazy.



Jednoznaczna identyfikacja rekordów

- Zadbaj o to, żeby każda tabela zawierała niepowtarzalny klucz główny dla każdego rekordu danych. Klucz może się składać z kilku pól.
- Wartość klucza głównego usuniętego rekordu nie może być ponownie użyta podczas tworzenia nowego rekordu.

Projektowanie bazy danych

- Projektowanie bazy danych** krok po kroku:
1. Ustal, jakie dane będzie przechowywała twoja baza danych.
 2. Zaplanuj tabele bazy danych wraz z atrybutami.
 3. Wskaż relacje między tabelami oraz ich rodzaje.

Ćwiczenie 1

Zaprojektuj bazę danych dowolnego muzeum, zawierającą rejestr zbiorów. Baza powinna składać się z dwóch tabel: Ekspozycje i Sale. Ustal nazwy atrybutów w tabelach, klucze główne oraz relacje między tabelami. Projekt możesz przygotować na kartce.

Zapamiętaj

Relacyjna baza danych to zorganizowany zbiór tabel połączonych relacjami. Każda tabela powinna opisywać jeden typ obiektu i składać się z kolumn zwanych atrybutami, które są zbiorem cech opisujących ten typ.

 Zadania

- * **1** Zaprojektuj bazę danych dowolnej kolekcji, którą masz w swoim domu (np. książek, gier komputerowych). Uwzględnij w niej nie tylko opis posiadanych przedmiotów, lecz także miejsce ich przechowywania.
- * **2** Podaj przynajmniej dwa obszary zastosowania baz danych niewymienione w tabeli 4.1.
- * **3** Poszukaj w sieci informacji na temat serwisów oferujących dostęp do danych przydatnych na co dzień (np. przepisy kulinarne, dane statystyczne, wyniki sportowe). Zbierz je w tabeli. Podaj tytuł serwisu, link do niego oraz krótkie opisy: jakie dane tam znajdziemy oraz do czego mogą nam się przydać.
- ** **4** Znajdź i opisz przykłady pięciu aplikacji, które wykorzystują bazę danych, są dostępne na smartfon, tablet lub komputer i które ułatwiają nasze codzienne funkcjonowanie. Mogą to być aplikacje z zakresu wykonywania obowiązków, uprawiania hobby lub dbania o zdrowy tryb życia. Podaj link do strony opisującej daną aplikację, przygotuj krótki opis jej funkcjonalności oraz uzasadnij, dlaczego uważasz, że może być przydatna.
- ** **5** Zastanów się, czy bazę danych znanych w twoim smartfonie można zorganizować w postaci jednej tabeli. A może lepszym pomysłem byłoby zastosowanie kilku tabel i relacji między nimi? Rozważ oba przypadki i podaj przemawiające za nimi argumenty. Jak sądzisz, jaki model bazy danych zastosowano w twoim telefonie? Uzasadnij odpowiedź.
- ** **6** Wyjaśnij słuszność zastosowania relacji typu jeden do wielu na wybranym przykładzie bazy danych, innym niż opisany w podręczniku. Uzasadnij konieczność zastosowania w tym przypadku tego typu relacji.
- *** **7** Zaprojektuj dla systemu zarządzania wydarzeniami bazę danych, która będzie przechowywała informacje o wydarzeniach, uczestnikach, lokalizacjach oraz rejestracjach.
- *** **8** Podaj trzy przykłady baz danych, w których podczas projektowania należy uwzględnić relację wiele do wielu.
- *** **9** Zaprojektuj uniwersalną bazę danych różnych przedmiotów, które posiadasz (np. książek, czasopism, gier komputerowych). Weź pod uwagę przedmioty, które zdarza ci się pożyczać innym osobom (rodzinie, przyjaciołom, znajomym). Baza powinna umożliwiać przechowywanie informacji o osobach, które coś od ciebie wypożyczyły, kiedy miało to miejsce, do kiedy przedmiot powinien wrócić oraz kiedy faktycznie został zwrócony.

5 **Tworzenie bazy danych**

Gdy mamy gotowy projekt bazy danych, możemy przejść do jej utworzenia w jednym z wielu dostępnych narzędzi. Następnie możemy przetestować jej działanie i sprawdzić, czy spełnia ona nasze oczekiwania oraz założenia projektowe. W tym temacie omówimy ten proces, korzystając z programu Microsoft Access, który ma wszystkie funkcje niezbędne do zarządzania danymi, a przy tym prosty interfejs obsługi.

Ważne w tym temacie

- Zarządzanie bazą danych w programie Microsoft Access
- Wprowadzanie, przechowywanie i modyfikowanie informacji w bazie danych
- Tworzenie tabel i relacji między nimi
- Wyciąganie wybranych informacji za pomocą kwerend

5.1. Od projektu do bazy danych

W praktyce do tworzenia baz danych i zarządzania nimi wykorzystuje się **system zarządzania bazą danych – SZBD** (ang. *Database Management System – DBMS*). Jest to program lub zestaw programów umożliwiających korzystanie z relacyjnej bazy danych. Jednym z takich programów jest Microsoft Access.

Dane w programie Microsoft Access są przechowywane w tabelach. Między tabelami można tworzyć relacje, które odpowiadają typom opisanym w poprzednim temacie. Znajdowanie odpowiednich informacji w bazie danych jest możliwe dzięki kwerendom.

5.2. Tabele w programie Access

Podstawowym elementem programu Access jest **tabela**. Są w niej przechowywane dane dotyczące opisywanych obiektów. W poprzednim temacie omówiliśmy projektowanie tabel bazy danych oraz podstawowe pojęcia z nimi związane. Teraz zajmiemy się tworzeniem takich tabel.

Tworzenie tabeli w programie Access krok po kroku:

1. Przejdź do karty **Tworzenie**.
2. W kategorii **Tabele** naciśnij przycisk **Tabela**.
3. Dodaj atrybuty tabeli.
4. Wpisz rekordy do tabeli.

Załóżmy, że chcemy dodać tabelę opisującą autorów, taką jak zaprojektowana na rysunku 4.3 ze s. 68. Na początku powinniśmy zmienić nazwę domyślnie dodawanego atrybutu Identyfikator na ID. Aby zmienić nazwę atrybutu, należy ją kliknąć prawym przyciskiem myszy i z menu kontekstowego wybrać opcję **Zmień nazwę pola**.

 **System zarządzania bazą danych (SZBD)**

 Warto wiedzieć

Aby podkreślić relacyjny charakter systemu zarządzania bazą danych, stosuje się niekiedy skrót SZRBD – system zarządzania relacyjną bazą danych.

 **Tabela programu Access**

 **Tworzenie tabeli w programie Access**

 Warto wiedzieć

W programie Microsoft Access termin *atrybut* zastępuje się niekiedy określeniem *pole tabeli* lub *kolumna tabeli*.

OKNO PROGRAMU MICROSOFT ACCESS

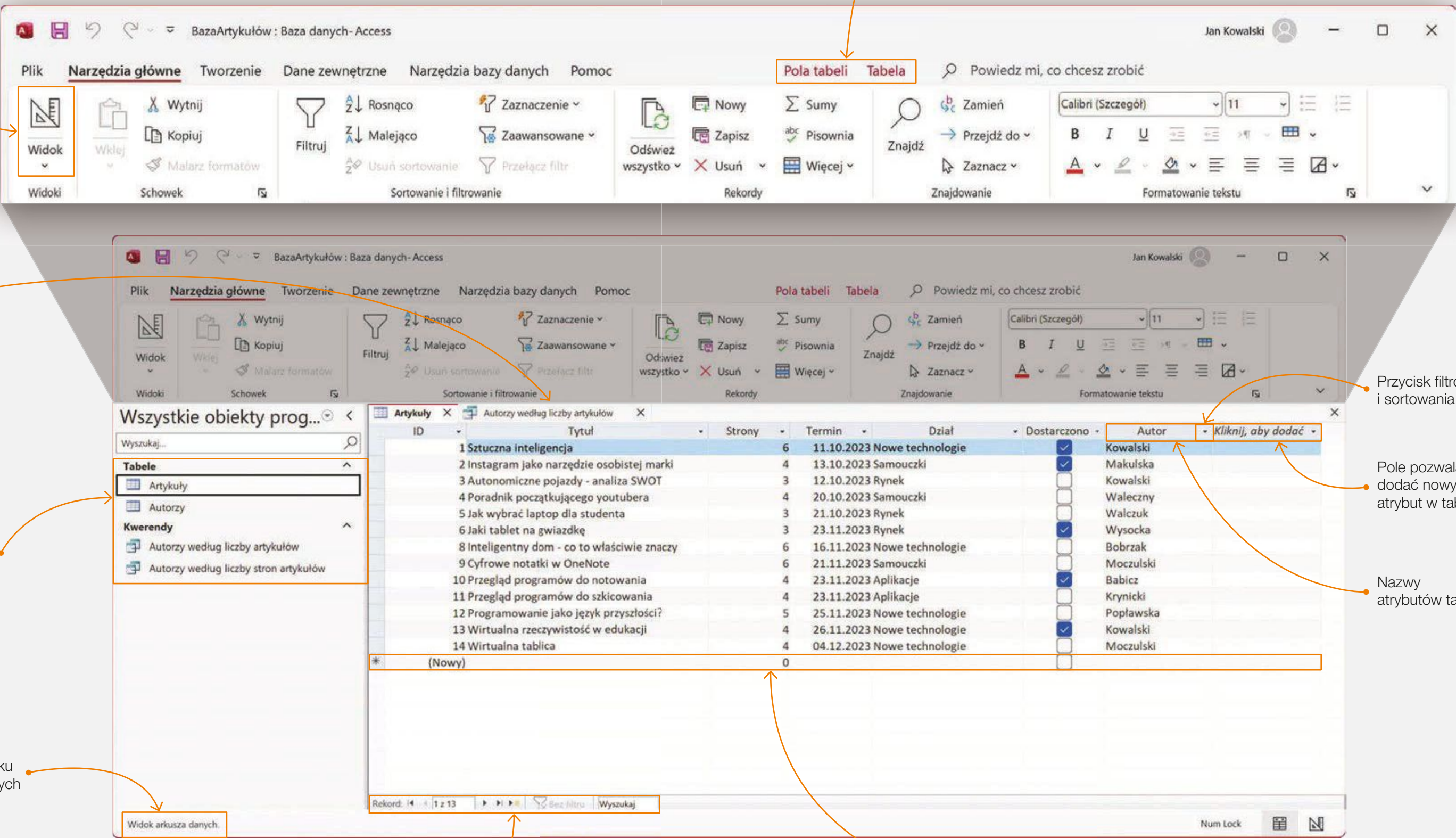
W programie Access na kartach głównych są pogrupowane narzędzia o podobnym przeznaczeniu. Jednym z podstawowych narzędzi jest **Widok**, dzięki któremu można się przełączać między widokiem projektu obiektu bazy (np. tabeli lub kwerendy) a np. widokiem arkusza danych.

Po uaktywnieniu wybranego obiektu (w tym wypadku tabeli Artykuły) pojawiają się dodatkowe karty zawierające narzędzia dotyczące tylko tego obiektu (tu: karty **Pola tabeli** oraz **Tabela**).

W kolejnych zakładkach można przeglądać i modyfikować zawartość elementów bazy danych w różnych trybach widoku. Zakładkę danego obiektu możemy otworzyć po dwukrotnym kliknięciu jego nazwy na liście wszystkich obiektów.

Lista wszystkich obiektów utworzonych w bazie danych

Informacja o aktualnym widoku aktywnego obiektu bazy danych



Przycisk filtrowania i sortowania

Pole pozwalające dodać nowy atrybut w tabeli

Nazwy atrybutów tabeli

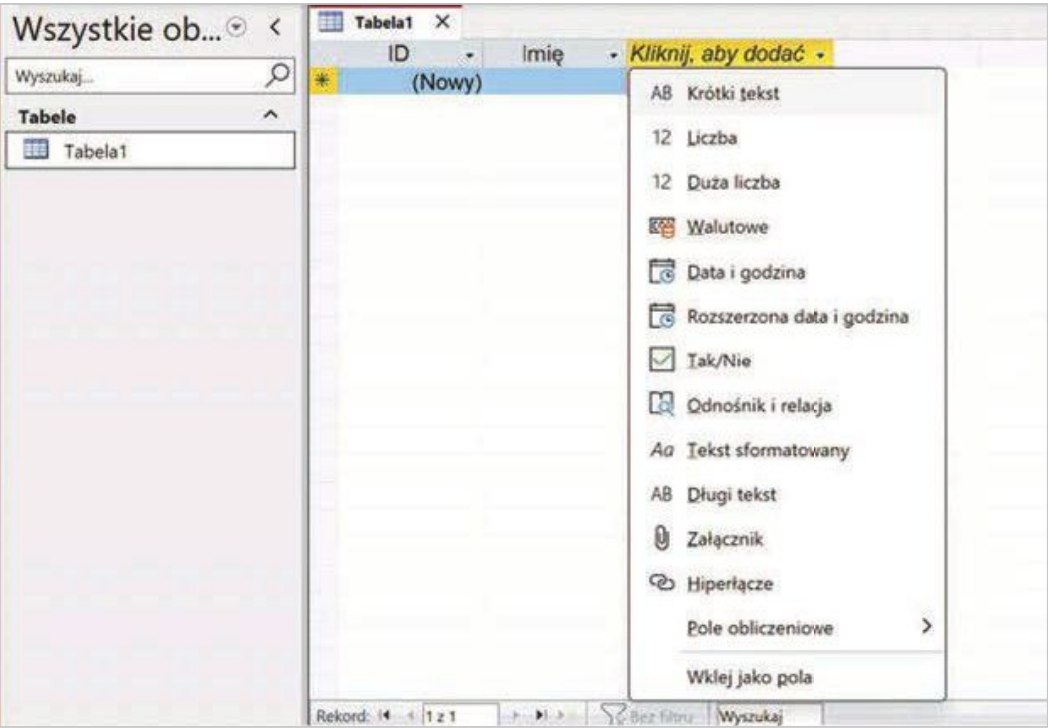
Pasek nawigacji po rekordach tabeli

Wypełnienie ostatniego wiersza tabeli powoduje dodanie do niej nowego rekordu z danymi.

Typy danych i maski w programie Microsoft Access → s. 216

Aby dodać kolejny atrybut, wybieramy opcję **Kliknij, aby dodać**, znajdującą się po prawej stronie ostatniego atrybutu tabeli (w naszym przypadku atrybutu ID). Na rozwiniętej liście wskazujemy typ danych, które będą w nim przechowywane, i wprowadzamy nazwę atrybutu, np. Imię. Projektując tabelę bazy danych i jej atrybuty, należy określić **typ** poszczególnych danych. W przypadku imienia, nazwiska lub adresu e-mail odpowiednie będzie pole tekstowe (**Krótki tekst**) o długości do 255 znaków. Dla numeru telefonu można wybrać pole typu **Liczba**. W tabeli Autorzy wprowadzamy zatem następujące typy atrybutów: Nazwisko (**Krótki tekst**), Telefon (**Liczba**), E-mail (**Krótki tekst**). Na rysunku 5.1 pokazano widok okna programu Microsoft Access podczas tworzenia tabeli i dodawania do niej atrybutów wraz z widoczną listą typów danych.

Warto wiedzieć
W przypadku numeru telefonu ustalenie typu pola nie jest oczywiste. Zazwyczaj wystarczy podać tylko 9 cyfr, a w polu ustawić typ **Liczba**. Dla numerów z zagranicy to jednak za mało, ponieważ trzeba uwzględnić choćby numer kierunkowy kraju (np. +33). Dla tak zapisanych danych można skorzystać z typu **Krótki tekst**.

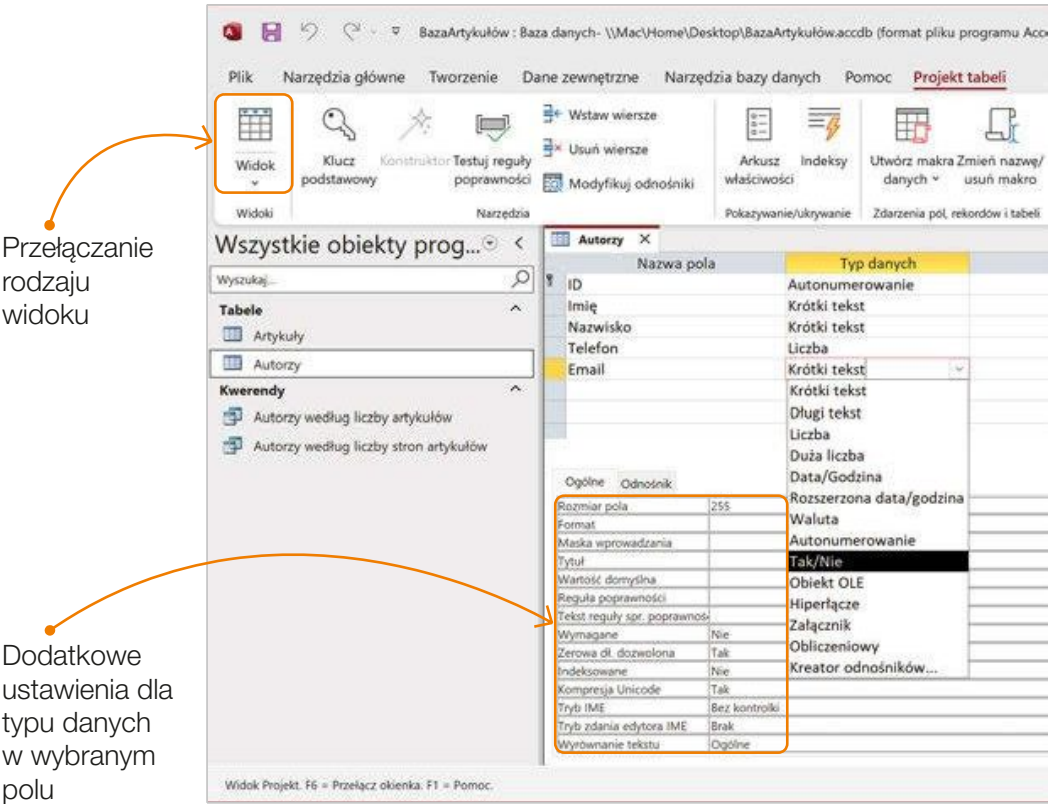


Rys. 5.1. Dodawanie atrybutów do tabeli

Po dodaniu wszystkich kolumn (atrybutów) należy zapisać tabelę. W tym celu możemy kliknąć prawym przyciskiem myszy jej zakładkę, a następnie z menu kontekstowego wybrać opcję **Zapisz**. Ponieważ zapisujemy tabelę po raz pierwszy, pojawi się okno **Zapisywanie jako**, w którym możemy wprowadzić nazwę tabeli – w naszym przypadku Autorzy. Dodawać nowe atrybuty oraz określić typ danych możemy również w **Widoku projektu**. Aby przełączyć się na ten widok, należy zaznaczyć zakładkę danego obiektu (np. tabeli Autorzy), a następnie wybrać **Narzędzia główne** → **Widoki** → **Widok** → **Widok projektu** (rys. 5.2).

Widok projektu

Warto wiedzieć
Po zapisaniu tabeli jej nazwa pojawi się na liście wszystkich obiektów bazy.



Rys. 5.2. Zarządzanie tabelą w Widoku projektu

Pliki bazy danych programu Access są zapisywane w pliku z domyślnym rozszerzeniem *accdb*.

Ćwiczenie 1

W bazie danych udostępnionej przez nauczyciela (np. w pliku *baza_blog_1.accdb*), przy wykorzystaniu narzędzia **Widok projektu**, utwórz tabelę Autorzy z atrybutami odpowiednich typów. Plik zapisz pod nazwą *baza_blog_2.accdb*.

Przeglądanie i modyfikowanie danych w tabelach

Aby ułatwić sobie przeglądanie danych w pojedynczej tabeli, warto korzystać z **filtrowania** i **sortowania**. Do posortowania danych wystarczy zaznaczyć nazwę atrybutu w tabeli, a następnie wybrać **Narzędzia główne** → **Sortowanie i filtrowanie** oraz wskazać sposób

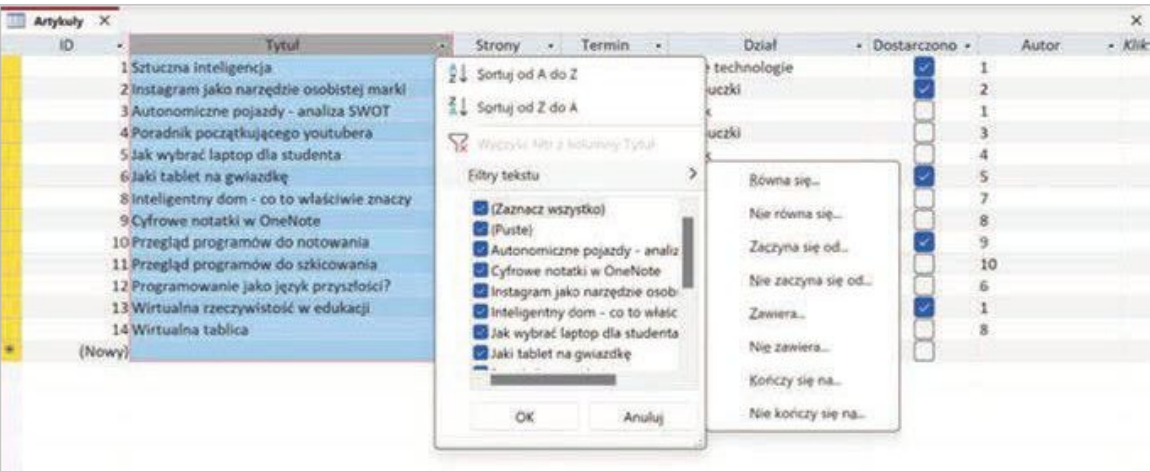
Warto wiedzieć
Program Access pozwala stosować polskie znaki diakrytyczne w nazwach tabel oraz atrybutów, ale nie wszystkie SZBD oferują taką możliwość. Dlatego w bazach danych zazwyczaj unika się polskich znaków.

A to ciekawe

Bazy danych a big data

W biznesie dużą popularność zyskują rozwiązania typu big data, czyli nierelacyjne bazy danych, w których informacje są zapisywane nie w tabelach, ale w plikach o dowolnej strukturze. Takie rozwiązania umożliwiają powiązanie różnorodnych danych, znajdowanie relacji między nimi i wyciąganie wniosków. W ten sposób można np. na podstawie analizy zapytań w wyszukiwarkach internetowych ustalić miejsce wybuchu epidemii grypy lub przewidywać trendy na rynkach finansowych. Nazwa *big data* ma swoje uzasadnienie, ponieważ odnosi się do gigantycznych ilości danych zmieniających się w czasie. Te dane są liczone w tera- lub petabajtach, a do ich analizy często wykorzystuje się systemy sztucznej inteligencji.

sortowania. Możemy także skorzystać z menu podręcznego, wywołując je przez kliknięcie ikony strzałki znajdującej się przy nazwie danego atrybutu w tabeli (rys. 5.3).



Rys. 5.3. Sortowanie danych w programie Microsoft Access

Jeśli chcemy poprawić dane, w otwartej zakładce z wybraną tabelą wskazujemy odpowiednie pole w widoku arkusza danych i wprowadzamy poprawkę lub dodajemy informację, której brakowało.

Zapisywanie zmian w tabeli Aby **zapisać zmiany**, klikamy zakładkę z nazwą tabeli prawym przyciskiem myszy i wybieramy opcję **Zapisz**.

Warto wiedzieć

O tym, że na dane w tabeli został założony filtr, świadczy ikona lejka wyświetlona przy nazwie danej kolumny (obok ikony narzędzia filtrowania).

Ćwiczenie 2

Otwórz plik *baza_blog_2.accdb* utworzony w ćwiczeniu 1., a następnie:

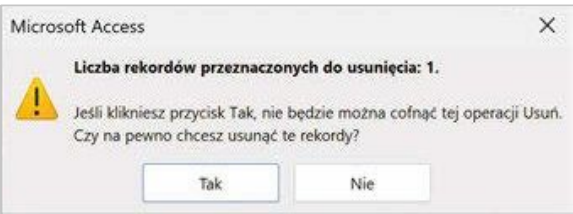
- a. posortuj rekordy tabeli Artykuły od najwcześniejszego terminu dostarczenia utworu,
- b. wyświetl tylko tytuły artykułów, których nie dostarczono,
- c. zmień tytuł artykułu *Sztuczna inteligencja* na *Systemy sztucznej inteligencji*.

Plik bazy zapisz pod nazwą *baza_blog_3.accdb*.

Usuwanie i dodawanie rekordu

Jeśli wiemy, że któryś z artykułów do naszego serwisu nie zostanie napisany, wówczas możemy usunąć z bazy danych cały rekord, który go dotyczy. W tym celu zaznaczamy odpowiadający mu wiersz, a następnie z menu kontekstowego wybieramy opcję **Usuń rekord**. Po potwierdzeniu operacji (rys. 5.4) wiersz zostanie usunięty z tabeli.

Aby dodać nowy rekord, wystarczy uzupełnić pola w pustym wierszu na końcu tabeli. Pole identyfikatora program uzupełni automatycznie.



Rys. 5.4. Potwierdzenie usunięcia rekordu

Ćwiczenie 3

W pliku *baza_blog_3.accdb* z ćwiczenia 2. wykonaj następujące czynności:

- a. usuń z tabeli Artykuły artykuł o ID 4 i tytule *Roboty wokół nas*,
 - b. uzupełnij w tabeli Autorzy listę autorów zgodnie z projektem bazy danych z rysunku 4.5 ze s. 70,
 - c. dodaj nowy artykuł autorstwa Elizy Makulskiej.
- Plik zapisz pod nazwą *baza_blog_4.accdb*.

W bazie danych nie mogą istnieć dwa rekordy z takim samym kluczem głównym. Jeśli dany rekord zostanie usunięty, przynależny do niego klucz główny nie powinien zostać przyporządkowany innemu rekordowi (również nowemu). Gdy w tabeli bazy danych programu Access usuniemy rekord o danym identyfikatorze, to żaden z nowych rekordów nie otrzyma już tego teoretycznie zwolnionego numeru.

Maski wprowadzania i format wyświetlania danych

W wielu sytuacjach określenie typu danych nie wystarczy. W przypadku numeru telefonu zapisanego jako liczba system nie powinien dopuszczać wartości składających się np. wyłącznie z jednej cyfry, ponieważ takie numery telefonów nie istnieją.

Aby zapewnić poprawność informacji wpisywanych do bazy, możemy użyć **maski wprowadzania**, która ma postać ciągu znaków określającego sposób wprowadzania danych. W programie Access maski wprowadzania określa się w **Widoku projektu** wybranej tabeli.

W przypadku numeru telefonu możemy założyć, że składa się on przynajmniej z 9 cyfr, ale nie więcej niż z 15 cyfr. W programie Access taka maska będzie miała postać: 000000000999999, gdzie 0 oznacza cyfrę obowiązkową, a 9 – opcjonalną.

W **Widoku projektu** możemy także zmienić format danych widocznych w polu, m.in. zdecydować, jak będzie wyświetlana data (np. *17.11.2025* czy *poniedziałek, 17 listopada 2025*). **Format** możemy wybrać z listy rozwijanej lub użyć do jego określenia odpowiednich symboli. Aby zobaczyć wprowadzoną zmianę formatu, należy zapisać zmiany w tabeli jeszcze przed przełączeniem się do **Widoku arkusza danych**.

5.3. Definiowanie relacji

W naszej bazie zaplanowaliśmy relację między tabelami **typu jeden do wielu**. Aby utworzyć taką relację, wykorzystamy możliwości programu Access. Dzięki temu będziemy mogli wyświetlać wybrane dane o autorach, mimo że formalnie znajdują się one w odrębnej tabeli.

Warto wiedzieć

Systemy zarządzania relacyjnymi bazami danych zazwyczaj zachowują zasadę niepowtarzalności klucza głównego i uzupełniają go automatycznie.

Maska wprowadzania

Typy danych i maski w programie Microsoft Access
→ s. 218

Relacja typu jeden do wielu
→ s. 70

Kreator odnośników

Do stworzenia relacji w programie Microsoft Access wykorzystamy **Kreator odnośników**, który pozwoli nie tylko powiązać ze sobą pola tabel, lecz także wyświetlać w obu tabelach powiązane ze sobą informacje (rys. 5.8 i 5.9, s. 84).

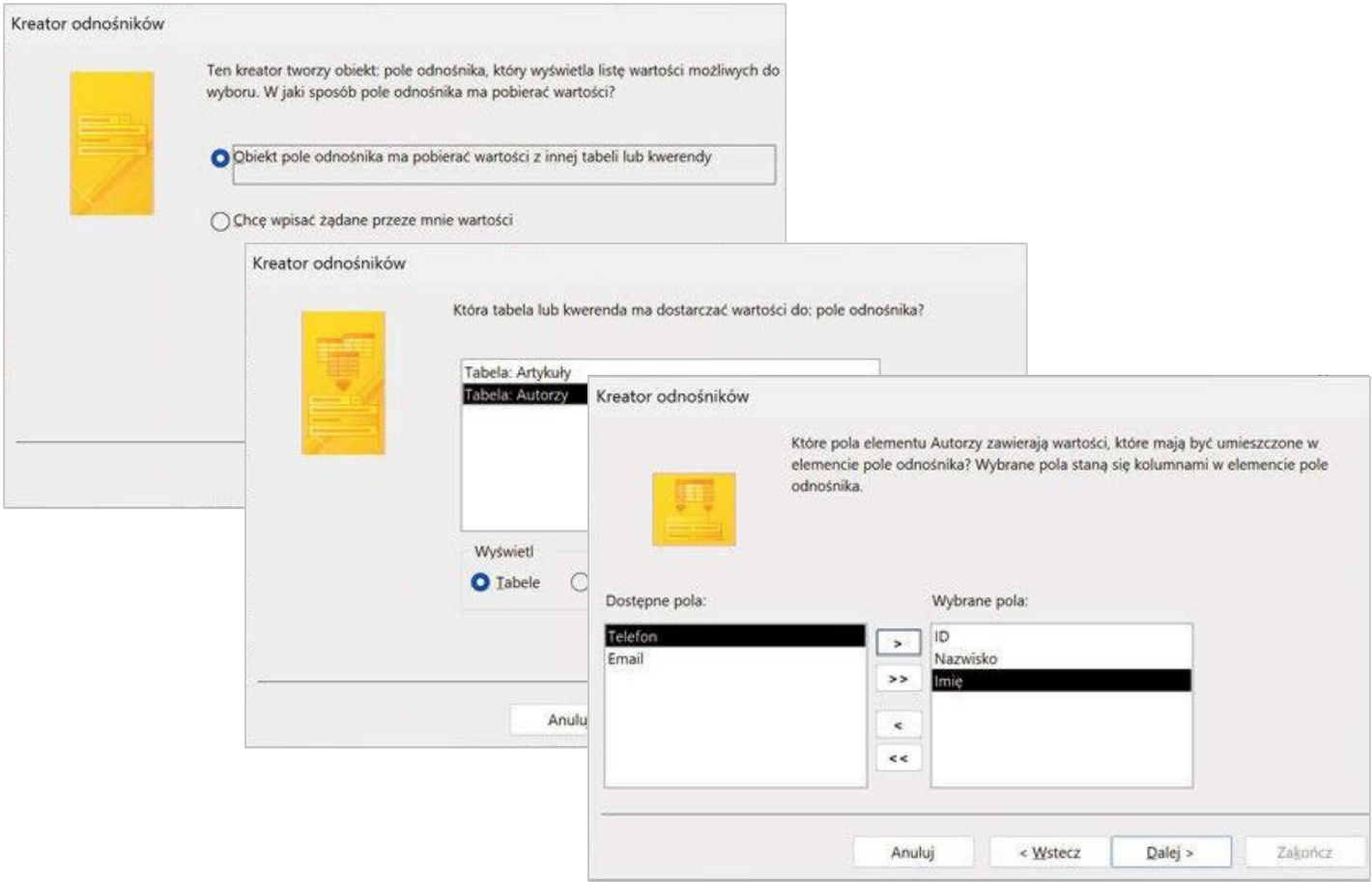
Aby go uruchomić, przechodzimy do **Widoku projektu** dla tabeli Artykuły. Następnie dla pola Autor w kolumnie **Typ danych** (rys. 5.2, s. 79) rozwijamy menu kontekstowe i wybieramy narzędzie **Kreator odnośników**. Najpierw wskazujemy tabelę Autorzy, a następnie pola, których chcemy użyć w tabeli Artykuły (rys. 5.5).

Dodawanie relacji między tabelami

- Dodawanie relacji między tabelami** krok po kroku:
1. Otwórz tabelę, w której chcesz dodać relację do danych z innej tabeli.
 2. Przejdź do **Widoku projektu**.
 3. Dla pola, w którym ma się znaleźć klucz obcy, rozwiń listę **Typ danych** i wybierz z niej opcję **Kreator odnośników**.
 4. Zaznacz opcję **Obiekt pole odnośnika ma pobierać wartości z innej tabeli lub kwerendy** i przejdź dalej.
 5. Wybierz tabelę, z którą stworzysz relację, i przejdź dalej.
 6. Wybierz pola, z których mają być pobierane dane, i przejdź dalej.
 7. Określ, według których pól i w jaki sposób pobrane dane mają być sortowane, i przejdź dalej.
 8. Określ szerokość kolumn i przejdź dalej.
 9. Zakończ działanie kreatora, określając nazwę etykiety odnośnika i dodatkowe parametry.

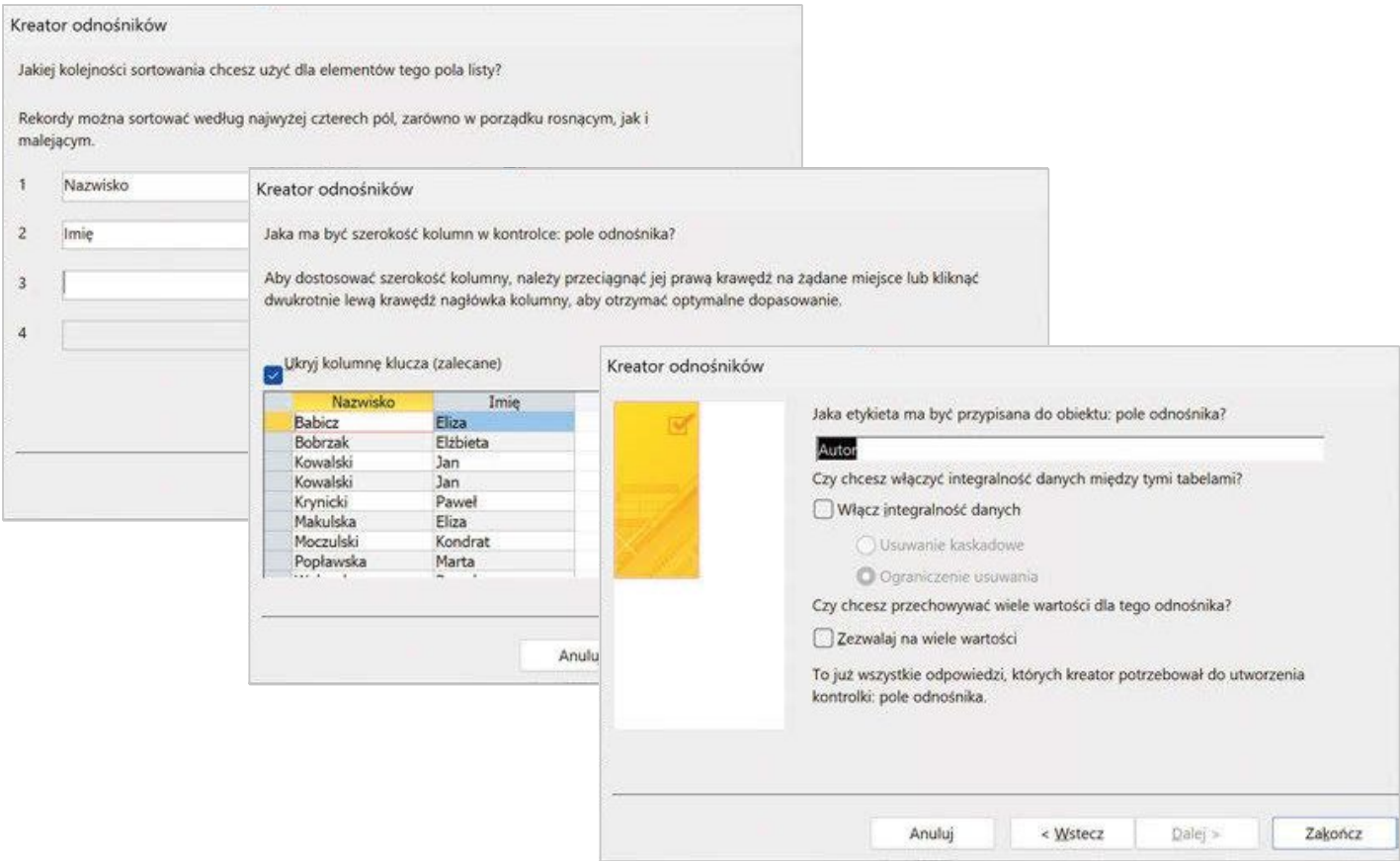
Dobra rada

Jeśli w twojej bazie danych znajdują się dwie osoby o takim samym imieniu i nazwisku, to podczas tworzenia odnośnika zawierającego te dane dodaj dodatkowy atrybut, który będzie je odróżniał, np. adres e-mail.



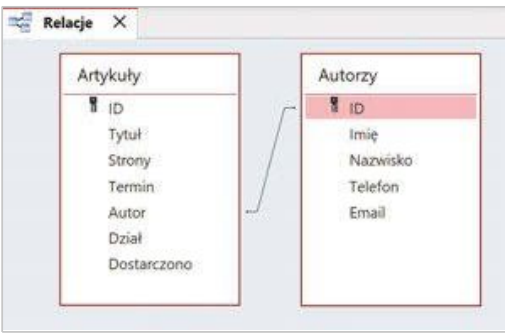
Rys. 5.5. Wybór pól tabeli w Kreatorze odnośników

Po wybraniu pól i ich zatwierdzeniu możemy zdecydować, w jaki sposób zostanie posortowana lista z ich wartościami. Wybieramy atrybuty Nazwisko oraz Imię i w obu przypadkach zostawiamy opcję sortowania rosnącego, które dla tekstu oznacza kolejność alfabetyczną. Ustawiamy również szerokość kolumn i ich kolejność, a także nazwę etykiety (rys. 5.6). Domyślnie pole ID (które jest kluczem obcym) będzie ukryte.



Rys. 5.6. Ustawienia wyświetlania pól w Kreatorze odnośników

Po zapisaniu zmian w tabeli możemy sprawdzić utworzoną relację między tabelami Autorzy oraz Artykuły. W tym celu wybieramy: **Narzędzia bazy danych** → **Relacje** → **Relacje** (rys. 5.7). Powinna być widoczna linia łącząca atrybut Autor tabeli Artykuły z atrybutem ID tabeli Autorzy.



Rys. 5.7. Relacja między tabelami Artykuły i Autorzy

Poza utworzeniem relacji między tabelami w tabeli zawierającej **klucz obcy** pojawią się dane z powiązanej z nią tabeli. Aby to zobaczyć, należy przełączyć się na **Widok arkusza danych**. W kolumnie Autor powinny być widoczne nazwiska autorów zamiast ich identyfikatorów. Po wybraniu pola z nazwiskiem autora powinna się pojawić lista wszystkich autorów. Pokazano to na rysunku 5.8 na następnej stronie.

Dobra rada

Aby zamienić miejscami wybrane kolumny, zaznacz jedną z nich, a następnie chwyć kursorem nazwę kolumny i przeciągnij ją na właściwe miejsce.

Dobra rada

Jeśli po utworzeniu odnośnika kolejność kolumn jest niewłaściwa, kliknij prawym przyciskiem myszy nazwę kolumny, wybierz z menu kontekstowego opcję **Modyfikuj odnośniki** i przejdź do kroku, w którym możesz zmienić kolejność kolumn.

Klucz obcy

→ s. 70

ID	Tytuł	Strony	Termin	Dział	Dostarczone	Autor	Kliknij, aby dodać
1	Sztuczna inteligencja	6	11.10.2023	Nowe technologie	☒	Kowalski	
2	Instagram jako narzędzie osobistej marki	4	13.10.2023	Samouczki	☒	Babicz Eliza	
3	Autonomiczne pojazdy - analiza SWOT	3	12.10.2023	Rynek	☒	Bobrzak Elżbieta	
4	Poradnik początkującego youtubera	4	20.10.2023	Samouczki	☐	Kowalski Jan	
5	Jak wybrać laptop dla studenta	3	21.10.2023	Rynek	☐	Kowalski Jan	
6	Jaki tablet na gwiazdkę	3	23.11.2023	Rynek	☒	Krynicky Pawel	
8	Inteligentny dom - co to właściwie znaczy	6	16.11.2023	Nowe technologie	☐	Makulska Eliza	
9	Cyfrowe notatki w OneNote	6	21.11.2023	Samouczki	☐	Moczulski Kondrat	
10	Przegląd programów do notowania	4	23.11.2023	Aplikacje	☒	Poplawska Marta	
11	Przegląd programów do szkicowania	4	23.11.2023	Aplikacje	☐	Walczuk Pawel	
12	Programowanie jako język przyszłości?	5	25.11.2023	Nowe technologie	☐	Waleczny Piotr	
13	Wirtualna rzeczywistość w edukacji	4	26.11.2023	Nowe technologie	☒	Wierzbicki Janusz	
14	Wirtualna tablica	4	04.12.2023	Nowe technologie	☐	Wysocka Gabriela	
(Nowy)		0					

Rys. 5.8. Widok tabeli Artykuły po powiązaniu z tabelą Autorzy

W **Widoku arkusza danych** tabeli Autorzy przy każdym rekordzie pojawia się znak plusa. Po wybraniu go wyświetlą się wszystkie rekordy z tabeli Artykuły powiązane z danym rekordem tabeli Autorzy (rys. 5.9).

ID	Imię	Nazwisko	Telefon	Email	Kliknij, aby dodać
1	Jan	Kowalski	991123123	janwie@nowaera.pl	
2	Eliza	Makulska	991231231	kowal@nowaera.pl	
3	Piotr	Waleczny	990212121	piowa@nowaera.pl	

ID	Tytuł	Strony	Termin	Dział	Dostarczone	Kliknij, aby d
1	Sztuczna inteligencja	6	11.10.2023	Nowe technologie	☒	
3	Autonomiczne pojazdy - analiza SWOT	3	12.10.2023	Rynek	☐	
13	Wirtualna rzeczywistość w edukacji	4	26.11.2023	Nowe technologie	☒	
(Nowy)		0			☐	

Dane z tabeli Artykuły związane z wybranym autorem

Rys. 5.9. Widok tabeli Autorzy po utworzeniu relacji

Dobra rada

Jeśli dwie tabele twojej bazy danych są związane relacją, to w **Widoku arkusza danych** jednej z tych tabel możesz wprowadzać zmiany dotyczące rekordów drugiej tabeli.

Ćwiczenie 4

W pliku bazy *danychbaza_blog_4.accdb* z ćwiczenia 3. stwórz odnośnik powodujący, że w kolumnie Autor tabeli Artykuły wyświetli się nazwisko autora z odpowiadającego mu rekordu w tabeli Autorzy. Plik zapisz pod nazwą *baza_blog_5.accdb*.

A to ciekawe

Bieżące raportowanie sprzedaży

Jedna z największych firm technologicznych wdrożyła we wszystkich swoich sklepach system, który w czasie rzeczywistym zbiera dane na temat sprzedanych produktów. Zaprojektowano w tym celu specjalną bazę danych. Dzięki temu rozwiązaniu można przewidywać zapotrzebowanie na produkty i wytwarzać je tak, żeby zaspokoić popyt bez ryzyka nadprodukcji.

Zapamiętaj

Za pomocą **Kreatora odnośników** w programie Microsoft Access można jednocześnie stworzyć relację między tabelami bazy danych oraz spowodować, że w polu klucza obcego będą wyświetlane odpowiadające mu dane z tabeli, na którą ten klucz wskazuje.

5.4. Kwerendy – wyszukiwanie i łączenie danych

Kwerenda pozwala znajdować i pobierać z tabel lub innych kwerend dane spełniające określone kryteria. Dzięki kwerendom można odpytywać bazę z informacji dotyczących przechowywanych w niej danych. W naszej bazie utworzymy kwerendę, dzięki której dowiemy się, ile w sumie stron artykułów zobowiązał się dostarczyć każdy z autorów.

Tworzenie kwerendy krok po kroku:

1. Przechodzimy do karty **Tworzenie** i w kategorii **Kwerendy** wybieramy **Projekt kwerendy**.
2. Z panelu **Dodawanie tabeli** wybieramy tabele, które chcemy dodać do projektu kwerendy.
3. Z dodanych tabel wybieramy nazwy pól i przeciągamy je do dolnej części okna projektu.

W naszym przypadku powinniśmy do kwerendy dodać tabele Artykuły oraz Autorzy. Następnie, ponieważ chcemy mieć dane autora, na początek przeciągniemy z tabeli Autorzy pola ID, Imię oraz Nazwisko. W dalszej kolejności z tabeli Artykuły wybierzemy pole Strony, bo to dzięki niemu będzie można obliczyć, ile stron zobowiązał się napisać każdy z autorów. Na rysunku 5.10 widoczne jest tworzenie kwerendy w **Widoku projektu**.

Kwerenda1				
Artykuły ID Tytuł Strony Termin Autor				
Autorzy ID Imię Nazwisko Telefon Email				
Pole:	ID	Imię	Nazwisko	Strony
Tabela:	Autorzy	Autorzy	Autorzy	Artykuły
Sortuj:				
Pokaż:	☒	☒	☒	☒
Kryteria:				
lub:				

Rys. 5.10. Dodawanie tabel i wybieranie z nich pól tworzących kwerendę

Tworzenie kwerendy

Warto wiedzieć

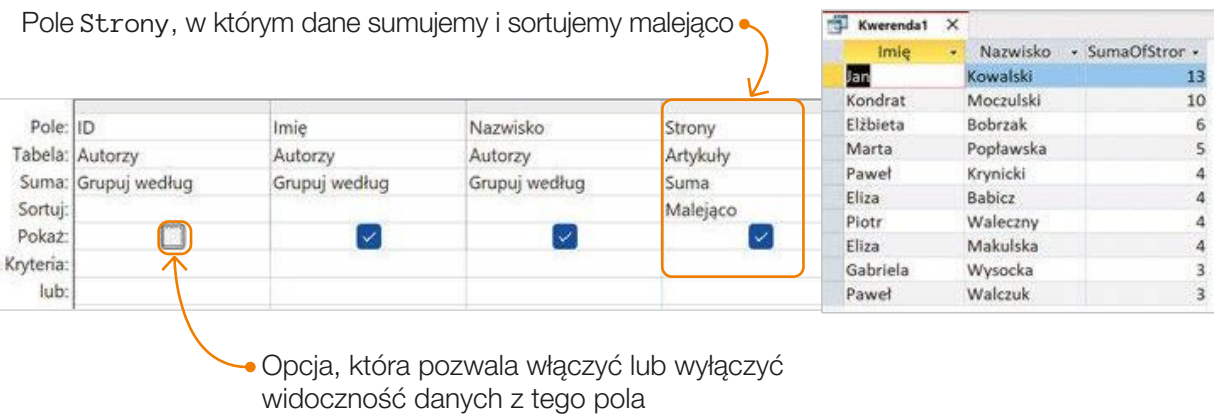
Do kwerendy można dodawać nie tylko tabele, lecz także inne, wcześniej utworzone kwerendy. Jest to możliwe, ponieważ wynikiem działania kwerendy jest w rzeczywistości zbiór wybranych danych ułożonych w formie tabeli.

Warto wiedzieć

SQL to język, który służy do tworzenia zapytań w relacyjnych bazach danych. W programie Access po utworzeniu kwerendy możemy zobaczyć jej zapis w języku SQL. W tym celu wystarczy przełączyć się z **Widoku projektu** lub **Widoku arkusza danych** na **Widok SQL**.

Dobra rada
Do **Widoku projektu** kwerendy przejdiesz, wybierając: **Narzędzia główne** → **Widoki** → **Widok** → **Widok projektu**.

Na tym etapie możemy już sprawdzić działanie kwerendy – wystarczy na karcie **Projekt kwerendy** wybrać przycisk **Uruchom**. Powinniśmy zobaczyć listę autorów, a przy każdym nazwisku – liczbę stron. Autorzy się jednak powtarzają, a my chcemy znać łączną liczbę stron dla każdego autora. Należy więc zmodyfikować kwerendę. W tym celu przechodzimy do **Widoku projektu** kwerendy i wybieramy **Projekt kwerendy** → **Pokazywanie/ukrywanie** → **Sumy**.
Na liście pól wybranych do kwerendy pojawiła się opcja **Grupuj według**. Aby zsumować liczbę stron, rozwijamy tę opcję pod polem Strony i wybieramy **Suma**. Jeżeli chcemy, żeby lista była posortowana w kolejności od autora z największą łączną liczbą stron, powinniśmy jeszcze wybrać opcję sortowania **Malejąco**. Możemy także odznaczyć wyświetlanie identyfikatora autora. Na rysunku 5.11 pokazano ustawienia kwerendy oraz jej wywołanie.



Rys. 5.11. Ustawienia kwerendy wyświetlającej liczbę wszystkich stron

Jeżeli kwerenda działa zgodnie z naszymi oczekiwaniami, możemy ją zapisać. W tym celu klikamy nazwę otwartej zakładki kwerendy prawym przyciskiem myszy i z menu kontekstowego wybieramy opcję **Zapisz**. W oknie **Zapisywanie jako** należy wprowadzić nazwę, która będzie odpowiadała działaniu kwerendy (rys. 5.12).

Warto wiedzieć
Po zapisaniu kwerendy jej nazwa pojawi się na liście wszystkich obiektów bazy.



Rys. 5.12. Zapisywanie kwerendy

Jeżeli wynik kwerendy jest inny od oczekiwanego, należy przełączyć się do **Widoku projektu** i wprowadzić poprawki.

Ćwiczenie 5

W pliku *baza_blog_5.accdb* z poprzedniego ćwiczenia utwórz kwerendę wyświetlającą imiona i nazwiska autorów oraz łączną liczbę stron artykułów, do których napisania się zobowiązali. Posortuj listę według nazwisk. Plik zapisz pod nazwą *baza_blog_6.accdb*.

Zadania

- 1 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z1.accdb*) przefiltruj i posortuj tabelę *Artykuły* tak, aby wyświetlić tylko nieoddane artykuły przeznaczone do działu *Nowe technologie* w kolejności, w której powinny zostać dostarczone.
- 2 Stwórz krótką notatkę tekstową na temat innych niż relacyjne rodzajów baz danych.
- 3 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z3.accdb*) utwórz relację typu jeden do jednego tak, aby jeden autor mógł mieć przypisany tylko jeden artykuł.
- 4 Zaproponuj listę atrybutów tabeli zawierającej informacje o:
 - a. uczniach danej szkoły,
 - b. artykułach w magazynie ogrodniczym,
 - c. kołach zainteresowań prowadzonych przez dom kultury.
- 5 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z5.accdb*) utwórz relację typu wiele do wielu.
- 6 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z6.accdb*) dla pól *Imię* i *Nazwisko* ustal maskę wprowadzania, która będzie wymuszała podanie imienia i nazwiska od wielkiej litery.
- 7 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z7.accdb*) stwórz kwerendę wyświetlającą informacje na temat liczby artykułów, do których napisania zobowiązali się poszczególni autorzy.
- 8 Zaprojektuj bazę danych, która pozwoliłaby samorządowi szkolnemu gromadzić informacje na temat uczniów gotowych pomagać kolegom w nauce przed maturą. Baza powinna zawierać dane kontaktowe wolontariuszy oraz umożliwiać odnajdywanie ich ze względu na przedmioty, w których mogą pomóc. Uwzględnij to, że jeden wolontariusz może pomagać w wielu przedmiotach oraz że dla jednego przedmiotu pomoc może oferować wielu wolontariuszy.
- 9 Zaproponuj, jak zmodyfikować opisaną w podręczniku bazę danych autorów bloga tak, aby można w niej było gromadzić dodatkowo informacje na temat nie tylko autorów, lecz także innych osób współpracujących przy tworzeniu bloga (np. grafików, animatorów, redaktorów).
- 10 W bazie danych otrzymanej od nauczyciela (np. w pliku *baza_blog_z10.accdb*) dodaj nową tabelę służącą do przechowywania informacji na temat działów, w których są publikowane artykuły. Wprowadź w niej nazwy działów oraz ich opisy na podstawie danych gromadzonych dotychczas w kolumnie *Dział* tabeli *Artykuły*. Następnie dodaj nowe pole *Id_Działu* w tabeli *Artykuły*. Za pomocą kreatora odnośników stwórz powiązanie między tabelami *Artykuły* oraz *Działy*. Przyporządkuj artykuły do działów, których nazwy zostały umieszczone w tabeli *Działy*. Następnie usuń kolumnę *Dział* z tabeli *Artykuły*.

Relacyjne bazy danych

Wyobraź sobie, że...

...jesteś dietetyczką w ośrodku sportowym. Do układania planów żywieniowych wykorzystujesz zaprojektowaną przez siebie **bazę danych**. Dzięki temu możesz gromadzić, a następnie wyszukiwać przepisy na posiłki skomponowane z określonych produktów i składników odżywczych. Dane przechowujesz w **tabelach**, a do filtrowania tych danych wykorzystujesz **kwerendy**. Dzięki temu przygotowujesz przepisy uwzględniające różne kryteria, takie jak płeć, dyscyplina, BMI oraz alergię.



...rozpoczynasz pracę w zespole opracowującym nowe elementy sterownicze do rakiet kosmicznych. Wiesz, że musicie przygotować dużą liczbę prototypów i przeprowadzić wiele testów. Projektujesz więc bazę danych do przechowywania wszystkich niezbędnych informacji. W **tabelach** dbasz o unikalność **kluczy głównych**, aby każdy test można było jednoznacznie określić. Dobierasz również odpowiednio **klucze obce**, aby **relacje** w bazie pozwalały na szybkie łączenie ze sobą informacji.

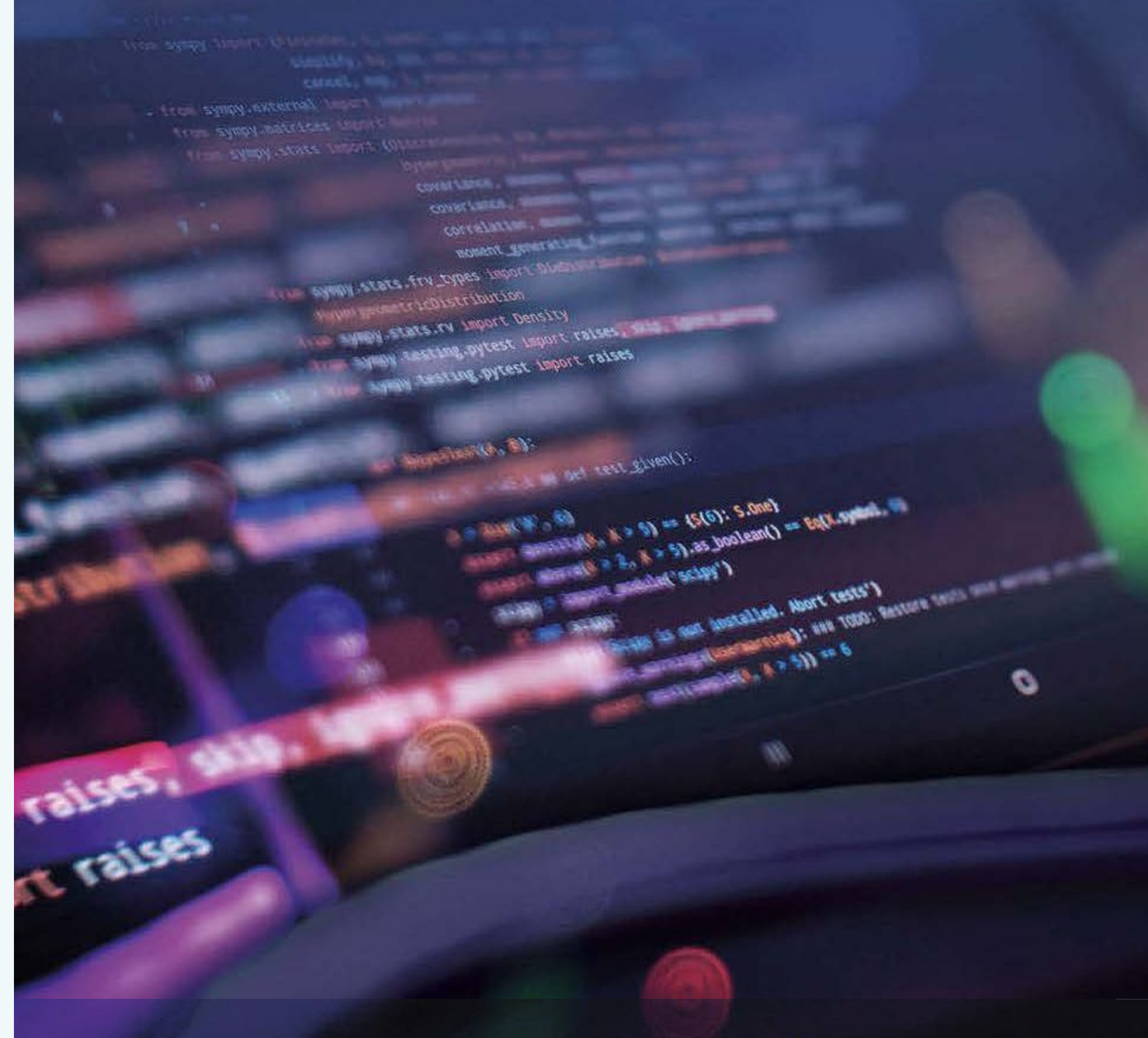
...zakładasz własną firmę kurierską. Chcesz konkurować z największymi graczami na rynku. Wiesz jednak, że musisz zacząć od podstaw. Projektujesz więc bazę danych z **tabelami**, które będą przechowywały informacje o klientach i przewoźnikach. Dobierasz **atrybuty** tabel tak, aby każdy **rekord** zawierał wszystkie niezbędne informacje. Łączysz też tabele **relacjami**, aby niepotrzebnie tych informacji nie dublować.



...pracujesz w dziale kontrolingu dużego przedsiębiorstwa. Twoim zadaniem jest gromadzenie i przetwarzanie danych kluczowych dla funkcjonowania firmy, takich jak wydajność produkcyjna, przewidywane koszty decyzji strategicznych czy polityki kadrowej. Dane pobierasz z **tabel** bazy danych przedsiębiorstwa. Dzięki dobrze ustawionym **kwerendom** twoja praca przebiega szybko i sprawnie.



3. Algorytmika i programowanie w języku Python





Algorytmika i programowanie w języku Python

Poniżej znajdziesz podstawowe informacje związane z pisaniem programów w języku Python.

Instrukcja przypisania (=) pozwala na zapisanie w pamięci komputera różnego rodzaju informacji (danych) pod wybraną nazwą (**zmienną**). Zapisana wartość może się zmieniać w trakcie działania programu.

`ile = 7` zapamięta liczbę 7 w zmiennej `ile`.

`imie = "Piotr"` zapamięta ciąg znaków Piotr w zmiennej `imie`.

Na liczbach i zmiennych, którym przypisano wartości liczbowe, można wykonywać **operacje arytmetyczne**.

`suma = ile + 3` zwiększy wartość zapisaną w zmiennej `ile` o 3 i zapisze ją w zmiennej `suma`.

`iloczyn = 2 * ile` podwoi wartość zmiennej `ile` i zapisze ją w zmiennej `iloczyn`.

Funkcja `input()` pozwala na wprowadzanie do programu danych z klawiatury.

`kto = input()` przypisze zmiennej `kto` tekst wpisany na klawiaturze.

Funkcja `print()` pozwala na wyprowadzanie informacji na ekran.

`print("Anna")` wypisze na ekranie ciąg znaków Anna.

`print("Anna i", imie)` spowoduje wyświetlenie na ekranie ciągu znaków Anna i Piotr, jeśli w zmiennej `imie` zostanie zapamiętany ciąg znaków Piotr.

`print(2 * ile)` służy do wypisania na ekranie podwojonej wartości zmiennej `ile`.

Instrukcja warunkowa pozwala wybrać jedną z dróg przebiegu programu w zależności od spełnienia lub niespełnienia warunku.

`if ile > 0:`
 `print("Więcej niż 0")`
wyświetli na ekranie Więcej niż 0 pod warunkiem, że wartość zmiennej `ile` jest dodatnia.

6

Od problemu do programu

W ostatnim stuleciu rozwój komputerów znacznie ułatwił rozwiązywanie różnego rodzaju problemów. Zadania, których wykonanie zajmuje człowiekowi kilka dni, tygodni, a czasem nawet lat, komputer wykona w zaledwie kilka sekund. Istotę danego problemu możemy przedstawić w odpowiednim, zrozumiałym dla komputera języku, czyli w języku programowania. W tym temacie poznasz podstawy programowania komputerów w języku Python.

Ważne w tym temacie

- Analiza zadania (problemu)
- Tworzenie algorytmu rozwiązania zadania
- Zapis algorytmu w języku programowania
- Testowanie działania programu

Ćwiczenie 1

Z czym kojarzy ci się słowo *algorytm*? Przywołaj sytuacje, w których ktoś użył tego słowa.

6.1. Algorytm – precyzyjna instrukcja

Wiemy, czym są instrukcja, przepis czy procedura. Tymczasem coraz powszechniej używa się technicznego synonimu tych słów, czyli algorytmu. **Algorytm** to precyzyjny opis kolejnych działań, które prowadzą do wykonania zadania (rozwiązania problemu). Pierwotnie matematycy nazywali tak pisemny sposób wykonywania rachunków na liczbach (np. $634 + 87 + 347$).

Z zadaniami algorytmicznymi spotykamy się właściwie na każdym kroku. Wyobraźmy sobie, że na zajęciach wychowania fizycznego każdy uczeń proponuje po jednym ćwiczeniu na rozgrzewkę. Najprościej będzie wtedy ćwiczenie zademonstrować, zamiast o nim mówić (rys. 6.1, s. 92). Gdyby jednak trzeba było zapisać instrukcję kolejnych działań składających się na ćwiczenie fizyczne, można to zrobić, tworząc listę kroków.

Lista kroków to precyzyjny opis postępowania składający się z zapisanych słownie kolejnych pojedynczych instrukcji, nazywanych krokami, które mogą się odwoływać do siebie nawzajem. Lista kroków może się rozgałęziać (mogą być wykonywane różne kroki, w zależności od spełnienia lub niespełnienia warunku). Część kroków może też być powtarzana.

Algorytm

Warto wiedzieć

Słowo *procedura* zostało zapożyczone z języka prawniczego, w którym oznacza kolejne kroki procesu, prowadzenia sprawy lub reguły określające te kroki.

Lista kroków



Rys. 6.1. Demonstracja ćwiczenia na rozgrzewkę

Lista kroków algorytmu wykonania ćwiczenia polegającego na wypadach do przodu może wyglądać następująco:

- 1 Stań prosto na płaskiej powierzchni.
- 2 Powtórz 8 razy kroki od 3 do 6 dla obu nóg.
- 3 Pochyl się mocno do przodu, opierając ciężar ciała na jednej nodze.
- 4 Drugą nogę pozostaw prostą.
- 5 Odczekaj 2 sekundy.
- 6 Wróć do pozycji wyjściowej.

Warto zauważyć, jak precyzyjny jest opis kolejnych czynności. W przypadku tych, które należy powtórzyć, podana jest dokładna liczba powtórzeń. Inaczej nie uzyska się oczekiwanego efektu (rozgrzania ciała). Opis ćwiczenia na rozgrzewkę ma **cechy poprawnego algorytmu**, ponieważ:

- ▶ jest skończony – można go zrealizować w skończonej liczbie kroków,
- ▶ jest precyzyjny (dobrze określony i niezawodny) – jednoznacznie definiuje, jakie czynności należy wykonać, aby osiągnąć cel,
- ▶ jest uniwersalny – może być użyteczny w różnych okolicznościach.

Oprócz listy kroków algorytm można **zapisać w inny sposób**, np. w postaci schematu blokowego lub opisu słownego.

Ćwiczenie 2

Przygotuj dwa opisy (w postaci listy kroków) wykonywania zadań z życia codziennego, które mają cechy algorytmu.

Zapamiętaj

Algorytm to opis postępowania prowadzącego do rozwiązania problemu. Znajomość cech poprawnego algorytmu pomaga w tworzeniu dobrych algorytmów.

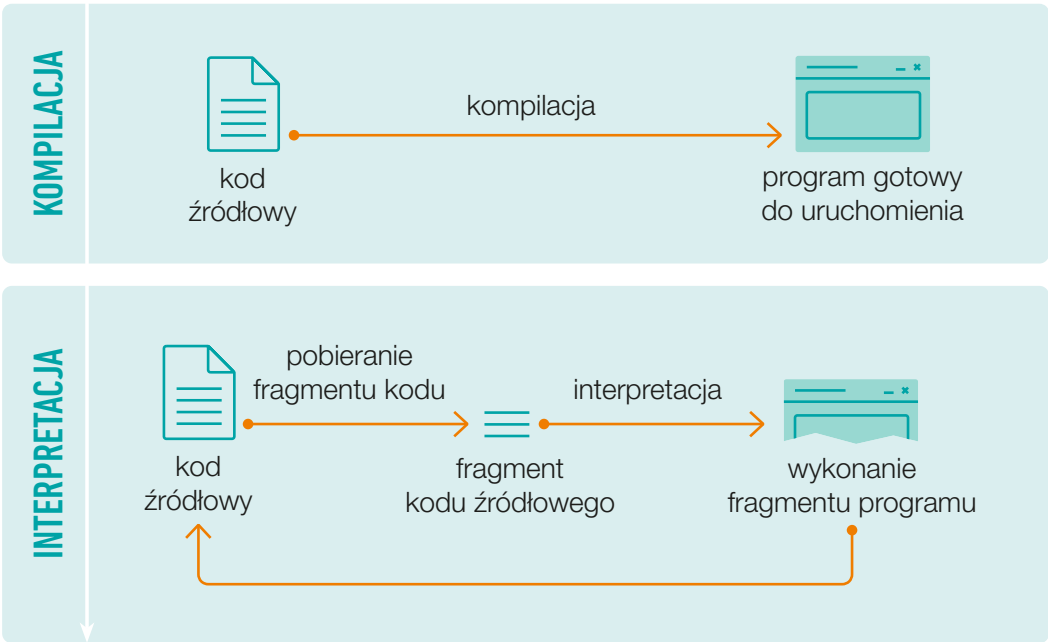
6.2. Od algorytmu do programu komputerowego

Do rozwiązywania problemów z pomocą komputera służą programy komputerowe. W rzeczywistości są one zbiorami algorytmów zapisanych w języku zrozumiałym dla komputera.

Do tworzenia programów komputerowych (oprogramowania) używa się ściśle zdefiniowanych słów i składni **języka programowania**. Zapis algorytmu w takim języku nazywamy **kodem źródłowym** (ang. *source code*) programu komputerowego lub po prostu tekstem programu. Do zapisu kodu źródłowego używa się zwykle specjalnego edytora, często wchodzącego w skład środowiska programistycznego, czyli **IDE**.

Aby można było uruchomić na komputerze program zapisany w kodzie źródłowym, trzeba go najpierw przetłumaczyć **translatorem** na **kod wynikowy (maszynowy)**, czyli taki, który będzie zrozumiały dla procesora.

Translatory, które od razu tworzą kod wynikowy całego programu, nazywamy **kompilatorami**. Drugi typ programów tłumaczących to **interpretery**, które na bieżąco tłumaczą kod źródłowy w trakcie wykonywania programu (rys. 6.2).



Rys. 6.2. Różnica między kompilacją a interpretacją kodu źródłowego

Trzeba tutaj zaznaczyć, że w przypadku programów zapisanych w językach interpretowanych na komputerze musi być zainstalowany interpreter. Programy skompilowane mogą być uruchamiane bez dodatkowego oprogramowania. Przykładem języka kompilowanego jest **język C++**, a interpretowanego – omówiony w tym rozdziale **język Python**. Przykład prostego kodu źródłowego zapisanego w języku Python przedstawiamy na infografice na następnej stronie. Znajdują się na niej również najważniejsze zasady, o których należy pamiętać, tworząc programy w tym języku.

Warto wiedzieć
Interpreter działa według zasady: przeczytaj fragment tekstu programu, przetłumacz go na kod maszynowy, pozwól komputerowi na wykonanie przetłumaczonego fragmentu, a następnie przeczytaj kolejny fragment tekstu programu, przetłumacz go itd.

ZAPISYWANIE PROGRAMÓW W JĘZYKU PYTHON

Kody źródłowe języka Python zapisujemy w edytorze kodu połączonym z interpreterem. Powszechnie korzysta się ze zintegrowanych środowisk programistycznych (IDE). Jednym z nich jest program Mu, który automatycznie koloruje elementy składniowe kodu (np. słowa kluczowe języka, teksty i liczby).



Przykładowy kod źródłowy programu w języku Python

Funkcja `print` pozwala wyświetlić tekst na ekranie.

Program wyświetla komunikat zachęcający do podania nazwy użytkownika, a następnie w zmiennej `imie` zapamiętuje wprowadzony tekst, zatwierdzony klawiszem **Enter**.

```
1. print("Podaj swoje imię:")
2. imie = input()
3. print(imie, "uczy się programowania.")
```

Jeśli funkcja `print` ma wyświetlić więcej niż jeden argument, trzeba je oddzielić przecinkami.



6.3. Tworzymy własne programy użytkowe

Wykorzystamy język programowania Python. Kody źródłowe programów będziemy pisać w środowisku programistycznym Mu. Każdy kod będzie się składał z **instrukcji** języka, czyli poleceń zrozumiałych dla interpretera, który będzie umiał je wykonać. W oknie Mu uruchomimy też napisane programy.

- Tworzenie programu** krok po kroku:
- 1. Przygotuj **specyfikację zadania**, czyli określ cel, który chcesz osiągnąć.
 - 2. Zaprojektuj algorytm rozwiązania zadania.
 - 3. Zapisz algorytm w postaci kodu źródłowego.
 - 4. Uruchom i przetestuj program.
 - 5. Popraw ewentualne błędy.

Istotne jest, aby już na etapie tworzenia listy kroków poszczególne czynności (działania) zapisywać zwięźle i jednoznacznie, co uprości proces tworzenia kodu źródłowego.

Program *Powitanie* – interakcja użytkownika z komputerem

Zacniemy od napisania programu, który najpierw zapyta o imię użytkownika, a następnie wyświetli tekst powitalny. Powitanie powinno zawierać imię użytkownika. Najpierw określimy specyfikację zadania. Specyfikacja składa się z danych przekazywanych do programu oraz spodziewanego wyniku działania programu.

Specyfikacja

Dane: imię użytkownika wpisane z klawiatury.

Wynik: wyświetlony tekst powitania zawierający imię użytkownika.

Dobra rada

Aktualną wersję programu Mu możesz pobrać ze strony <https://codewith.mu/>. W wersji dla systemu Windows do edytora dołączony jest interpreter języka Python.

Zasady tworzenia kodu w języku Python

Kolejne instrukcje należy umieszczać w kolejnych wierszach.

Wielkość liter w kodzie ma znaczenie – zmienne `a` i `A` mogą przechowywać różne wartości.

O typie zmiennej decyduje nadana jej wartość. Wartość wprowadzona z klawiatury domyślnie traktowana jest jako tekst.

Między wartościami a operatorami można wstawiać spacje (np. `a + 2`).

Warto stosować w kodzie puste wiersze, ponieważ zwiększają czytelność programu.

Tekst po znaku `#` do końca wiersza jest komentarzem – interpreter go ignoruje.

A to ciekawe

Zen of Python – zasady programowania

Kod źródłowy programu zapisany w języku Python w założeniach jego twórcy, Guida van Rossuma, miał być zwięzły i prosty. Spisano zbiór zasad, którymi powinni się kierować programiści przy tworzeniu kodu źródłowego w języku Python. Wśród tych zasad znajdziemy m.in. takie zasady, jak *Piękne jest lepsze niż brzydkie* (ang. *Beautiful is better than ugly*) czy *Proste jest lepsze niż złożone* (ang. *Simple is better than complex*).



Listę kroków algorytmu realizacji powyższego zadania można zapisać następująco:

- 1 Poproś użytkownika o podanie imienia.
- 2 Zapamiętaj podane imię.
- 3 Wypisz na ekranie powitanie zawierające imię użytkownika.

Z kolei kod źródłowy programu realizującego algorytm zapisany w postaci powyższej listy kroków można zapisać tak:

Kod źródłowy programu **Powitanie**

```
1. print("Podaj swoje imię:")
2. imie = input()
3.
4. print(imie, "uczy się programowania.")
```

- Funkcja print** W 1. linijce znajduje się instrukcja wywołania **funkcji print**, która
- Instrukcja wyjścia** wyświetla tekst na ekranie. Mówimy, że jest to **instrukcja wyjścia**, ponieważ dane wychodzą z komputera i są przekazywane do użytkownika. Z kolei na to, co znajduje się w nawiasie, mówimy **argument** – jest to wartość przyjmowana przez funkcję.
- Zmienna** Druga linijka zaczyna się od zdefiniowania **zmiennej**, czyli miejsca w pamięci, które w trakcie działania programu będzie przechowywało zapisaną wartość. Wykorzystujemy **operator przypisania** =. Po prawej stronie tego operatora znajduje się **funkcja input**, która po wywołaniu poczeka na wpisanie czegoś przez użytkownika i w naszym przypadku przypisze wpisaną wartość zmiennej imie. Ta funkcja jest nazywana
- Operator przypisania** **instrukcją wejścia**, ponieważ przyjmuje dane od użytkownika.
- Funkcja input**

Warto wiedzieć
Podczas wypisywania znaków tekstu na ekranie argumenty podane między nawiasami funkcji **print** są oddzielane spacją.

Dobra rada
Stosuj puste linie do oddzielania logicznych części kodu. Ułatwi to innym programistom analizę tekstu programu.

Dobra rada
Możesz też uruchomić program, naciskając klawisz F5.

Pusta 3. linijka została umieszczona w kodzie ze względów estetycznych. Warto dzielić kod na mniejsze logiczne fragmenty dla lepszej czytelności.
W 4. linijce kodu za pomocą funkcji **print** program wyświetli na ekranie zdanie składające się z elementów zawartych w nawiasie, czyli wartości zmiennej **imie** oraz zwrotu **uczy się programowania**.
Gotowy program zapisujemy w pliku z rozszerzeniem **py**, a następnie uruchamiamy, klikając **Wykonuj** (rys. 6.3).



Rys. 6.3. Przycisk służący do uruchomienia programu

Efekt wywołania programu widzimy na rysunku 6.4.



Rys. 6.4. Wywołanie programu *Powitanie*

Działanie programu przerywamy, klikając **Zatrzymaj**.

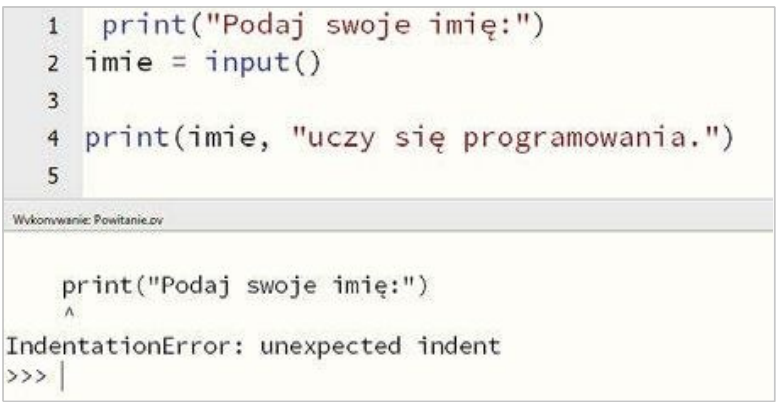
Ćwiczenie 3

Przepisz tekst programu i zapisz go pod nazwą *Powitanie.py*. Uruchom program. Jeśli pojawi się informacja o błędzie, popraw tekst programu.

Jeżeli w kodzie źródłowym wystąpi **błąd składniowy**, czyli taki, który wynika z zapisania kodu niezgodnie ze **składnią** (z regułami danego języka programowania), to próba uruchomienia takiego programu zakończy się wyświetleniem komunikatu o błędzie.

Na przykład jeśli na początku wiersza kodu w programie *Powitanie.py* pojawi się dodatkowy odstęp (choćby pojedyncza spacja), to po próbie uruchomienia programu pojawi się komunikat o błędzie składniowym **unexpected indent**, czyli nieoczekiwane wcięcie, a znak **^** wskaże miejsce, w którym interpreter wykrył błąd. W języku Python wcięcia mają kluczowe znaczenie i trzeba je konsekwentnie stosować w całym kodzie. Jeśli przed instrukcjami znajdą się zbędne spacje, program będzie to interpretował jako błąd (rys. 6.5).

Warto wiedzieć
Niektórzy programiści zamiast cudzysłowu (w postaci ") wolą używać apostrofu ('), ponieważ jego wpisanie nie wymaga przytrzymywania wciśniętego klawisza **Shift**.



Rys. 6.5. Błąd składniowy w kodzie źródłowym programu

Program *Szarlotka* – korzystamy z instrukcji warunkowej **if**

Program, który napiszemy, policzy jabłka zebrane przez Anię i Wojtkę w ich przydomowym sadzie. Mamie potrzeba 15 jabłek do upieczenia szarlotki. Po wprowadzeniu liczby jabłek zebranych przez Anię i Wojtkę program zsumuje je i wyświetli informację, czy suma jabłek wystarczy do upieczenia szarlotki.

Specyfikacja

Dane: liczba jabłek zebranych przez Anię i Wojtka wprowadzona z klawiatury.

Wynik: informacja, czy suma zebranych jabłek wystarczy na szarlotkę, lub komunikat o tym, że liczba jabłek jest niewystarczająca.

Oto propozycja listy kroków algorytmu:

- 1 Zapytaj Anię o to, ile ma jabłek. Zapamiętaj tę liczbę.
- 2 Zapytaj Wojtka o to, ile ma jabłek. Zapamiętaj tę liczbę.
- 3 Oblicz sumę liczb z kroków 1 i 2. Zapamiętaj tę sumę.
- 4 Jeśli suma jest liczbą mniejszą niż 15, to wypisz informację, że jabłek jest za mało na szarlotkę. W przeciwnym razie wypisz informację, że liczba jabłek jest wystarczająca.

Warto wiedzieć

Błąd algorytmu rozwiązania zadania nazywa się błędem logicznym.

Wykonajmy symulację działania tego algorytmu dla przykładowych liczb, realizując kolejne czynności z listy kroków (tab. 6.1). Na tym etapie można już zaproponować brzmienie tekstu pytań, których użyjemy w programie. Gdyby się okazało, że w projekcie algorytmu jest błąd, to będziemy mogli go usunąć, zanim przystąpimy do zapisywania tekstu programu.

Wyjście	Wejście	Działania
Aniu! Ile masz jabłek?	4	
Wojtku! Ile masz jabłek?	5	
		4 + 5 = 9
		Czy 9 < 15
Jabłek jest za mało.		

Tabela 6.1. Symulacja algorytmu Szarlotka

Oto tekst programu realizującego powyższy algorytm.

Kod źródłowy programu Szarlotka

```
1. print("Aniu, ile masz jabłek? ")
2. ile_Ania = int(input())
3. print("Wojtku, ile masz jabłek? ")
4. ile_Wojtek = int(input())
5.
6. ile_razem = ile_Ania + ile_Wojtek
7. if ile_razem < 15:
8.     print("Jabłek jest za mało na szarlotkę.")
9. else:
10.    print("Jabłek wystarczy na szarlotkę.")
```

W 2. linijce programu, po prawej stronie znaku = znajduje się instrukcja wywołania funkcji `input`. Jest ona umieszczona pomiędzy nawiasami innej instrukcji – wywołania funkcji `int`. Dzięki umieszczeniu instrukcji wejścia jako argumentu funkcji `int` interpreter języka Python potraktuje wartość wpisaną przez użytkownika jako liczbę, a nie tekst. Analogiczne rozwiązanie mamy w 4. linijce.

Dzięki temu zabiegowi będzie możliwe obliczenie sumy jabłek – wyrażenie `ile_razem = ile_Ania + ile_Wojtek` w 6. linijce interpreter zrozumie następująco: oblicz sumę liczb zapamiętanych w zmiennych o podanych nazwach i tę wartość przypisz zmiennej `ile`.

W liniach od 8. do 11. zastosowano instrukcję warunkową `if`, od której zależy dalszy przebieg programu. Schemat ogólnej instrukcji warunkowej wygląda następująco:

```
if warunek logiczny:
    instrukcja na tak
else:
    instrukcja na nie
```

Jeśli warunek logiczny ma wartość logiczną **prawda**, to jest wykonywana instrukcja na tak, w przeciwnym razie – instrukcja na nie.

Zapis `if ile_razem < 15` z 7. linijki mówi o tym, że jeśli wartość zmiennej `ile_razem` jest liczbą mniejszą niż 15, to zostaną wykonane instrukcje znajdujące się w kolejnych liniach i wyróżnione wcięciem. Kolejne linijki z wcięciem to **blok instrukcji** (może być ich wiele). W przeciwnym razie zostaną wykonane instrukcje z bloku znajdującego się po słowie `else`.

Słowa `if` i `else` to przykłady **słów kluczowych** języka Python, czyli słów zdefiniowanych przez twórcę języka programowania. Programista nie może używać nazw należących do zbioru słów kluczowych w nazwach, które sam tworzy w tekście programu (np. w nazwach zmiennych).

Funkcja int

Warto wiedzieć

Nazwa funkcji `int` to skrót angielskiego słowa *integer* oznaczającego liczbę całkowitą.

Warto wiedzieć

Domyślnie wcięcie bloku instrukcji składa się z czterech spacji.

Blok instrukcji

Słowa kluczowe

Ćwiczenie 4

Przepisz tekst programu i zapisz go pod nazwą *Szarlotka.py*. Uruchom program i przetestuj go dla różnych danych.

Program Zbiór jabłek – wprowadzamy instrukcję powtarzania while

Program policzy sumę jabłek zebranych przez Anię i Wojtka w czasie ich pracy u sadownika. Ania z Wojtkiem powinni zebrać wspólnie 60 kg jabłek, co zdaniem pracodawcy odpowiada 300 jabłkom stosownej wielkości. Mogą oni zbierać jabłka partiami, po kilkanaście lub kilkadziesiąt sztuk. Program będzie tak długo pytał o kolejne wartości, aż suma będzie wynosiła 300 lub więcej.

Specyfikacja

Dane: kolejno wprowadzane liczby jabłek zebranych za każdym razem przez Anię i Wojtkę.

Wynik: sprawdzenie, czy suma zebranych jabłek wynosi przynajmniej 300.

Lista kroków algorytmu może wyglądać tak:

- 1 Ustaw wartość sumy na 0.
- 2 Powtarzaj kroki 3 i 4, dopóki suma jest liczbą mniejszą niż 300.
- 3 Zapytaj o liczbę przyniesionych jabłek. Zapamiętaj tę liczbę.
- 4 Dodaj zapamiętaną liczbę do sumy.
- 5 Wypisz na ekranie informację, że liczba jabłek jest wystarczająca.

Kroki 3 i 4 będą powtarzane tak długo, aż warunek określony w kroku 2 zostanie spełniony. Symulację działania algorytmu przedstawia tabela 6.2. Odpowiedź „NIE” oznacza, że nastąpiło przejście do 5 kroku algorytmu.

Liczba jabłek przy bieżącym podejściu	Suma jabłek	Czy wartość sumy jest mniejsza niż 300?
–	0	TAK
55	55	TAK
51	106	TAK
63	159	TAK
55	214	TAK
52	266	TAK
50	316	NIE

Tabela 6.2. Symulacja algorytmu Zbiór jabłek

Poniżej znajduje się ten algorytm w postaci kodu źródłowego języka Python.

Kod źródłowy programu Zbiór jabłek

```
1. suma = 0
2. while suma < 300:
3.     ile = int(input("Ile jabłek przynieśliście? "))
4.     suma += ile
5. print("Brawo! Uzbieraliście wymaganą liczbę jabłek. ")
```

Rysunek 6.6 pokazuje przykładowy przebieg działania programu.

```
Wykonywanie: zbior_jabłek.py
Ile jabłek przynieśliście? 65
Ile jabłek przynieśliście? 71
Ile jabłek przynieśliście? 73
Ile jabłek przynieśliście? 75
Ile jabłek przynieśliście? 70
Brawo! Uzbieraliście wymaganą liczbę jabłek.
>>>
```

Rys. 6.6. Przykładowy przebieg działania programu Zbiór jabłek

W 1. linijce zmiennej suma przypisujemy początkową wartość 0. Linijka 2. stanowi nagłówek instrukcji powtarzania (iteracyjnej), nazywanej też pętlą. W języku Python występują dwa rodzaje instrukcji powtarzania. W powyższym kodzie użyliśmy pętli while, która ma następującą budowę:

```
while warunek:
    instrukcje
```

Wszystkie instrukcje znajdujące się za dwukropkiem i wyróżnione wcięciem będą powtarzane tak długo, dopóki warunek jest prawdziwy. Wartość zmiennej ile (w 3. linijce) jest przy każdym obrocie pętli podawana przez użytkownika. Skorzystaliśmy przy tym z możliwości, które daje instrukcja input. Wypisaliśmy tekst na ekranie w miejscu bezpośrednio poprzedzającym położenie kursora wskazującego, gdzie ma być wprowadzony tekst z klawiatury. W 4. linijce znajduje się zapis: suma += ile. Oznacza on, że poprzednią wartość zmiennej suma zwiększamy o aktualną wartość zmiennej ile i tę sumę przypisujemy jako nową wartość zmiennej suma.

Ćwiczenie 5

Przepisz kod programu i zapisz go pod nazwą Zbiór_jabłek.py. Uruchom program i przetestuj go dla różnych danych. Rozpocznij od przykładu danych używanych w tabeli powyżej, podczas symulacji.

Zapamiętaj

Instrukcji warunkowej używa się do wyboru ścieżki przebiegu programu. Pętla służy do wielokrotnego powtarzania bloku instrukcji.

 Zadania

- * **1** Przedstaw w postaci listy kroków czynności składające się na:
 - a. wyjście z domu do szkoły (od pobudki do dotarcia do szkoły),
 - b. przygotowanie do wyjazdu na wakacje (pakowanie),
 - c. przygotowanie dowolnej potrawy,
 - d. wstawienie prania.
- * **2** Napisz program, który zapyta użytkownika o imię i ulubiony kolor, a następnie wyświetli zdanie złożone z imienia, słowa *lubi* i nazwy koloru.
- * **3** Napisz program, który wczyta z klawiatury dwie liczby i sprawdzi, czy są one różne. W zależności od tego, czy są różne, czy nie, program powinien wyświetlić odpowiedni komunikat.
- * **4** Po opróżnieniu skarbonki Piotr poukładał na stosach monety o nominałach 5 zł, 2 zł i 1 zł oraz banknoty o nominałach 10 zł. Napisz program, który pozwoli obliczyć wartość pieniędzy w skarbonce.
- ** **5** Napisz program, który wyświetli kolejne liczby naturalne od 1 do 10.
- ** **6** Napisz program, który wyświetli wszystkie liczby nieparzyste od 1 do 99, a następnie wszystkie liczby parzyste od 2 do 100.
- ** **7** Ania znalazła taką instrukcję przeliczania lat ludzkich na lata psie: *Wiek 2-letniego psa odpowiada wiekowi 7-letniego dziecka. Każdy kolejny rok ludzki odpowiada 7 latom psim.* Napisz program, który pozwoli Ani przeliczyć wiek dowolnego psa na lata psie.
- ** **8** Napisz program obliczający, ile minut minęło od północy. Użytkownik podaje informację o bieżącym czasie (godzina i minuta) w postaci liczb zapisanych w kolejnych wierszach.
- *** **9** Napisz program, który wczyta z klawiatury liczbę całkowitą dodatnią i sprawdzi, z ilu cyfr składa się ta liczba.
- *** **10** Napisz program, który przeliczy informację o bieżącym czasie z formatu 24-godzinnego na 12-godzinny.
- *** **11** Napisz program, który będzie określać porę dnia (noc, dzień, wieczór) na podstawie informacji o godzinie podanej z dokładnością do minuty. Danymi do programu powinny być dwie liczby określające aktualny czas zegarowy.

7

Systemy liczbowe

Kiedy korzystasz z komputera lub smartfona, wykonujesz wiele różnych czynności, m.in. słuchasz muzyki, przetwarzasz zdjęcia, redagujesz teksty, wysyłasz e-maile i oglądasz transmisje wideo. To wszystko jest możliwe dzięki temu, że zarówno oprogramowanie, jak i różnego typu dane są zapisywane w pamięci komputera w jednolity sposób – cyfrowo, czyli jako liczby w systemie binarnym (dwójkowym).

Ważne w tym temacie

- Zapisywanie liczb w systemie binarnym
- Inne systemy liczbowe
- Cyfrowy zapis informacji
- Algorytmy zamiany reprezentacji liczb
- Korzystanie ze struktury listy w języku Python
- Zaprogramowanie algorytmu do zamiany liczby dziesiętnej na liczbę dwójkową

7.1. Liczby dziesiętne i dwójkowe

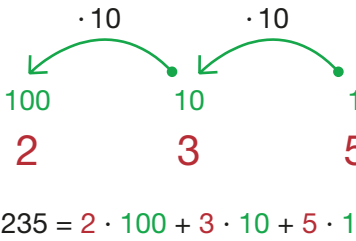
Gdy na co dzień wykorzystujemy dane liczbowe, zazwyczaj posługujemy się **systemem dziesiętnym (decymalnym)**. Dlaczego właśnie ten system się upowszechnił? Zapewne dlatego, że pierwszymi liczydłami były palce u dłoni człowieka. Liczby zapisujemy więc za pomocą dziesięciu cyfr: 0, 1, 2, 3, 4, 5, 6, 7, 8 i 9.

Naturalne jest dla nas, że zapis 2350 oznacza liczbę dziesięć razy większą od 235. Wynika to z faktu, że system dziesiętny jest **systemem pozycyjnym**. To oznacza, że z pozycją cyfry (liczoną od prawej strony) wiąże się jej waga – w tym wypadku: 1, 10, 100, 1000 itd. Są to kolejne potęgi liczby 10, która w tym przypadku jest **podstawą systemu liczbowego**. Każdą liczbę przedstawia się w tym systemie jako sumę wielokrotności potęg liczby 10. Na przykład liczba 235 jest sumą dwóch potęg 10^2 , trzech potęg 10^1 i pięciu potęg 10^0 .

Na rysunku 7.1 pokazano zapis liczby 235 w systemie dziesiętnym.

Dziesiętny system pozycyjny
Podstawa systemu: 10
Cyfry: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Wagi cyfr na kolejnych pozycjach (liczonych od prawej strony) są kolejnymi potęgami liczby dziesięć: 10^0 , 10^1 , 10^2 , 10^3 , ...



$235 = 2 \cdot 100 + 3 \cdot 10 + 5 \cdot 1$

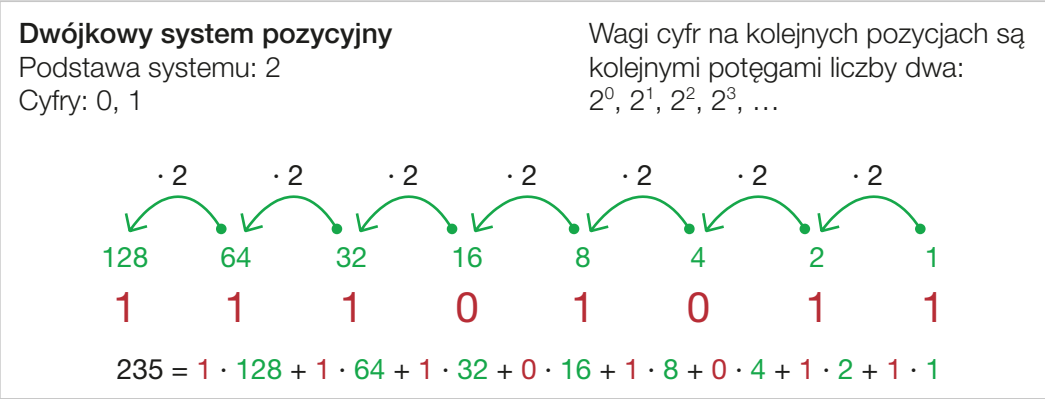
 Warto wiedzieć

Słowo *waga* w matematyce oznacza czynnik liczbowy, którym poprzedza się jakąś liczbę, aby zaznaczyć stopień jej ważności (istotności).

Rys. 7.1. Zapis dziesiętny liczby 235

System dwójkowy (binarny) W technice komputerowej szczególne znaczenie ma **system dwójkowy (binarny)**, w którym występują tylko dwie cyfry: 0 i 1. Wagi cyfr na kolejnych pozycjach w zapisie liczby w systemie dwójkowym są potęgami liczby 2. Na rysunku 7.2 przedstawiono binarną postać liczby 235.

Warto wiedzieć
O liczbie przedstawionej w systemie dwójkowym mówi się w skrócie: *liczba dwójkowa*.



Rys. 7.2. Zapis dwójkowy liczby 235

Zauważ, że liczba 235 jest sumą $2^7, 2^6, 2^5, 2^3, 2$ oraz 1 i jest – w zapisie dwójkowym – liczbą ośmiocyfrową 11101011.

Tabela 7.1 zawiera zapis binarny liczb od 0 do 15. Możemy na tej podstawie sformułować kilka

Podstawowe własności liczb binarnych **podstawowych własności liczb binarnych:**

1. Liczby nieparzyste w zapisie binarnym kończą się cyfrą 1.
2. Liczby binarne kończące się cyfrą 0 dzielą się przez 2, a więc są parzyste.
3. Aby uzyskać liczbę binarną dwa razy większą od wyjściowej, należy na końcu dopisać cyfrę 0. Oznacza to, że jeśli usuniemy ostatnią cyfrę parzystej liczby binarnej, otrzymamy połowę tej liczby.
4. Zapisy dwóch kolejnych liczb binarnych – mniejszej parzystej i większej nieparzystej – różnią się tylko ostatnią cyfrą. Liczba parzysta kończy się cyfrą 0, a nieparzysta – cyfrą 1, np. 12 to 1100, a 13 to 1101.

Takich własności można sformułować więcej i wszystkie można matematycznie udowodnić.

Liczba	Zapis binarny	Uzasadnienie
0	0	0
1	1	2^0
2	10	2^1
3	11	$2^1 + 2^0$
4	100	2^2
5	101	$2^2 + 2^0$
6	110	$2^2 + 2^1$
7	111	$2^2 + 2^1 + 2^0$
8	1000	2^3
9	1001	$2^3 + 2^0$
10	1010	$2^3 + 2^1$
11	1011	$2^3 + 2^1 + 2^0$
12	1100	$2^3 + 2^2$
13	1101	$2^3 + 2^2 + 2^0$
14	1110	$2^3 + 2^2 + 2^1$
15	1111	$2^3 + 2^2 + 2^1 + 2^0$

Tabela 7.1. Zapis binarny liczb od 0 do 15

Dobra rada
Liczby dwójkowe nie nazywaj tak samo jak liczby w systemie dziesiętnym. Na przykład liczbę zapisaną jako 1010 w systemie dziesiętnym nazywamy „tysiąc dziesięć”, ale w systemie binarnym należy ten zapis odczytać „jeden zero jeden zero”.

Zapamiętaj

Dwójkowy system liczbowy (podobnie jak system dziesiętny) opiera się na zasadzie zapisu wagowo-pozycyjnego. Cyfry w zapisie liczby znajdują się na ponumerowanych pozycjach. Każdej pozycji (licząc od prawej do lewej) odpowiada pewna waga: 1, 2, 4, 8 itd. Wagi są kolejnymi potęgami liczby 2, którą nazywa się podstawą tego systemu liczbowego.

Ćwiczenie 1

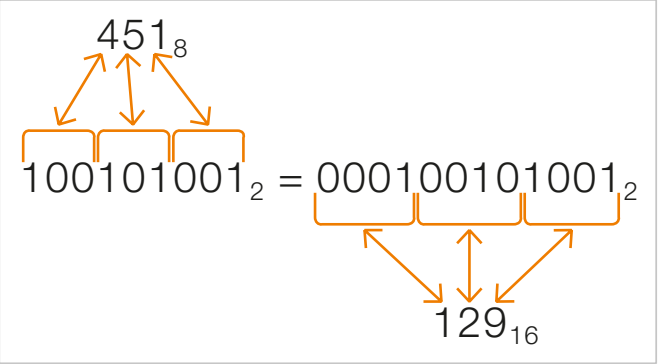
Jaką największą liczbę można zapisać w systemie dwójkowym za pomocą ośmiu cyfr?

Inne systemy liczbowe

W informatyce często wykorzystuje się również systemy ósemkowy i szesnastkowy. W **systemie ósemkowym** liczby tworzy się z ośmiu cyfr (od 0 do 7), natomiast w **systemie szesnastkowym** jest ich aż szesnaście: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Do zapisu sześciu ostatnich cyfr wykorzystuje się znaki będące kolejnymi literami alfabetu łacińskiego. Litera F w systemie szesnastkowym odpowiada liczbie 15 w systemie dziesiętnym. Aby oznaczyć, w jakim systemie została zapisana liczba, w indeksie dolnym zapisuje się podstawę systemu (rys. 7.3).

Podstawy systemów szesnastkowego i ósemkowego są potęgami liczby 2, dlatego stosunkowo łatwo można zamienić liczby zapisane w tych systemach na reprezentację w systemie binarnym i odwrotnie.

Każdą cyfrę ósemkową można zapisać za pomocą trzech cyfr binarnych, a każdą cyfrę szesnastkową – za pomocą czterech cyfr binarnych, przy czym mniejsze wartości uzupełniamy z lewej strony zerami (rys. 7.3).



Rys. 7.3. Zamiana zapisu liczby pomiędzy systemami dwójkowym, ósemkowym i szesnastkowym

7.2. Cyfrowy zapis informacji

Wszelkiego typu informacje (liczby, teksty, grafika, dźwięk itd.) w systemach komputerowych są przedstawiane (kodowane) jako liczby. Najmniejszą, niepodzielną porcją informacji jest **bit** (od ang. *binary digit*), czyli informacja, którą można przedstawić w postaci jednej cyfry dwójkowej: 0 albo 1.

- System ósemkowy
- System szesnastkowy

Dobra rada

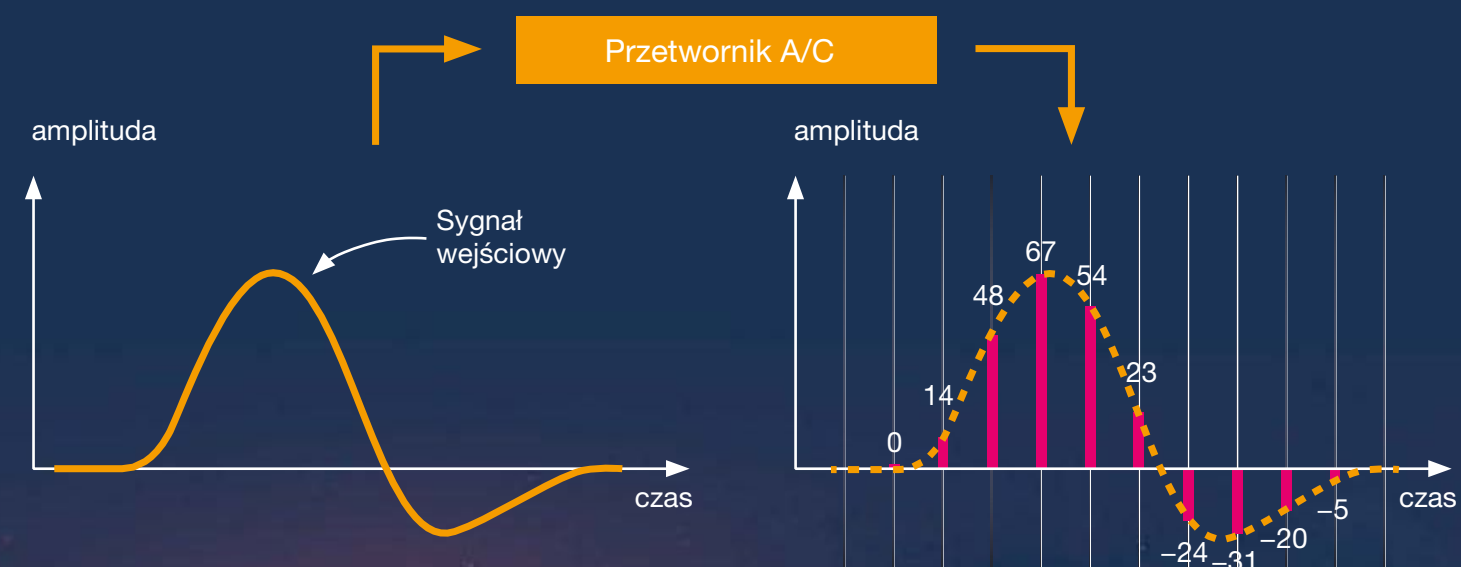
Jeśli posługujesz się jednocześnie przynajmniej dwoma systemami liczbowymi, stosuj dodatkowy indeks dolny do określenia podstawy systemu, w którym liczba została zapisana. Jest to szczególnie ważne w przypadku liczb, których postaci wyglądają tak samo w różnych systemach, ale mają różne wartości, np.: $100_{10} = 1100100_2$, a $100_2 = 4_{10}$.

JAK DANE ANALOGOWE SĄ PRZEKSZTAŁCANE NA CYFROWE?

Podczas nagrywania filmu smartfonem dźwięk i ruchomy obraz są przetwarzane na postać cyfrową. Dzięki temu możemy przechowywać je w pamięci telefonu, a następnie odtwarzać. Jak działają cyfrowe urządzenia rejestrujące dźwięk i obraz? Przeanalizujmy to na przykładzie.

Mikrofon wbudowany w telefon zamienia falę dźwiękową na sygnały elektryczne odpowiadające amplitudzie i częstotliwości dźwięku.

Przetwornik A/C (analogowo-cyfrowy) próbuje amplitudy sygnałów elektrycznych, czyli mierzy je tysiące razy w ciągu jednej sekundy i zamienia na ciągi liczb binarnych. Aby zapis cyfrowy był zgodny z dźwiękiem, który słyszy ludzkie ucho, próbkowanie musi się odbywać z odpowiednio wysoką częstotliwością i rozdzielczością. Na przykład dla płyt CD standardem są częstotliwość 44,1 kHz i rozdzielczość 16 bitów na próbkę, czyli $2^{16} = 65\,536$ możliwych wartości amplitudy fali dźwiękowej.

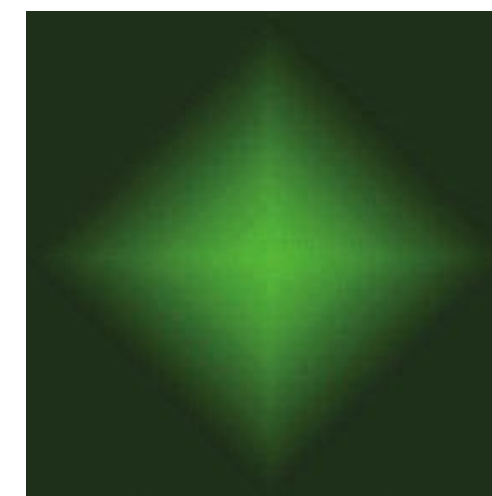


Jak w komputerze zapisuje się obrazy cyfrowe?

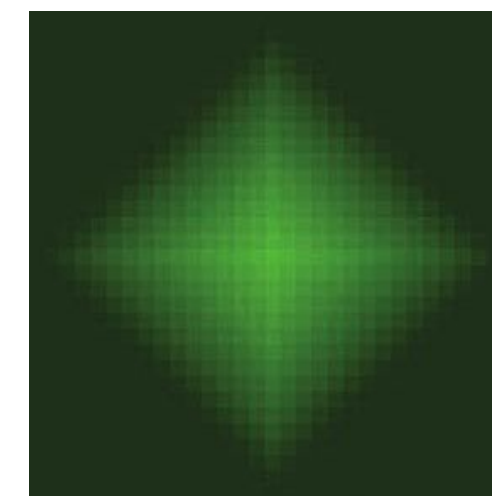
Aby wyjaśnić dokładniej istotę cyfrowego zapisu informacji, posłużymy się przykładem z dziedziny grafiki komputerowej. Zobaczmy, jakie informacje może zawierać jeden piksel obrazu.

Rysunek 7.4 to przykład gradientu, czyli wypełnienia obszaru równomiernym przejściem z jednego odcienia tej samej barwy do innego.

Rysunek 7.5 powstał z rysunku 7.4 w wyniku zmniejszenia rozdzielczości, czego efektem jest mniejsza liczba pikseli.



Rys. 7.4. Gradient



Rys. 7.5. Gradient po zmniejszeniu rozdzielczości obrazu

Warto wiedzieć

Graficy komputerowi i osoby zajmujące się projektowaniem wizualnych aspektów stron internetowych informacje o kolorach zwykle zapisują z użyciem systemu szesnastkowego.

Dzięki zastosowaniu odpowiedniej **rozdzielczości** mamy wrażenie, że obraz z rysunku 7.4 jest ciągły – kolory płynnie przechodzą jeden w drugi. Obraz z rysunku 7.5 sprawia wrażenie zbioru kwadratów o różnych odcieniach zieleni, które są ułożone schodkowo.

Przyjrzyjmy się dokładnie rysunkowi 7.5. W poziomie i w pionie mamy po 31 pikseli. Na rysunku 7.6 pokazano środkową linię tego obrazu w powiększeniu:



Rys. 7.6. Środkowa linia obrazu w powiększeniu

W cyfrowej reprezentacji każda informacja, również informacja o obrazie, musi mieć postać liczbową. Każdemu z odcieni można przypisać liczbę całkowitą, począwszy od 0.

Jeśli przyjąć, że odcień o najmniejszej intensywności (jasności) opiszemy liczbą 0, a ten o największej – 15, to piksele tworzące tę linię obrazu opiszemy ciągiem 31 liczb. Przedstawiono to na rysunku 7.7.



Rys. 7.7. Środkowa linia obrazu z ponumerowanymi odcieniami

Rozdzielczość obrazu

podręcznik *NOWA Informatyka na czasie 1. Zakres podstawowy* s. 46

Warto wiedzieć

Czasami zamiast słowa *piksel* używa się słowa *pel* (ang. *picture element*). Stosuje się je zwłaszcza wtedy, gdy mówimy o elementach obrazu wydrukowanego.

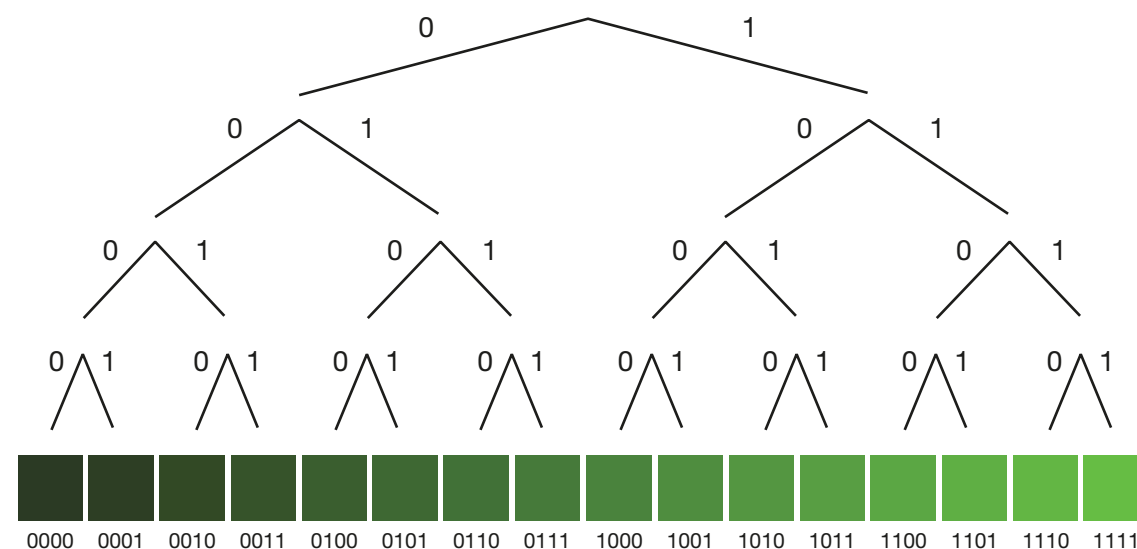
Każda z tych liczb dziesiętnych w pamięci komputera lub na nośniku danych jest w rzeczywistości zapisywana jako czterocyfrowa liczba dwójkowa. Aby zrozumieć, w jaki sposób są tworzone kody kolorów, przeanalizujemy diagram z rysunku 7.8.

Różnym odcieniom koloru zielonego przyporządkowano liczby binarne od 0000 do 1111. **Najbardziej znaczący bit** – reprezentowany przez pierwszą od lewej cyfrę liczby binarnej – decyduje o tym, czy dany odcień jest w grupie ciemniejszych czy jaśniejszych.

Drzewo decyzyjne • Diagram z rysunku 7.8 ma formę **drzewa decyzyjnego** – czytamy je od góry. Jeśli postawilibyśmy pytanie „Czy dany odcień jest jednym z jaśniejszych?”, to odpowiedź „tak” jest reprezentowana przez cyfrę 1, a odpowiedź „nie” – przez 0. Dlatego wszystkie odcienie bliższe czerni mają pierwszy bit o wartości 0.

Drugi bit określa, czy odcień jest wśród $\frac{1}{4}$ najciemniejszych czy wśród $\frac{1}{4}$ średnio ciemnych (lub analogicznie $\frac{1}{4}$ najjaśniejszych czy w $\frac{1}{4}$ średnio jasnych).

Po wyznaczeniu trzeciego bitu wybór organicza się do dwóch odcieni, **Bit najmniej znaczący** a **najmniej znaczący bit** – reprezentowany przez ostatnią cyfrę liczby binarnej – pozwala jednoznacznie określić odcień docelowy.



Rys. 7.8. Diagram ilustrujący tworzenie kodów odcieni w obrazie

 Warto wiedzieć

W pierwszych komputerach w latach 40. XX w. nie korzystano z systemu binarnego. W słynnym komputerze ENIAC zastosowano system dwupiętkowy.

Każdy odcień koloru zielonego na obrazie można więc jednoznacznie przedstawić jako czterocyfrową liczbę binarną. Do opisanego jednego piksela obrazu z rysunku 7.7 ze s. 107 potrzeba zatem 4 bitów.

 Zapamiętaj

Wszelkiego typu informacje przechowywane i przetwarzane w systemach komputerowych są przedstawiane (kodowane) jako liczby binarne, w postaci ciągu cyfr: 0 i 1.

7.3. Algorytmy zamiany reprezentacji liczb

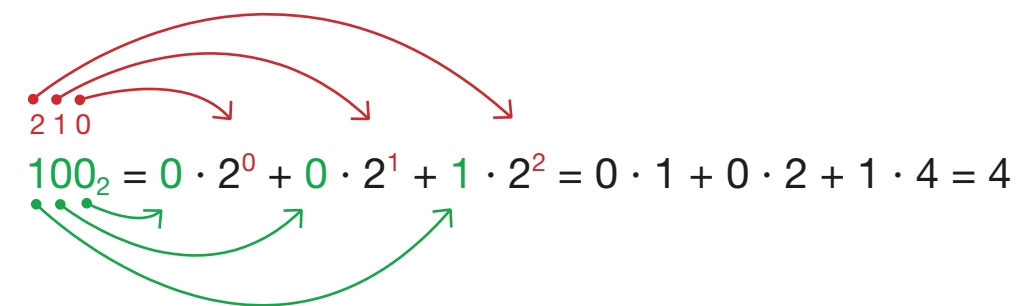
W komputerze liczby zapisuje się w systemie dwójkowym, jednak użytkownicy są przyzwyczajeni do używania systemu dziesiętnego. Dlatego np. podczas pracy z kodem HTML, tworzenia grafiki komputerowej czy konfiguracji sieci pojawia się potrzeba konwersji między systemami liczbowymi.

Zamiana liczby binarnej na liczbę dziesiętną

Zapis dziesiętny liczby binarnej można wyznaczyć, sumując kolejne potęgi liczby 2 odpowiadające wagom kolejnych cyfr. Algorytm można zapisać następująco:

- ➊ Ponumeruj cyfry liczby binarnej od prawej do lewej, zaczynając od 0.
- ➋ Każdą cyfrę potraktuj odpowiednio jako liczbę dziesiętną 0 albo 1 i pomnóż przez liczbę 2 podniesioną do takiej potęgi, jaki jest numer cyfry.
- ➌ Zsumuj otrzymane iloczyny.

Zamianę liczby binarnej 100_2 na postać dziesiętną ilustruje rysunek 7.9.



Rys. 7.9. Schemat zamiany liczby binarnej na dziesiętną

Ćwiczenie 2

Zapisz poniższe liczby binarne w systemie dziesiętnym.

- 111_2
- 10001_2
- 1101101_2

Zamiana liczby dziesiętnej na liczbę binarną

Założmy, że chcemy zapisać w postaci binarnej liczbę dziesiętną 123. Aby wyznaczyć kolejne cyfry (bity) zapisu binarnego, musimy wskazać potęgi liczby 2, które w sumie dają tę liczbę.

Zacznijmy od wyznaczenia najmniej znaczącego, czyli ostatniego bitu. Z podanych wcześniej **własności liczb binarnych** wiemy, że wszystkie liczby nieparzyste kończą się cyfrą 1 (wynika to z własności 1), dlatego zapis binarny liczby 123 będzie się kończyć cyfrą 1.

 Warto wiedzieć

Liczby większe od 1 zapisane w postaci binarnej składają się z większej liczby cyfr niż ich odpowiedniki w postaci dziesiętnej. Z punktu widzenia człowieka to z pewnością wada, dlatego binarnym systemem liczbowym nie posługujemy się na co dzień.

Podstawowe własności liczb binarnych

→ s. 104

Liczby 122 i 123 to kolejne liczby całkowite, więc z własności 4 wiemy, że wszystkie bity poza ostatnim będą takie same jak bity liczby 122. Przedostatni bit liczby 122 (a więc i 123) będzie taki, jak ostatni bit liczby o połowę mniejszej, a więc 61 (wynika to z własności 3). Ta liczba jest nieparzysta, więc jej postać binarna będzie się kończyć cyfrą 1. To oznacza, że zapis binarny liczby 122 kończy się parą cyfr 10 (jeden, zero), a liczby 123 – parą 11 (jeden, jeden).

Jak znaleźć kolejną cyfrę (trzecią od prawej) zapisu binarnego liczby 123? Wystarczy powtórzyć powyższe rozumowanie: określić parzystość liczby $(61 - 1)/2$. Wyznaczanie parzystości kolejnych ilorazów należy powtarzać do momentu, gdy otrzymamy iloraz równy 0.

Przebieg tego algorytmu można przedstawić w tabelarycznej formie (tab. 7.2).

Dobra rada
Kolejne etapy realizacji danego algorytmu możesz w wygodny sposób umieścić w tabeli. Może ona pomóc w zrozumieniu trudniejszego algorytmu oraz pozwala znaleźć ewentualne błędy.

Liczba	Iloraz z dzielenia przez 2	Reszta z dzielenia przez 2
123	61	1
61	30	1
30	15	0
15	7	1
7	3	1
3	1	1
1	0	1

Tabela 7.2. Przebieg algorytmu zapisu liczby 123 w postaci binarnej

Ostatnią kolumnę czyta się od dołu i zapisuje cyfry od lewej do prawej. Liczba 123_{10} w zapisie binarnym to 1111011_2 .

A to ciekawe

Komputery analogowe

Maszyny analogowe, inaczej niż urządzenia cyfrowe, przetwarzają dane w postaci analogowej, tj. za pomocą wielkości fizycznych zmieniających się w sposób ciągły, takich jak napięcie elektryczne. Jednym z pierwszych komputerów analogowych na świecie był analizator AKAT-1, skonstruowany w 1957 r. przez zespół pod kierunkiem inż. Jacka Karpińskiego. Komputer pomagał naukowcom w analizie dynamicznie zmieniających się procesów, takich jak przepływy cieczy, wymiana ciepła, resorowanie pojazdów. Ergonomiczną obudowę komputera zaprojektowano na warszawskiej Akademii Sztuk Pięknych.



Ćwiczenie 3

Wyznacz postać binarną liczby dziesiętnej 117. Zapisz przebieg algorytmu w tabeli. Jego poprawność możesz sprawdzić np. w kalkulatorze systemu Windows (tryb programisty) albo korzystając z odpowiedniej aplikacji w telefonie lub w serwisie internetowym.

Dobra rada
Jeśli chcesz wyznaczyć postać binarną wielu liczb, możesz skorzystać z funkcji `DZIES.NA.DWÓJK` lub `DEC2BIN` w arkuszu kalkulacyjnym.

7.4. Program zamieniający liczbę dziesiętną na binarną

Przygotujemy się teraz do stworzenia programu, który po wpisaniu z klawiatury liczby naturalnej, nie większej niż 255, wypisze ciąg zer i jedynek przedstawiający binarną reprezentację komputerową tej liczby. Ograniczymy się do liczb, które da się zapisać w jednym bajcie (ang. *byte*), czyli jako liczby binarne złożone z ośmiu bitów.

Zapiszmy specyfikację problemu algorytmicznego.

Specyfikacja

Dane: liczba całkowita (nieujemna) nie większa niż 255.

Wynik: zapis binarny liczby całkowitej.

Warto wiedzieć
Jeden bajt reprezentuje aż $2^8 = 256$ różnych liczb binarnych. Tę właściwość wykorzystano w kodzie ASCII, w którym za pomocą jednobajtowych kodów przedstawia się litery alfabetu łacińskiego (języka angielskiego), cyfry, znaki przestankowe, symbole i inne znaki służące do komunikacji tekstowej.

Aby wyznaczyć postać binarną liczby dziesiętnej, możemy postępować w sposób przedstawiony w tabeli 7.2. Wystarczy zapamiętywać kolejne ilorazy z dzielenia przez 2 oraz otrzymywane reszty z dzielenia. Dokładny zapis działania algorytmu przedstawia poniższa lista kroków.

- 1 Podziel liczbę przez 2, zapisz resztę z dzielenia i zapamiętaj iloraz (część całkowitą).
- 2 Dopóki ostatnio otrzymany iloraz jest większy od 0, powtarzaj kroki 3, 4 i 5.
- 3 Wyznacz resztę z dzielenia ostatnio zapamiętanego ilorazu przez 2.
- 4 Zapisz resztę z dzielenia przed ostatnią, wcześniej zapisaną resztą.
- 5 Podziel iloraz przez 2 i zapamiętaj część całkowitą wyniku jako nowy iloraz.

Po otrzymaniu ilorazu zerowego postać binarną tworzą zapisane reszty.

Do zapisania algorytmu w języku Python potrzebna będzie nam nowa wiedza, ponieważ będziemy wyznaczać reszty z dzielenia oraz zapamiętywać kolejne ilorazy (części całkowite). Do tego celu wykorzystamy operator wyznaczania reszty z dzielenia oraz strukturę listy.

Dzielenie całkowite i reszta z dzielenia

W programowaniu rozróżnia się dwie operacje dzielenia, gdy dzielna i dzielnik są liczbami całkowitymi: dzielenie w sensie znanym ze szkoły podstawowej (np. operacja 123/2 daje iloraz równy 61,5) i dzielenie całkowite (operacja 123/2 daje iloraz całkowity 61 i resztę 1).

W języku Python do wyznaczania ilorazu z możliwą częścią ułamkową służy operator /, a do wyznaczania ilorazu całkowitego (części całkowitej ilorazu) – operator //. Resztę z dzielenia wyznaczamy z wykorzystaniem operatora %, zwanego **modulo**. Na przykład wartością zmiennej iloraz po przypisaniu `iloraz = 123//2` będzie 61, a wartością zmiennej reszta po przypisaniu `reszta = 123%2` będzie 1.

Modulo to operacja wyznaczania reszty z dzielenia jednej liczby przez drugą.

Ćwiczenie 4

```
Sprawdź efekt działania programu, którego kod ma postać:
iloraz = 123//2
reszta = 123%2
print(iloraz)
print(reszta)
```

Warto wiedzieć
W języku Python modulo można stosować również dla liczb zmiennoprzecinkowych. Na przykład: 4.5 % 3 = 1.5

Wykorzystanie pętli while do zamiany liczby dziesiętnej na binarną

Kolejne reszty z dzielenia liczby dziesiętnej przez 2 możemy wyznaczyć, wykorzystując pętlę while.

```
1. liczba = int(input("Podaj liczbę:"))
2.
3. while liczba > 0:
4.     print(liczba % 2, end="")
5.     liczba = liczba // 2
```

W 1. wierszu program prosi o podanie liczby dziesiętnej z klawiatury, a następnie zapisuje ją w zmiennej `liczba`. W wierszach 3.–5. powtarzamy w pętli wyznaczanie reszty z dzielenia liczby przez 2 i zapamiętywanie ilorazu całkowitego w zmiennej `liczba`.

W wierszu 4. znajduje się dodatkowy parametr `end=""` funkcji `print`. Skutkiem zapisu `end=""` jest brak odstępu między wypisywanymi cyframi.

Reszta z dzielenia jest za każdym razem wypisywana na ekranie. Kroki pętli są powtarzane, dopóki iloraz jest większy od zera.

Powyższy program ma jedną wadę: reszty są wypisywane w kolejności ich wyznaczania, a powinny być zapisane odwrotnie. Rozwiązaniem tego problemu jest wprowadzenie nowej struktury danych do zapisania kolejnych reszt. Ta struktura powinna pozwalać na prostą zamianę kolejności zawartych w niej elementów.

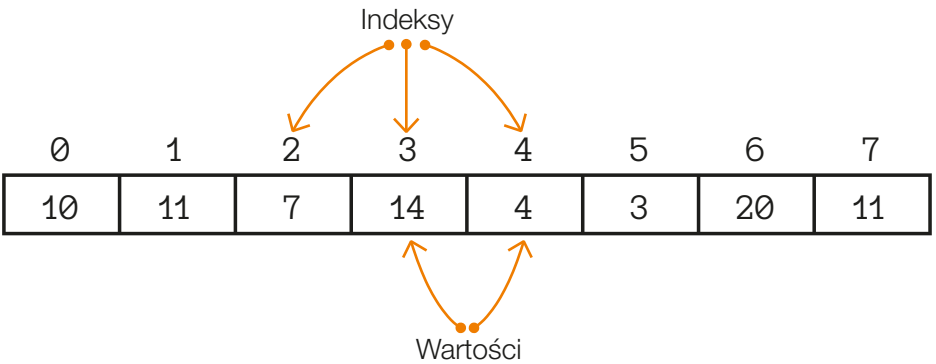
Lista – struktura danych

Aby zapisać w pamięci i przetwarzać w programie większą liczbę danych, np. liczby całkowite, bez konieczności tworzenia osobnej zmiennej dla każdej z nich, można się posłużyć strukturą danych nazywaną **listą** (ang. *list*).

Lista

Lista jest uporządkowana, tzn. każda zapisana na niej wartość ma określone miejsce. Listę liczb można porównać do wiersza tabeli, gdzie w kolejnych komórkach są zapisane kolejne liczby. Każda wartość zapisana na liście ma swoją współrzędną, zwaną **indeksem**. Listy są indeksowane od 0 (rys. 7.10).

Indeks



Warto wiedzieć

Przypisanie wartości do wybranego elementu listy oznacza zastąpienie wartości dotychczas przechowywanej nową wartością.

Rys. 7.10. Interpretacja graficzna listy o 8 wartościach liczbowych

Aby odwołać się do jakiegoś elementu listy, wystarczy podać jego współrzędną (indeks).

Instrukcje:

```
lista[0] = 10
lista[7] = 11
```

powodują przypisanie wartości 10 pierwszemu elementowi listy (o indeksie 0) i przypisanie wartości 11 ostatniemu elementowi (o indeksie 7).

Listy wartości tworzy się z użyciem nawiasów kwadratowych, które zapisuje się po operatorze przypisania. Między nawiasami wstawia się kolejne wartości, oddzielone przecinkami. Aby utworzyć listę o nazwie `lista` zawierającą osiem konkretnych wartości całkowitych, należy napisać: `lista = [1, 11, 7, 14, 4, 3, 20, 10]`.

Struktura listy może przechowywać elementy różnego typu. Przeanalizujmy zapis:

```
1. a = 12 * 3
2. lista = ["Marta", 'kot', 's', a, -123, 5.23]
3. print(lista)
```

W powyższym kodzie do listy wprowadzono napisy, literę, wartość zmiennej `a`, a także liczby – całkowitą oraz wymierną. Instrukcja z wiersza 3. umożliwia wypisanie całej zawartości listy.

Liczba elementów listy może się zmieniać podczas działania programu. Można również utworzyć w pamięci komputera listę bez elementów, pisząc w kodzie źródłowym np. `lista = []`.

Listy – wczytywanie i wypisywanie danych

Napiszemy krótki program o nazwie *Cyfry wspak*, który po wczytaniu z klawiatury ośmiu cyfr (liczb jednocyfrowych) wyświetli je w odwróconej kolejności.

Kod źródłowy programu *Cyfry wspak*

```
1. lista = [0]
2.
3. for i in range(8):
4.     lista.append(int(input()))
5.
6. for cyfra in reversed(lista):
7.     print(cyfra)
```

Kod w wierszu 1. pozwala na utworzenie listy składającej się z ośmiu zer. W dalszej części programu będziemy zamieniać te wartości (zera na liście) na wartości, które wprowadzi użytkownik. W wierszach 3. i 4. zapisano kod pętli, która służy do wprowadzenia docelowych ośmiu liczb z klawiatury. Wykorzystaliśmy **metodę** `append`, która pozwala na dodawanie nowych elementów do listy i dynamicznie powiększa długość tej listy.

Metoda `append`

Wykorzystaliśmy tutaj **pętlę** `for`, która w przeciwieństwie do pętli `while` nie czeka na spełnienie określonego warunku, ale ma z góry ustaloną liczbę powtórzeń. W naszym kodzie ta liczba jest przekazana przez **funkcję** `range`, która służy do generowania ciągów liczb o długości podanej jako jej argument.

Pętla `for`

Funkcja `range`

Do wskazania położenia liczby na liście użyto zmiennej pomocniczej `i`, która jest jednocześnie licznikiem pętli. Ponieważ liczby mają być przez program wypisane w odwróconej kolejności (od elementu z indeksem 7 do elementu z indeksem 0), w pętli z wierszy 6. i 7. zastosowano **funkcję** `reversed`. Umożliwia ona dostęp do elementów listy w kolejności od ostatniego.

Funkcja `reversed`

Zauważ, że zmienna pomocnicza `cyfra` służy zarówno do kontroli liczby iteracji (powtórzeń pętli), jak i do ustalenia pozycji cyfry w liczbie.

Ćwiczenie 5

- a. Przepisz kod programu *Cyfry wspak* i sprawdź jego działanie dla różnych zestawów danych.
- b. Przyjmij, że liczby na liście od razu powinny być zapamiętane w kolejności odwróconej. Napisz program realizujący takie założenie. Kod programu zapisz w pliku *cyfry_wprost.py*.

Dobra rada
Aby wyświetlić wartości znajdujące się na liście o nazwie `L`, napisz `print(L)`. Wartości zostaną wypisane w nawiasie kwadratowym i oddzielone przecinkami.

Program *Zamiana systemów liczbowych*

Teraz możemy wykorzystać listy do napisania programu *Zamiana*, realizującego algorytm zamiany liczby dziesiętnej na liczbę dwójkową.

Oto kod programu znajdujący się w pliku *zamiana.py*:

Kod źródłowy programu *Zamiana*

```
1. cyfry = []
2.
3. liczba = int(input("Podaj liczbę:"))
4. iloraz = liczba
5.
6. while iloraz > 0:
7.     cyfry.append(iloraz % 2)
8.     iloraz = iloraz // 2
9.
10. print("Zapis binarny liczby", liczba, "to", end=" ")
11. for cyfra in reversed(cyfry):
12.     print(cyfra, end=" ")
```

W wierszu 1. tworzymy pustą listę. Następnie w wierszu 3. program prosi o podanie liczby do zamiany i zapisuje ją w zmiennej `liczba`. W wierszu 4. tę samą liczbę przypisuje zmiennej `iloraz`, której wartość będzie się zmieniała w pętli. W wierszach 6.–8. za pomocą pętli `while` do listy dodawane są kolejne reszty z dzielenia liczby przez 2. Z kolei w zmiennej `iloraz` jest zapisywany iloraz całkowity z tego dzielenia.

W wierszach 10.–12. podana wcześniej liczba w systemie dziesiętnym jest wypisywana na ekranie w postaci binarnej. Wykorzystywana jest pętla `for` do wypisania kolejnych elementów listy oraz funkcja `reversed`, aby te elementy wypisać od końca do początku.

Ćwiczenie 6

- a. Zapisz kod źródłowy programu *zamiana.py* i uruchom program dla różnych liczb (mniejszych niż 256) wpisanych z klawiatury. Zaczynij od sprawdzenia liczby 123.
- b. Zmień kod źródłowy programu *zamiana.py* tak, aby program pozwalał zapisywać liczby za pomocą dwóch bajtów (16 bitów).

Zapamiętaj

Lista to struktura programistyczna służąca do przechowywania wielu danych w uporządkowany sposób. Dane są przechowywane w ponumerowanych (indeksowanych) komórkach pamięci. Na liście n -elementowej pierwszy element ma indeks 0, a ostatni: $n - 1$.

 Zadania

- * **1** Najprostszym systemem liczbowym jest kod jedynekowy (unarny). Poszukaj informacji na temat tego systemu liczbowego. Podaj przykład jego praktycznego zastosowania w życiu codziennym.
- * **2** Napisz program, który będzie określać parzystość liczby wprowadzonej z klawiatury. W przypadku liczby parzystej powinien się wyświetlić komunikat „Liczba jest parzysta”, natomiast w przypadku liczby nieparzystej – komunikat „Liczba jest nieparzysta”.
- * **3** Napisz program, który zapisze na liście dziesięć kolejnych liczb nieparzystych, począwszy od 1, i wyświetli zawartość listy.
- * **4** Napisz program, który wczyta na listę podane przez użytkownika liczby, a następnie wypisze je w odwrotnej kolejności. Przyjmij, że liczba 0 kończy wprowadzanie danych.
- * **5** Napisz program, który dla podanych dwóch liczb dwucyfrowych wypisze ich postać binarną, a następnie postać binarną ich sumy.
- ** **6** Napisz program, który dla liczby wprowadzonej z klawiatury wyznaczy największą potęgę liczby 2, nie większą od tej liczby.
- ** **7** Napisz program, który wczyta 8 liczb na listę, a następnie każdą z nich zmniejszy o 1.
- ** **8** Napisz program, który wczyta liczbę z klawiatury i wyznaczy jej dzielniki mniejsze od tej liczby. Na przykład po wczytaniu liczby 18 na ekranie powinny zostać wypisane liczby 1, 2, 3, 6 i 9.
- ** **9** W słynnym komputerze ENIAC stosowano kodowanie oparte na systemie dwupiętkowym, znane wcześniej z japońskich liczydeł, tzw. sorobanów. Przygotuj krótką prezentację lub artykuł na stronę internetową szkoły na temat tego systemu liczbowego i jego zastosowania do kodowania liczb w komputerze ENIAC.
- ** **10** Napisz program, który dla liczby wprowadzonej z klawiatury wypisze po przecinku wszystkie jej dzielniki jednocyfrowe (łącznie z liczbą 1).
- ** **11** Napisz program odwracający kolejność cyfr w liczbie całkowitej. Program może zapamiętywać cyfry w strukturze listy.
- ** **12** Napisz program, który będzie określać parzystość liczby wprowadzonej z klawiatury. Program nie powinien korzystać z operatora %.
- *** **13** Napisz program tworzący zestawienie liczb binarnych odpowiadających liczbom dziesiętnym od 0 do 255. Efektem działania programu powinny być dwie kolumny: po lewej – liczby dziesiętne, a po prawej – liczby binarne.
- *** **14** Napisz program odwracający kolejność cyfr w liczbie całkowitej. Program nie może zapamiętywać cyfr w strukturze listy.

8 **Liczby pierwsze**

Liczby pierwsze znasz z lekcji matematyki w szkole podstawowej. Choć odkryto je już w starożytności, przez lata twierdzenia dotyczące liczb pierwszych uchodziły wśród matematyków za piękne, ale bezużyteczne. Dopiero w drugiej połowie XX w. znalazły praktyczne zastosowanie w informatyce, zwłaszcza w kryptografii. Te twierdzenia są podstawą działania niektórych protokołów szyfrowania, czyli algorytmów, które zapewniają poufność komunikacji w Internecie i bezpieczeństwo transakcji bankowych. W tym temacie dowiesz się więcej o liczbach pierwszych i poznasz ich znaczenie.

Ważne w tym temacie

- Liczby złożone i liczby pierwsze
- Podzielność liczb całkowitych
- Algorytmy sprawdzające, czy liczba jest pierwsza
- Funkcje w języku Python
- Funkcja `sqrt` z modułu `math`

8.1. Liczby złożone i liczby pierwsze

Dodatknie liczby naturalne, które można przedstawić w postaci iloczynu liczb naturalnych mniejszych od nich, nazywamy **liczbami złożonymi**.  **Liczby złożone** Ich przykładami są: $9 = 3 \cdot 3$, $10 = 2 \cdot 5$, $120 = 2 \cdot 3 \cdot 4 \cdot 5$, $1115 = 3 \cdot 5 \cdot 7 \cdot 11$.

Pozostałe liczby, z wyjątkiem liczby 1, są liczbami pierwszymi. **Liczbą pierwszą** nazywamy więc liczbę większą od 1, która jest podzielna wyłącznie przez 1 i przez samą siebie. Wśród liczb mniejszych od 100 jest dwadzieścia pięć liczb pierwszych: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97. Na rysunku 8.1 oznaczono je pomarańczowym tłem.

Jedno z najważniejszych twierdzeń matematycznych dotyczących liczb pierwszych nazywane jest **podstawowym twierdzeniem arytmetyki**  **Podstawowe twierdzenie arytmetyki** i brzmi następująco: każda liczba naturalna (większa od 1) jest albo liczbą pierwszą, albo iloczynem liczb pierwszych.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	

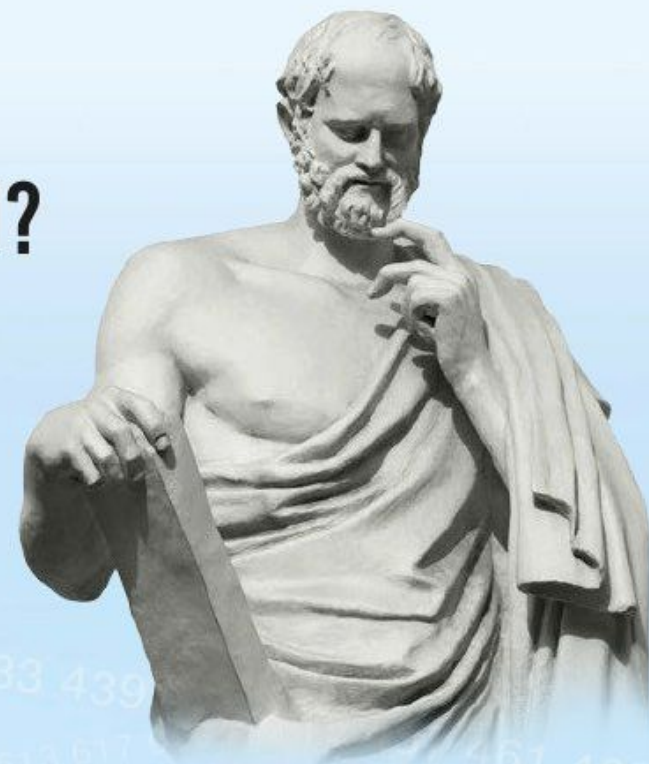
Rys. 8.1. Liczby złożone i liczby pierwsze mniejsze niż 100

 **Liczba pierwsza**

 Warto wiedzieć

Liczba 1 jest wyjątkowa. Przez setki lat uważano ją za liczbę pierwszą. Dziś przyjmuje się, że nie jest ani pierwsza, ani złożona.

CO WARTO WIEDZIEĆ O LICZBACH PIERWSZYCH?



Ile jest liczb pierwszych?

Liczyby pierwsze odkryto i badano już w starożytności. W IV w. p.n.e. Euklides udowodnił, że jest ich nieskończenie wiele. Dotąd nie odkryto jednak wzoru, według którego można byłoby dla konkretnej liczby n wyznaczyć n -tą liczbę pierwszą.

2 to najmniejsza i jedyna parzysta liczba pierwsza.

Liczyby postaci $2^n - 1$, gdzie n jest liczbą naturalną, nazywamy liczbami Mersenne'a. Wiele z nich jest liczbami pierwszymi.

Liczyby pierwsze większe od 5 kończą się cyframi 1, 3, 7, 9.



Liczyby czworaczce to cztery liczby pierwsze mające postać: $p, p+2, p+6, p+8$.

Liczyby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.

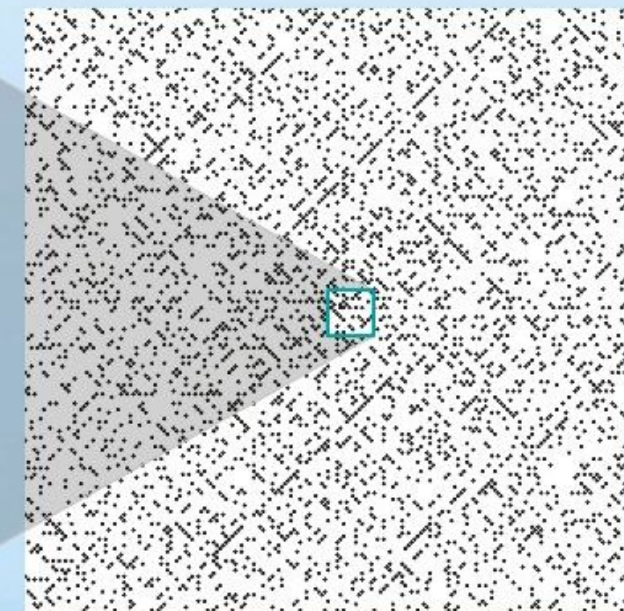
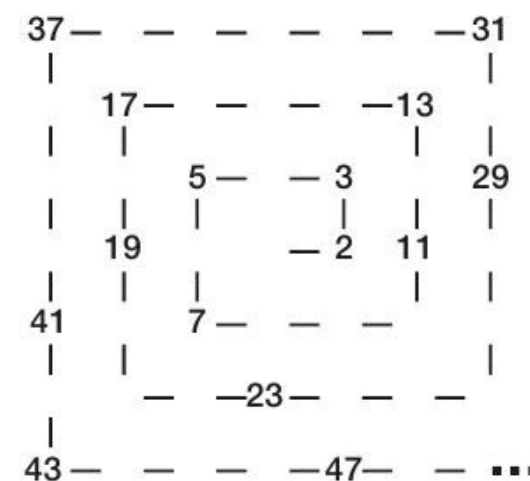
Liczyby pierwsze w kryptografii

Liczbę utworzoną z iloczynu dwóch dużych liczb pierwszych bardzo trudno w rozsądnym czasie rozłożyć na czynniki pierwsze, nawet z użyciem najszybszych komputerów. Wykorzystali to twórcy algorytmu RSA, który stosuje się do szyfrowania komunikacji w sieci Internet oraz w bankowości elektronicznej. Liczyby pierwsze używane w tym algorytmie mają nawet 2048 bitów, czyli składają się z 2048 cyfr binarnych.



Spirala Ulama

W 1963 r. polski matematyk Stanisław Ulam przedstawił graficznie rozmieszczenie liczb pierwszych. Na kwadratowej tablicy spiralnie wypisał kolejne liczby naturalne. Po naniesieniu bardzo wielu liczb okazało się, że na niektórych liniach liczby pierwsze grupują się częściej niż na innych. Do tej pory nie wyjaśniono, dlaczego tak się dzieje.



Jak liczby pierwsze pomagają rozpoznawać anagramy?

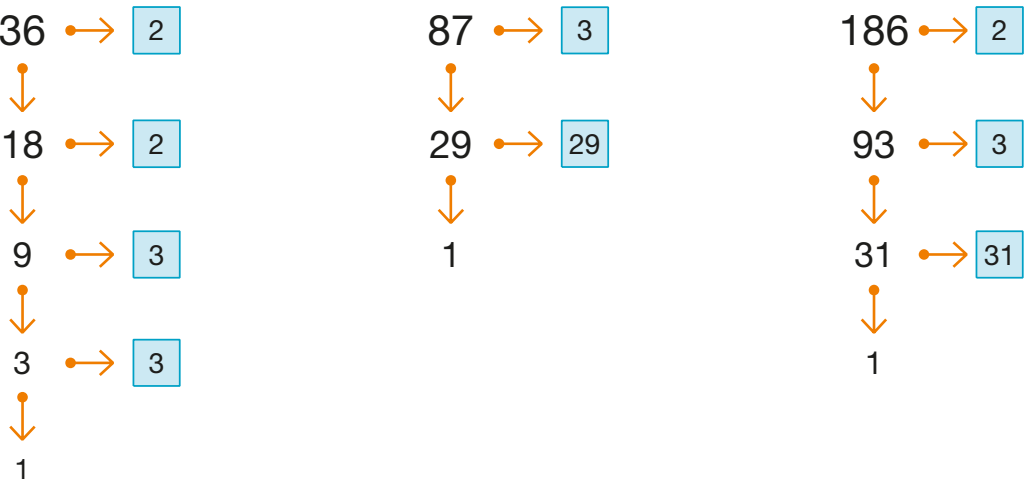
Rozkład dowolnej liczby na czynniki pierwsze jest jednoznaczny (np. dla 30 jest to wyłącznie rozkład $2 \cdot 3 \cdot 5$). Ta własność jest podstawą działania pewnego algorytmu sprawdzającego, czy dwa wyrazy są anagramami (składają się z tych samych liter). Kolejnym literom alfabetu przypisuje się kolejne liczby pierwsze. Jeśli iloczyny liczb odpowiadających literom w wyrazach są równe, wyrazy są anagramami.

$$\begin{array}{ccc} 3 & 2 & 5 \\ | & | & | \\ \text{b} & \text{a} & \text{c} \end{array} \quad 3 \cdot 2 \cdot 5 = 30$$

$$\begin{array}{ccc} 2 & 3 & 5 \\ | & | & | \\ \text{a} & \text{b} & \text{c} \end{array} \quad 2 \cdot 3 \cdot 5 = 30$$

Rozkład na czynniki pierwsze

Iloczyn liczb pierwszych, który tworzy daną liczbę, nazywamy **rozkładem na czynniki pierwsze**. Jest on jedyny w swoim rodzaju, tzn. każda liczba ma tylko jedną unikatową formułę rozkładu, która do niej przynależy. Na rysunku 8.2 przedstawiono rozkład na czynniki pierwsze liczb 36, 87 i 186. Te liczby możemy zapisać następująco: $36 = 2 \cdot 2 \cdot 3 \cdot 3$; $87 = 3 \cdot 29$; $186 = 2 \cdot 3 \cdot 31$.



Rys. 8.2. Rozkład liczb 36, 87 i 186 na czynniki pierwsze

Warto wiedzieć

Stosunkowo łatwo sprawdzić, czy podana liczba nieparzysta jest podzielna przez 5, ponieważ kończy się ona cyfrą 5.

Okazuje się, że im większa liczba, tym trudniej sprawdzić, czy jest pierwsza czy złożona, a więc czy da się ją rozłożyć na czynniki pierwsze. W przypadku dużych liczb rozstrzygnięcie tego w rozsądnym czasie stanowi zadanie praktycznie niewykonalne, nawet dla współczesnych komputerów. Ta trudność jest podstawą bezpieczeństwa działania niektórych algorytmów kryptograficznych (np. algorytmu RSA).

Liczba 2 jest jedyną parzystą liczbą pierwszą, więc w praktyce poszukiwania liczb pierwszych wymagają sprawdzania tylko liczb nieparzystych.

A to ciekawe

Szybki test pierwszości

Dla bardzo dużych liczb bardzo trudno ustalić, czy są one pierwsze czy złożone. Powszechnie stosowanym i w praktyce najszybszym testem pierwszości jest test Millera–Rabina. Algorytm stworzył w 1976 r. amerykański profesor informatyki Gary Miller, a w 1980 r. ulepszył go urodzony we Wrocławiu Michael O. Rabin (na zdjęciu). Dzięki temu algorytmowi można przeprowadzić test pierwszości nawet dla ogromnych liczb, rzędu 150 cyfr. Ponieważ jest to test probabilistyczny, nie wskazuje on żadnych dzielników testowanej liczby. Oznacza to, że liczba może zostać błędnie rozpoznana jako pierwsza. Prawdopodobieństwo takiego zdarzenia jest jednak znikome – wynosi ok. 1 do 10^{12} .



8.2. Podzielność liczb

Zanim stworzymy program sprawdzający, czy podana liczba jest pierwsza, napiszemy prostsze programy związane z podzielnością. Wprowadzimy przy tej okazji nową konstrukcję programistyczną – funkcje.

Funkcje w języku Python

Aby rozwiązać bardziej złożony problem algorytmiczny, najlepiej rozłożyć go na prostsze podproblemy. W praktyce oznacza to wyodrębnianie fragmentów programów (podprogramów), które w języku Python nazywa się **funkcjami**.

Każda funkcja musi mieć nadaną nazwę. Po nazwie zawsze występują nawiasy okrągłe, w których podaje się **parametry funkcji**.

`def nazwa_funkcji(lista parametrów):`

instrukcje

Definicja funkcji składa się z nagłówka funkcji i bloku instrukcji, które stanowią jej treść. Nagłówek rozpoczyna się słowem kluczowym **def**, po którym zapisuje się nazwę funkcji, a w nawiasie listę parametrów (nazwę każdego z nich), o ile funkcja ma jakieś parametry. Wiersz nagłówka kończy się dwukropkiem.

W treści funkcji umieszcza się instrukcje, które tworzą podprogram (stanowią pewną całość). Wszystkie linie kodu wchodzące w skład bloku instrukcji są przesunięte w prawo, podobnie jak w przypadku pętli i instrukcji warunkowej.

Efekt działania funkcji albo wyświetla się na ekranie, albo za pomocą **instrukcji return** jest przekazywany (zwracany) z powrotem do instrukcji programu, która funkcję wywołała.

Dzięki konsekwentnemu stosowaniu funkcji tworzymy kod bardziej przejrzysty i łatwiejszy do modyfikowania w przyszłości. Unikamy także powtórzeń, ponieważ tę samą funkcję można wywołać wiele razy dla różnych parametrów.

Badanie podzielności przez 3

Aby zobaczyć, jak w praktyce korzysta się z funkcji, napiszemy program, który będzie sprawdzał, czy podana przez użytkownika liczba jest podzielna przez 3.

Sprawdzenie podzielności danej liczby przez 3 sprowadza się do ustalenia, czy reszta z dzielenia tej liczby przez 3 jest równa zero.

Oto specyfikacja problemu:

Specyfikacja

Dane: dwie liczby naturalne a i b .

Wynik: reszta z dzielenia liczby a przez liczbę b .

Funkcja

Parametr funkcji

Warto wiedzieć

Zazwyczaj nie zaleca się używania polskich znaków w nazwach funkcji i zmiennych.

Instrukcja return

Stworzymy funkcję o nazwie `ResztaZDzielenia`, zwracającą resztę z dzielenia liczby a przez liczbę b , a wywołamy ją dla podanej przez użytkownika liczby a i b równego 3.

Poniżej znajduje się kod źródłowy programu.

Kod źródłowy programu sprawdzającego podzielność liczby przez 3

```
1. def ResztaZDzielenia(a, b):
2.     return a % b
3.
4. liczba = int(input("Podaj liczbę: "))
5.
6. print("Reszta z dzielenia", liczba, end=" ")
7. print("przez 3 wynosi:", end=" ")
8. print(ResztaZDzielenia(liczba, 3))
```

Dobra rada

Staraj się tworzyć funkcje o możliwie szerokim zastosowaniu. Dzięki temu będzie można je wywołać dla różnych parametrów. Tak tworzone są również funkcje w arkuszach kalkulacyjnych.

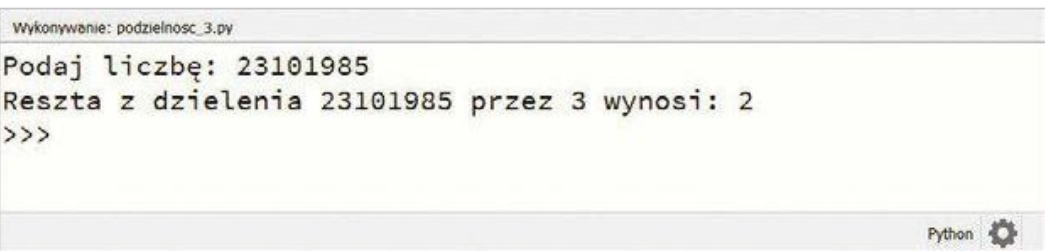
W wierszu 8. znajduje się wywołanie funkcji `ResztaZDzielenia` przy użyciu parametrów: wartości zmiennej `liczba` oraz 3. W ten sposób sprawdzimy podzielność podanej liczby przez 3. Kod w wierszu 4. odpowiada za nadanie przez użytkownika wartości zmiennej `liczba`.

Definicja funkcji `ResztaZDzielenia` znajduje się w wierszach 1. i 2. W nawiasach znajduje się lista parametrów funkcji. Te parametry przekazują wartości z programu głównego (odpowiednio: wartość zmiennej `liczba` oraz liczbę 3). Wiersz 2. to treść funkcji, która wyznacza resztę z dzielenia (z wykorzystaniem operatora `%`) i zwraca rezultat, stąd słowo kluczowe `return`.

Treść funkcji może się składać z wielu instrukcji. Jednak wszystkie one muszą być wcięte względem nagłówka o tę samą liczbę spacji (np. 4) lub wielokrotność tej liczby.

Zauważ, że tak zdefiniowana funkcja jest ogólna i może być zastosowana do sprawdzania podzielności dowolnych liczb.

Na rysunku 8.3 przedstawiono przykładowe wywołanie programu wyznaczającego resztę z dzielenia przez 3.



Rys. 8.3. Wywołanie programu sprawdzającego podzielność przez 3

Parametry formalny i aktualny funkcji

Parametry określone w momencie definiowania nowej funkcji nazywa się **parametrami formalnymi**. Natomiast parametry, dla których funkcja wywołana jest w programie głównym, to **parametry aktualne**. Parametr aktualny przekazuje więc faktyczną wartość (np. podaną przez użytkownika) do funkcji, która została wcześniej zdefiniowana.

Ćwiczenie 1

- a. Przepisz kod źródłowy programu sprawdzającego podzielność liczby przez 3 i sprawdź jego działanie na różnych przykładach.
- b. Zmodyfikuj treść programu tak, aby sprawdzał on podzielność podanej liczby przez 17.

Badanie podzielności przez kilka liczb jednocześnie

Zmodyfikujemy kod naszego programu tak, aby wyznaczał reszty z dzielenia podanej liczby przez cztery kolejne liczby pierwsze: 2, 3, 5, 7.

Wynikiem działania programu powinny być cztery liczby: reszta z dzielenia przez 2, reszta z dzielenia przez 3, reszta z dzielenia przez 5 i reszta z dzielenia przez 7.

Dzięki temu, że zdefiniowaliśmy wcześniej funkcję `ResztaZDzielenia`, modyfikacja programu polega wyłącznie na dopisaniu do programu głównego odpowiednich wywołań funkcji.

Oto kod źródłowy programu:

```
1. def ResztaZDzielenia(a, b):
2.     return a % b
3.
4. liczba = int(input("Podaj liczbę: "))
5.
6. print("Reszty z dzielenia", liczba, end=" ")
7. print("przez 2, 3, 5, 7", end=" ")
8. print("wynoszą odpowiednio:")
9. print(ResztaZDzielenia(liczba, 2))
10. print(ResztaZDzielenia(liczba, 3))
11. print(ResztaZDzielenia(liczba, 5))
12. print(ResztaZDzielenia(liczba, 7))
```

Fragment kodu źródłowego programu obliczającego reszty z dzielenia przez 2, 3, 5, 7

Ćwiczenie 2

- a. Przepisz kod źródłowy programu obliczającego reszty z dzielenia przez 2, 3, 5, 7 i sprawdź jego działanie dla różnych liczb. Instrukcję z wierszy 6.–8. zapisz w jednej linii.
- b. Skróć powyższy kod, korzystając z pętli `for`. Zastosuj listę `[2, 3, 5, 7]`.

Zapamiętaj

Funkcje definiujemy, gdy chcemy nazwać pewną grupę instrukcji i oddzielić fragment kodu od reszty programu – zwykle po to, aby go wielokrotnie wykorzystać.

8.3. Test pierwszośc

Matematycy od wieków zastanawiali się, jak sprawdzić, czy liczba jest pierwsza. Dziś naukowcy szukają przede wszystkim szybkich algorytmów, które będą to rozstrzygać.

Test pierwszośc – najprostszy algorytm

Aby stwierdzić, czy jakaś liczba n jest pierwsza czy złożona, należy sprawdzić, czy ma ona jakikolwiek dzielnik różny od 1 i od niej samej. Można to zrobić, dzieląc n kolejno przez 2, przez 3 itd., aż do $n - 1$.
Przed napisaniem programu komputerowego zapiszemy specyfikację.

Specyfikacja

- Dane:** liczba naturalna większa od 1.
- Wynik:** komunikat „Liczba pierwsza”, jeśli liczba jest liczbą pierwszą, komunikat „Liczba złożona” – w przeciwnym wypadku.

Poniżej znajduje się kod źródłowy programu *Test pierwszośc*.

Kod źródłowy programu
Test pierwszośc

```
1. def CzyPierwsza(n):
2.     for i in range(2, n):
3.         if n % i == 0:
4.             return False
5.     return True
6.
7. liczba = int(input("Podaj liczbę: "))
8.
9. if CzyPierwsza(liczba):
10.    print("Liczba pierwsza")
11. else:
12.    print("Liczba złożona")
```

Warto wiedzieć
W latach 90. XX w. pewien producent procesorów był zmuszony wymienić jeden z modeli, ponieważ wykonywał on operacje dzielenia z mniejszą dokładnością, niż zakładano. Wynosiła ona 9 zamiast 19 miejsc po przecinku. Ta wada ujawniła się przy szacowaniu stałej Bruna, która dotyczy liczb bliźniaczych.

W liniach 1.–5. znajduje się definicja funkcji *CzyPierwsza*. Jej celem jest sprawdzenie podzielności podanej przez użytkownika liczby przez liczby od 2 do $n - 1$. Gdy zostanie znaleziony dzielnik, funkcja zwraca wartość logiczną *False* (fałsz). Wówczas funkcja kończy swoje działanie i nie wykonuje kolejnych dzielen. Jeśli nie zostanie znaleziony dzielnik, to funkcja zwraca wartość logiczną *True* (prawda). Funkcja kończy wtedy swoje działanie (linia 5).
W programie głównym w linii 9. jest wywoływana funkcja *CzyPierwsza*. W zależności od zwróconej przez funkcję wartości logicznej na ekranie wyświetla się odpowiedni komunikat (linie 10. i 12.). Zapis *if CzyPierwsza(liczba)* możemy odczytać jako „jeśli *CzyPierwsza(liczba)* to prawda”.

Ćwiczenie 3

- a. Przepisz kod źródłowy programu i sprawdź jego działanie.
- b. Przygotuj tabelę, w której rozpiszesz kolejne etapy działania funkcji *CzyPierwsza* dla $n = 9$ oraz $n = 11$.

Program *Test pierwszośc*, choć daje poprawne wyniki, wykonuje wiele niepotrzebnych operacji wyznaczania reszt z dzielenia. Liczbę tych operacji możemy stosunkowo łatwo ograniczyć.

Test pierwszośc – ulepszony algorytm

Sprawdzanie podzielności podanej liczby przez liczby parzyste większe od 2 nie ma sensu. Wystarczy więc zbadać podzielność przez 2, a potem sprawdzać wyłącznie liczby nieparzyste.
Liczbę wykonywanych dzielen można ograniczyć jeszcze bardziej. Oto rozkłady liczb 36, 100 i 42 na czynniki:

$36 = 2 \cdot 18$	$100 = 2 \cdot 50$	$42 = 2 \cdot 21$
$36 = 3 \cdot 12$	$100 = 4 \cdot 25$	$42 = 3 \cdot 14$
$36 = 4 \cdot 9$	$100 = 5 \cdot 20$	$42 = 6 \cdot 7$
$36 = 6 \cdot 6$	$100 = 10 \cdot 10$	

Powyższe przykłady pokazują, że gdy liczbę złożoną n przedstawimy w postaci iloczynu dwóch czynników większych od 1, to przynajmniej jeden z tych czynników musi być mniejszy lub równy $\sqrt{n}$.
Dla liczby 36 są to odpowiednio liczby 2, 3, 4 i 6, przy czym $6 = \sqrt{36}$. Podobnie dla pozostałych liczb – mniejszy z dzielników liczby nie jest większy od pierwiastka z tej liczby. Zatem najmniejszy dzielnik właściwy liczby złożonej n musi być mniejszy bądź równy $\sqrt{n}$.

Ćwiczenie 4

Sprawdź za pomocą kalkulatora, czy 401 jest liczbą pierwszą. Ile dzielen było koniecznych, aby to rozstrzygnąć? Zastosuj wskazówkę zapisaną powyżej.

Operacja pierwiastkowania nie jest standardowym działaniem matematycznym w języku Python, dlatego skorzystamy z dodatkowej funkcji matematycznej. Takie funkcje przechowywane są w modułach. **Moduły** to pliki, które mają rozszerzenie *py* i które zawierają definicje wybranych funkcji dotyczących pewnego zagadnienia.
W naszym programie wykorzystamy **funkcję** *sqrt* znajdującą się w **module** *math*. Ten moduł zawiera wiele funkcji, które znasz z lekcji matematyki, m.in. funkcje trygonometryczne.

Warto wiedzieć
Dzielnik właściwy liczby to każdy dzielnik mniejszy od tej liczby.

Aby zaimportować funkcję `sqrt`, na początku dodajemy instrukcję:
`from math import sqrt`.
Poniżej znajduje się tekst programu *Ulepszony test pierwszości*.

Kod źródłowy programu
Ulepszony test pierwszości

```
1. from math import sqrt
2.
3. def CzyPierwsza(n):
4.     if n == 2:
5.         return True
6.     if n % 2 == 0:
7.         return False
8.
9.     pierwiastek = int(sqrt(n))
10.    for i in range(3, pierwiastek+1, 2):
11.        if n % i == 0:
12.            return False
13.    return True
14.
15. liczba = int(input("Podaj liczbę: "))
16.
17. if CzyPierwsza(liczba):
18.     print("Liczba pierwsza")
19. else:
20.     print("Liczba złożona")
```

Dobra rada
Pamiętaj, że funkcja `range` postaci `range(k, n)` generuje indeksy (ciąg liczb całkowitych) od `k` do `n - 1`. Na przykład wywołanie funkcji `range(3, 7)` utworzy ciąg 3, 4, 5, 6.

Ulepszenia wprowadzono w definicji funkcji `CzyPierwsza`. W liniach 4.–7. nadajemy status liczby pierwszej liczbie 2 (linia 5.) oraz status liczby złożonej wszystkim liczbom parzystym większym od 2 (linia 7.).
W linii 9. wprowadzono zmienną `pierwiastek`, której wartość stanowi część całkowita przybliżenia $\sqrt{n}$ otrzymanego dzięki funkcji `sqrt` znajdującej się w module `math` (wskazanym w linii 1.). W pętli z wierszy 10.–13. ograniczamy liczbę wykonań – w tym celu używamy wartości zmiennej `pierwiastek` oraz testujemy tylko liczby nieparzyste (trzecim argumentem funkcji `range` jest krok o długości 2).
Zwróćmy uwagę na to, że drugim argumentem funkcji `range` jest wartość `pierwiastek+1`. Jeśli nie dodamy 1, a `pierwiastek` będzie liczbą nieparzystą, to nie zostanie wzięty pod uwagę przy sprawdzaniu warunku w pętli.

Ćwiczenie 5

a. Przepisz kod źródłowy programu i sprawdź jego działanie.

b. Przygotuj tabelę, w której rozpiszesz kolejne etapy działania funkcji `CzyPierwsza` dla $n = 9$ oraz $n = 11$.

Zadania

- * 1 Napisz program wyznaczający wszystkie dzielniki liczby 80.
- * 2 Napisz program znajdujący najmniejszą i największą trzycyfrową liczbę pierwszą.
- * 3 Napisz funkcję, która będzie badała podzielność podanej liczby jednocześnie przez 3 i 7. W teście użyj pojedynczej instrukcji logicznej (bez operatora `and`).
- * 4 Napisz program, który dla dwóch podanych przez użytkownika liczb wyświetli informację o tym, czy ich suma jest liczbą pierwszą.
- ** 5 Napisz program wyświetlający liczby, których pierwiastki mieszczą się w podanym przez użytkownika przedziale. Na przykład dla przedziału [5; 6] są to liczby: 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36. W kodzie wykorzystaj moduł `math` i funkcję `sqrt`. Sprawdź działanie programu dla przedziału [14; 19].
- ** 6 Stwórz program, który wyznaczy wszystkie trzycyfrowe liczby pierwsze.
- ** 7 Przygotuj w arkuszu kalkulacyjnym skoroszyt, który pozwoli rozstrzygnąć, czy prawdziwe jest następujące twierdzenie: każda liczba złożona mniejsza od 100 dzieli się przynajmniej przez jedną z liczb: 2, 3, 5, 7. Skorzystaj z funkcji `MOD`.
- ** 8 Napisz program wyznaczający wszystkie dzielniki pierwsze liczby 80.
- ** 9 Napisz w języku Python prostą grę, w której dwóch użytkowników podaje po jednej liczbie sześciocyfrowej. Wygrywa ten, kto wskazał liczbę, która ma więcej dzielników właściwych.
- *** 10 Znajdź pary dwucyfrowych i trzycyfrowych liczb bliźniaczych.
Uwaga: Liczby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.
- *** 11 Liczba 362 881 ma tę własność, że przy dzieleniu przez 2, 3, 4, 5, 6, 7, 8, 9 daje resztę 1. Znajdź najmniejszą liczbę o tej własności.
- *** 12 Napisz program, który wyznaczy najmniejszą z trzycyfrowych liczb doskonałych. Liczba doskonała to liczba równa sumie swych dzielników różnych od niej samej, np. $28 = 1 + 2 + 4 + 7 + 14$.
- *** 13 Napisz program, który wypisze wszystkie pary liczb bliźniaczych z przedziału [2; 3000].
Uwaga: Liczby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.

9 Sortowanie liczb i wyszukiwanie

Porządkowanie jest ważną częścią ludzkiej aktywności. Zwykle wysiłek w nie włożony się opłaca, bo później wiele zadań można wykonać szybciej i łatwiej. Jeśli np. szukamy pokoju w hotelu, to docenimy fakt, że pokoje są ponumerowane, a numery często zawierają informację o piętrze. W świecie oprogramowania porządkowanie odbywa się dzięki algorytmom sortowania. W tym temacie przyjrzymy im się bliżej.

Ważne w tym temacie

- Poszukiwanie największej liczby
- Porządkowanie danych
- Algorytmy wyszukiwania sekwencyjnego i binarnego
- Algorytmy sortowania bąbelkowego i przez wstawianie
- Listy jako argumenty funkcji w języku Python

Możemy wyszukiwać dane w zbiorze nieuporządkowanych danych. Jeśli ten zbiór jest niewielki, możemy dość szybko znaleźć interesujący nas element. Ręczne przeszukiwanie większego zbioru wymagałoby od nas dużo więcej czasu. Wtedy warto wykorzystać do pomocy komputer. Znacznie szybciej od człowieka znajdzie on pożądaną przez nas informację nawet w ogromnej ilości danych.

9.1. Szukamy największej liczby

Rozważmy następujący problem: trener skoku w dal mierzy podczas treningu długości kolejnych skoków zawodnika i wpisuje wyniki na bieżąco do programu komputerowego. Chcemy stworzyć program, który na zakończenie treningu wyświetli długość najdłuższego skoku. Zakładamy, że wyniki podawane są w metrach, z dokładnością do 0,01 m, czyli 1 cm.

Nie wiemy, ile skoków lekkoatleta wykona podczas treningu. Nawet jeśli trener zaplanuje jakąś ich liczbę, to i tak niektóre skoki będą prawdopodobnie spalone (nieważne) i nie będą brane pod uwagę w obliczeniach.



Rys. 9.1. Skok w dal

Warto wiedzieć

Określenie jednostek, w których są wyrażone różne wielkości, jest dla programisty bardzo ważne, ponieważ musi on odpowiednio wybrać typ danych. Na przykład jeśli dokładność pomiaru skoku w dal wynosi 1 cm, to wyniki podawane w centymetrach będą liczbami całkowitymi. Jeśli jednak wyniki podajemy w metrach, mogą być ułamkami.

Program komputerowy trzeba więc napisać tak, aby trener mógł poinformować komputer o tym, że trening został zakończony. Jak to zrobić? Jednym ze sposobów może być wpisanie przez trenera liczby 0, która zakończy ciąg podawanych liczb. Wtedy program komputerowy powinien wypisać długość najdłuższego skoku.

Specyfikacja

Dane: ciąg liczb nieujemnych, zapisanych z dokładnością do 0,01, zakończony liczbą 0.

Wynik: największa z podanych liczb.

Aby wykonać to zadanie, program musi porównać długości skoków. Komputer nie potrafi jednak jednocześnie porównywać wielu liczb, nawet gdyby miał je wszystkie zapisane w pamięci. Pracuje on sekwencyjnie i nie może jak człowiek objąć wzrokiem kilku liczb, aby wskazać największą. W jednej operacji może brać pod uwagę tylko dwie liczby. Pozostałe są niedostępne.

Na poniższym schemacie zaprezentowano, jaką strategię wyboru największej spośród czterech podanych liczb stosuje człowiek, a jaką – komputer.

Jak widzi komputer?

Zadanie:

Która z podanych liczb jest największa?

6,40 6,53 6,34 6,55



Człowiek

Przy małej liczbie danych człowiekowi wystarczy jedno spojrzenie, aby wskazać największą liczbę:

6,40 6,53 6,34 6,55



Komputer

Komputer, nawet przy małej liczbie danych, etapami porównuje kolejne pary liczb i wskazuje większą liczbę z pary:

Etap 1	6,40	6,53		
Etap 2		6,53	6,34	
Etap 3		6,53		6,55

Dobra rada

Ciąg potraktuj jako układ liczb, w którym występują one w określonej kolejności.

Warto wiedzieć

W żargonie programistów specjalną wartość (np. 0), która wskazuje koniec ciągu wczytywanych do programu wartości, nazywa się wartownikiem (ang. *sentinel*). Wartość ta musi być unikatowa, aby nie można było jej pomylić z wartościami podanymi przez użytkownika programu.

Aby program mógł wyznaczyć najdłuższy z podanych skoków, będziemy porównywać ich długości parami: porównamy pierwszy skok z drugim, dłuższy z nich porównamy z trzecim skokiem, dłuższy z nich z czwartym itd. Te operacje program będzie wykonywał do momentu, aż użytkownik wpisze liczbę 0.

Dzięki takiemu rozwiązaniu program będzie musiał zapamiętywać tylko dwie wartości: aktualnie podaną przez użytkownika i największą z dotychczas podanych. W naszej sytuacji pamiętanie pozostałych wartości nie jest potrzebne.

Lista kroków algorytmu wyznaczania największej liczby z ciągu liczb mogłaby wyglądać następująco:

- 1 Jako wartość aktualnie największą (maksimum) przyjmij 0 i wczytaj pierwszą liczbę.
- 2 Dopóki ostatnia wczytana liczba jest różna od 0, wykonuj krok 3 i krok 4.
- 3 Jeśli ostatnia wczytana liczba jest większa od maksimum, to podstaw ją za maksimum.
- 4 Wczytaj kolejną liczbę z klawiatury.
- 5 Wypisz maksimum na ekranie.

Przeanalizujemy teraz działanie algorytmu na przykładowych danych. Wygodnie będzie zrobić to w tabeli.

Wartość aktualnie największa (maksimum)	Wczytana liczba	Czy ostatnia wczytana liczba jest różna od 0?	Czy ostatnia wczytana liczba jest większa od aktualnego maksimum?
0	6,20	tak	tak
6,20	6,23	tak	tak
6,23	6,34	tak	tak
6,34	6,30	tak	nie
6,34	6,41	tak	tak
6,41	6,14	tak	nie
6,41	6,34	tak	nie
6,41	6,27	tak	nie
6,41	6,10	tak	nie
6,41	0	nie	–

Tabela 9.1. Działanie algorytmu wyznaczania największej liczby

Warto wiedzieć
Procesor porównujący ze sobą dwie liczby używa sprzętowej instrukcji porównywania. Jej działanie polega na sprawdzeniu, czy różnica porównywanych liczb jest dodatnia, ujemna czy równa 0.

Dobra rada
Jeżeli trudno ci zrozumieć, jak działa algorytm, możesz utworzyć tabelę zawierającą pośrednie wartości zmiennych i warunków logicznych.

Tabelę uzupełniamy wierszami od lewej do prawej, według kolejnych kroków algorytmu. Kolorem zielonym zaznaczono pewien etap analizy działania algorytmu: wczytano w tym momencie liczbę 6,41. Kolejnymi operacjami będą: sprawdzenie, czy liczba 6,41 jest różna od 0 (krok 2), a następnie – czy jest większa od ostatniej największej wartości równej 6,34 (krok 3). Ponieważ oba warunki są spełnione, liczba 6,41 staje się aktualnie największą. W kolejnych iteracjach podawane są liczby od niej mniejsze, dlatego w momencie podania liczby 0 liczba aktualnie największa staje się największą spośród wszystkich podanych liczb.

Ćwiczenie 1
Zastanówcie się w grupie, czy jeśli na liście kroków ze s. 130 zamienimy miejscami kroki 3 i 4, to lista nadal będzie przedstawiała poprawny algorytm. W jaki sposób można to sprawdzić?

Zauważ, że program wyznaczający największą liczbę z ciągu powinien cyklicznie:

- 1. sprawdzać, czy liczba wprowadzona przez użytkownika jest liczbą 0, której podanie kończy działanie programu,
- 2. sprawdzać, czy wprowadzona liczba jest większa od aktualnie największej,
- 3. wczytywać kolejną wprowadzoną liczbę.

Możemy więc do rozwiązania tego zadania wykorzystać pętlę while, a kod źródłowy programu *Najdłuższy skok* może wyglądać następująco:

```
1. maks = 0
2.
3. print("Podaj kolejne długości skoków:")
4. a = float(input())
5.
6. while a != 0:
7.     if a > maks:
8.         maks = a
9.     a = float(input())
10.
11. print("\nNajdłuższy skok:", maks, "m")
```

W wierszach 4. i 9. podczas wczytywania długości skoków zastosowaliśmy funkcję float, która pozwala liczby wpisane z klawiatury przechować jako liczby wymierne. W zmiennej a będzie przechowywana liczba jako ostatnia wczytana z klawiatury, a w zmiennej maks – aktualnie największa z wczytanych liczb. W wierszu 1. przypisujemy tej zmiennej wartość 0.

Pętla while
→ s. 101

Kod źródłowy programu Najdłuższy skok

Warto wiedzieć
W języku Python zmienna musi mieć jawnie przypisaną wartość, zanim będzie można jej użyć, np. w warunku pętli. Jeśli programista nie nada wartości początkowej zmiennej, to po uruchomieniu programu jego działanie zostanie przerwane i pojawi się komunikat o błędzie.

Funkcja float

W wierszach 6.–9. pojawia się pętla `while`, która najpierw sprawdza, czy wartość zmiennej `a` jest różna od `0`. Jeżeli tak, to program wykonuje grupę instrukcji zapisanych w wierszach 7.–9. Są one w kodzie programu odsunięte od marginesu tak, aby interpreter mógł rozpoznać, że należą do pętli. Instrukcje są wykonywane tyle razy, ile razy użytkownik podał liczbę różną od zera.

W wierszu 7. sprawdzamy, czy podana przez użytkownika liczba jest większa od wartości zmiennej `maks`. Jeśli tak, to ta liczba jest przypisywana do zmiennej `maks`. Po zakończeniu wprowadzania liczb (czyli po wpisaniu przez użytkownika liczby 0) aktualnie największa liczba staje się liczbą największą dla tego zestawu danych. Warto zauważyć, że instrukcja przypisania wartości `a` do zmiennej `maks` (wiersz 8.) jest zapisana z dodatkowym odstępem po to, aby poinformować interpreter, że wiersze 7. i 8. stanowią zamkniętą całość. W wierszu 11. użyto zapisu `\n`, który odpowiada za przeniesienie kursora do kolejnego wiersza.

Okno programu *Najdłuższy skok* przedstawiono na rysunku 9.2. Zwróć uwagę na to, że w programie liczby ułamkowe podaje się z kropką zamiast przecinka.

Warto wiedzieć

Od lewego ukośnika zapisuje się stałe znakowe znaków niegraficznych kodu ASCII (np. `\n` dla nowego wiersza, `\t` dla tabulacji poziomej) lub znaków specjalnych (np. `\"` dla znaku `"`, gdy chcemy go wyświetlić na ekranie).

Dobra rada

Jeżeli używasz w pętli `while` warunku porównującego wartości typu `float` (liczby wymierne), staraj się stosować operatory porównania `<`, `>` i `!=` zamiast `<=`, `>=` oraz `==`. Pozwoli to uniknąć niektórych błędów wynikających z dokładności przybliżeń wartości zmiennych.

Warto wiedzieć

Zapisywanie liczb ułamkowych z kropką zamiast przecinka jest zwyczajem anglosaskim. To dlatego na klawiaturze numerycznej znajduje się właśnie znak kropki.



Rys. 9.2. Wywołanie programu *Najdłuższy skok*

Ćwiczenie 2

Utwórz plik `najdluzszy_skok.py` i zapisz w nim kod źródłowy programu *Najdłuższy skok*. Sprawdź kod i uruchom program kilkakrotnie dla różnych danych wpisanych z klawiatury.

9.2. Do czego służy sortowanie danych?

Sortowanie danych (ang. *sorting*), zwane też **porządkowaniem danych**, to ustawianie danych w zadanej kolejności, według określonego kryterium. **Sortowanie (porządkowanie) danych**

Zwykle gdy mówimy o sortowaniu liczb, mamy na myśli uporządkowanie ich w kolejności malejącej albo rosnącej. W rzeczywistości najczęściej jest to kolejność nierosnąca lub niemalejąca, ponieważ sortowane wartości mogą się powtarzać. Wyrazy z reguły ustawiamy w kolejności alfabetycznej (od A do Z) albo odwróconej kolejności alfabetycznej (od Z do A). Z kolei daty i godziny porządkuje się zwykle chronologicznie, czyli od najstarszych do najnowszych (ewentualnie w chronologii odwróconej – od najnowszych do najstarszych). Przykłady posortowanych danych widoczne są na rysunku 9.3.

2003	butelka	01.01.2001
2147	długopis	15.03.2005
2290	klucze	22.07.2010
2435	kubek	09.09.2012
2589	kurtka	30.04.2015
2741	laptop	18.08.2017
2896	ładowarka	05.11.2018
3050	notatnik	14.02.2020
3204	okulary	23.06.2021
3358	parasol	11.10.2022
3512	plecak	07.05.2023

Rys. 9.3. Posortowane liczby, wyrazy i daty

Uporządkowane informacje, zwłaszcza obszerne zbiory danych, o wiele łatwiej czytać, analizować oraz przeszukiwać.

Rysunek 9.4 przedstawia przykładową tabliczkę przystankową. Zauważ, że dane są uporządkowane na dwóch poziomach: kierunki – alfabetycznie, a godziny dla każdego kierunku – chronologicznie.

LGOTA WIELKA				
ROZKŁAD ODJAZDÓW				
Kierunek	Przez	Godziny odjazdu		
BOGUMIŁOWICE	SULMIERZYCE	14:50 S		
ŁÓDŹ	BEŁCHATÓW	6:58 dnw	12:08 dnw	17:03 dnw
PAJĘCZNO	SULMIERZYCE	09:20 Dm		
RADOMSKO		05:11 6s	07:01 S	07:09 S
		07:11 S	08:15 S	08:15 Dm
		12:05 Dm	14:35 dnw	18:05 dnw
		23:18 dnw		

Rys. 9.4. Przykład tabliczki przystankowej

PORZĄDEK WOKÓŁ NAS

Zwykle nie zwracamy uwagi na to, jak wiele obiektów wokół nas jest uporządkowanych. Ważne jest jednak nie tylko samo uporządkowanie, lecz także jego funkcjonalność, która znacznie ułatwia nam codzienne życie.

Kody pocztowe

Wprowadzenie kodów pocztowych znacznie usprawniło sortowanie i skróciło czas dostarczania listów. Wcześniej pracownicy poczty musieli sami odszyfrowywać nazwy miejscowości i sprawdzać ich lokalizację.

**Wąbrzeźno**

87

200

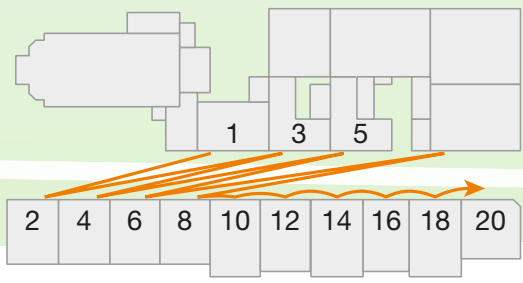
Okręg i strefa kodowa

Sektor kodowy i placówka pocztowa



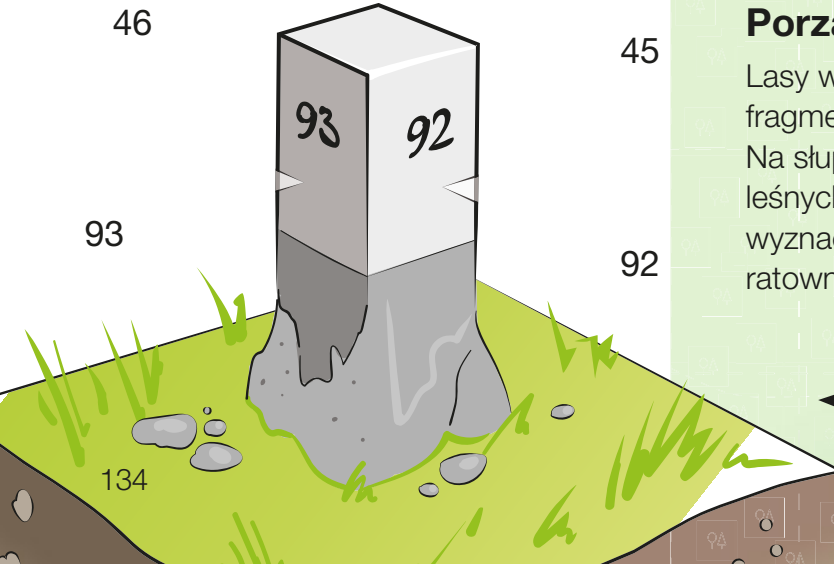
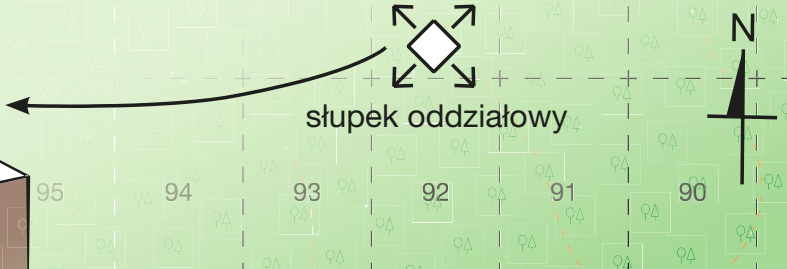
Numeracja domów

Domy znajdujące się w centrach miast są numerowane w uporządkowany sposób. Numery domów położonych wzdłuż głównych ulic rosną w kierunku od centrum miejscowości ku jej krańcom: po lewej stronie są numery nieparzyste, po prawej – parzyste.



Porządek w lesie

Lasy w Polsce są podzielone na oddziały (prostokątne fragmenty), które są identyfikowane przez słupki oddziałowe. Na słupkach znajdują się liczby oznaczające numery oddziałów leśnych. Słupki oddziałowe pomagają określić położenie, wyznaczyć kierunek oraz ułatwiają prowadzenie akcji ratowniczej.



Ćwiczenie 3

Z jakimi uporządkowanymi zbiorami danych masz do czynienia na co dzień? Wymień co najmniej pięć konkretnych przykładów.

Szukanie hasła w słowniku lub skorowidzu

W encyklopediach, słownikach i leksykonach obowiązuje porządek alfabetyczny. Gdy korzystasz z tradycyjnego słownika, np. polsko-angielskiego, zwykle nie szukasz na chybił trafił, lecz intuicyjnie stosujesz bardzo rozsądną strategię poszukiwania słowa. Podobnie postępujesz podczas szukania hasła w indeksie (skorowidzu) umieszczonym zwykle na końcu książki (rys. 9.5).

A	Grafika informacyjna (infografika) 74
Adres IP (adres logiczny) 30	— rastrowa (bitmapowa) 46
— MAC (adres fizyczny, adres sprzętowy) 27	— wektorowa 61
— opisowy (mnemoniczny) 34	
— URL 38	H
Atak socjotechniczny 127	Hejt 123
	Host 30
B	
Bezprzewodowa sieć lokalna (WLAN) 25	I
Big data 95	Identyfikator 108
Botnet 128	Importowanie danych 188
Brama 24	Indeksowanie plików 19
	Influencer 124

Rys. 9.5. Fragment indeksu podręcznika *Nowa Informatyka na czasie 1*

Jeśli chcesz znaleźć hasło *Adres IP*, to prawdopodobnie szukasz go wśród początkowych haseł, a jeśli *Influencer* – mniej więcej w środku. Następnie szukasz wyrazów zawierających dwie pierwsze litery – w naszej sytuacji będą to odpowiednio *Ad* i *In*. Za każdym razem zawężasz obszar poszukiwań, pomijając strony, na których dane hasło na pewno nie wystąpi. Posortowanie haseł przez twórców słownika pozwala więc znacznie skrócić czas przeszukiwania.

Warto wiedzieć

Porządek alfabetyczny nazywa się też porządkiem leksykograficznym.

Warto wiedzieć

Metodę wyszukiwania, którą stosujemy intuicyjnie podczas korzystania ze słownika, nazywa się wyszukiwaniem interpolacyjnym.

A to ciekawe

Algorytmy wyszukiwania

Znalezienie w sieci informacji, których potrzebujemy, byłoby praktycznie niemożliwe bez ich uporządkowania. Systemy rankingowe wyszukiwarek internetowych przeglądają indeksy wyszukiwania, stosując wiele różnych algorytmów. Analizują takie aspekty, jak treść zapytania, przydatność stron, wiarygodność źródła, czas zamieszczenia informacji, lokalizacja użytkownika. Gdyby wykorzystywane algorytmy nie działały bardzo szybko dla dużych zestawów danych, na wyniki dla pojedynczego zapytania trzeba byłoby czekać nawet kilkanaście minut lub dłużej.



Klasyfikacja zawodników w pięcioboju nowoczesnym

Przyjrzyjmy się bliżej zasadom ustalania kolejności w pięcioboju nowoczesnym. Składa się on z szermierki, pływania, jazdy konnej oraz konkurencji łączonej – biegu na przełaj i strzelania z pistoletu. Wyniki w poszczególnych konkurencjach są przeliczane na punkty wielobojo-we. W tabeli 9.2 przedstawiono wyniki pierwszych dziesięciu zawod-niczek w klasyfikacji końcowej Letnich Igrzysk Olimpijskich w Paryżu w 2024 r. W rywalizacji brało udział 36 zawodniczek.

Lista zawodniczek			Szermierka		Pływanie			Jazda konna		Bieg ze strzelaniem			Suma
			punkty	miejsce	czas	miejsce	punkty	miejsce	punkty	czas	miejsce	punkty	
1.	Michelle Gulyas	HUN	245	2	2:12.44	4	286	1	300	11:10.01	7	630	1461
2.	Élodie Clouvel	FRA	264	1	2:11.64	3	287	9	293	11:32.35	14	608	1452
3.	Seungmin Seol	KOR	225	7	2:11.47	2	288	2	300	11:12.87	8	628	1441
4.	Blanka Guzi	HUN	200	17	2:16.25	7	278	8	300	10:45.48	1	655	1433
5.	Elena Micheli	ITA	232	3	2:12.47	5	286	14	293	11:27.86	12	613	1424
6.	Ilke Ozyuksel	TUR	205	15	2:15.48	6	280	12	293	10:58.20	3	642	1420
7.	Gintare Venckauskaite	LTU	205	16	2:18.16	12	274	5	300	11:00.31	4	640	1419
8.	Sunwoo Kim	KOR	222	8	2:17.67	11	275	16	286	11:13.92	9	627	1410
9.	Kerenza Bryson	GBR	230	4	2:21.77	15	267	15	286	11:19.12	10	621	1404
10.	Lucie Hlavackova	CZE	209	13	2:20.54	13	269	11	293	11:08.44	6	632	1403

Tabela 9.2. Wyniki pięcioboju nowoczesnego kobiet – igrzyska w Paryżu w 2024 r.

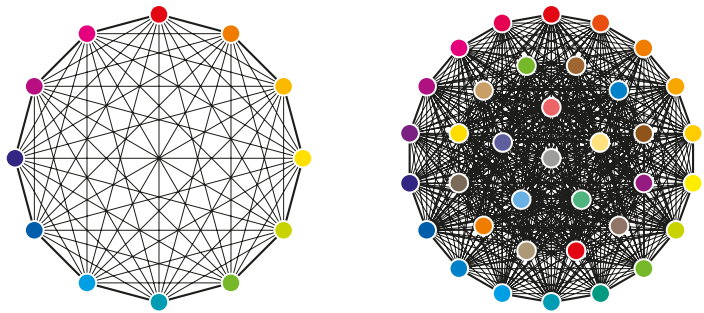
Ćwiczenie 4

a. Według jakiego kryterium uporządkowano zawodniczki w tabeli 9.2?

b. Czy klasyfikacje dla każdej pojedynczej konkurencji są tworzone na podstawie wyników uporządkowanych malejąco?

Ustalenie kolejności zawodników podczas zawodów w wieloboju jest szczególnie ważne przed rozegranie ostatniej konkurencji (strzelanie i bieg). Różnice punktowe w klasyfikacji są bowiem przeliczane na sekundy. Lider startuje jako pierwszy, a kolejni z opóźnieniem wyliczo-nym na podstawie dotychczasowych wyników.

Systemy wyłaniania zwycięzców wcześniejszych konkurencji muszą być sprawiedliwe. Na przykład w szermierce stosuje się system „każdy z każdym”, zwany systemem kołowym. Przy dużej liczbie rywalizują-cych jest on jednak bardzo czasochłonny. Dla 36 zawodniczek (każda walczy 35 razy) należy przeprowadzić aż $\frac{1}{2} \cdot 36 \cdot 35 = 630$ pojedynków. Dla porównania – przy 12 zawodniczkach odbyłoby się 66 walk. Tę róż-nicę pokazuje rysunek 9.6.



Rys. 9.6. Ilustracja liczby pojedynków w rywalizacji „każdy z każdym” dla 12 i 36 zawodniczek

Sortowanie danych w komputerze

Sortowanie zbiorów danych jest jedną z operacji najczęściej wykony-wanych przez komputery podczas przetwarzania informacji. Oto kilka przykładów zastosowań sortowania, od prostszych do mniej oczywi-stych:

- ▶ wyświetlanie listy plików w katalogu (folderze) na dysku – przed wyświetleniem zawartości katalogu system operacyjny sortuje pliki zgodnie z określonym kryterium,
- ▶ przeglądanie danych z arkusza kalkulacyjnego lub bazy danych – często stosujemy sortowanie, aby ułatwić czytanie danych zawartych w tabelach (nadać im odpowiednie znaczenie),
- ▶ wyszukiwarki internetowe – zasoby zgromadzone na serwerach wy-szukiwarek internetowych muszą być zindeksowane, aby wyszuki-warka mogła zwrócić wynik możliwie szybko,
- ▶ grafika komputerowa – obiekty w grze komputerowej lub np. w sy-mulatorze lotu są ułożone na warstwach, więc aby program mógł wy-sświetlić obiekty na ekranie w dobrej kolejności, może wymagać ich wcześniejszego posortowania (względem odległości od obserwatora).

9.3. Sortowanie a wyszukiwanie

Uporządkowanie danych ma zasadnicze znaczenie podczas prze-szukiwania ich zbioru. Prześledźmy to na przykładzie wyszukiwania w pamięci komputera.

Przyjmijmy, że po uruchomieniu jakiegoś programu w pamięci komputera mamy zapisaną listę jako uporządkowany ciąg komórek pamięci RAM, zaadresowanych kolejnymi liczbami całkowitymi.

Warto wiedzieć

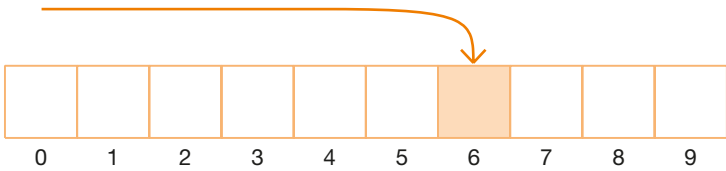
W języku angielskim system rywalizacji „każdy z każdym” nazywa się *round-robin tournament* (dosł. turniej karuzelowy).

Warto wiedzieć

Przykłady problemów, do których rozwiązania jest potrzebne sortowanie danych:

- ▶ usuwanie duplikatów,
- ▶ sprawdzanie, czy wszystkie elementy zbioru danych są różne,
- ▶ znajdowanie dwóch wartości o najmniejszej różnicy,
- ▶ wyznaczanie mediany, dominanty i innych statystyk,
- ▶ odtwarzanie pierwotnego porządku elementów zbioru danych.

Jeśli procesor zna położenie danych w pamięci, to odwołuje się do nich bezpośrednio, poprzez adres komórki, w której się znajdują (rys. 9.7).



Rys. 9.7. Dostęp bezpośredni

Warto wiedzieć

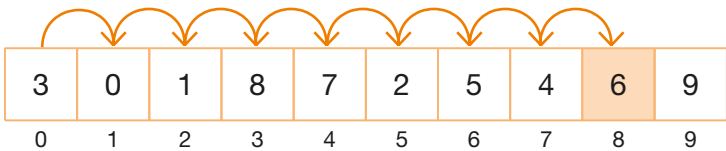
Czas dostępu do danych zależy nie tylko od ich uporządkowania, lecz także od rodzaju pamięci. Różny czas osiągają rejestry procesora, pamięć podręczna, pamięć RAM czy pamięć zewnętrzna (np. flash, dysk twardy).

Aby odczytać dane zapisane w siódmej komórce (indeks 6) fragmentu pamięci, procesor nie odczytuje wszystkich danych w kolejności, w której zostały zapisane, lecz pobiera dane z właściwej komórki (rys. 9.7).

Współczesne algorytmy komputerowe często wymagają, aby czas dostępu do danych z pamięci operacyjnej był jak najkrótszy. Wtedy kluczowe znaczenie – oprócz aspektu technicznego (budowy pamięci) – ma właśnie uporządkowanie danych. Jego brak znacznie spowalnia wyszukiwanie informacji, nawet jeśli korzysta się z szybkich procesorów.

Wyszukiwanie sekwencyjne

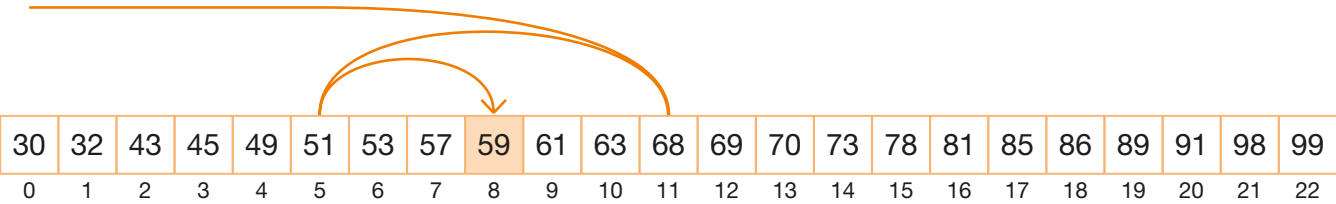
Kiedy szukamy informacji w zbiorach danych nieuporządkowanych, nie mamy wielkiego wyboru: albo zastosujemy metodę „na chybił trafił”, albo algorytm **wyszukiwania sekwencyjnego** (ang. *sequential search*), tj. przeglądanie zbioru danych w kolejności, w której są zapisane (np. według adresów w pamięci). Rysunek 9.8 pokazuje sekwencyjne wyszukiwanie liczby 6 w zbiorze nieuporządkowanym.



Rys. 9.8. Wyszukiwanie sekwencyjne

Wyszukiwanie binarne

Wyszukiwanie informacji można znacznie przyspieszyć, gdy zbiór danych jest uporządkowany. Stosujemy wówczas strategię połowienia zbioru, czyli zmniejszania liczby jego elementów o połowę w każdej kolejnej próbie. Tę strategię nazywamy **wyszukiwaniem binarnym (połówkowym)**. Przykładowy algorytm zastosowany do poszukiwania liczby 59 w zbiorze 23 elementów przedstawiono na rysunku 9.9.



Rys. 9.9. Wyszukiwanie binarne

Wyszukiwanie liczby 59 zaczęło się od sprawdzenia środkowej komórki obszaru pamięci. Jest to komórka o indeksie 11, przechowująca liczbę 68, która jest większa od 59. Ponieważ wiemy, że dane w pamięci są uporządkowane, mamy pewność, że przeszukiwanie komórek o indeksach większych od 11 można pominąć. W drugiej próbie program wyszukujący powinien sprawdzić komórkę o indeksie 5. Znajduje się w niej liczba 51, dlatego wartości należy szukać w komórkach o indeksach od 6 do 10. W kolejnym kroku sprawdzamy komórkę o adresie 8 – znajdziemy w niej poszukiwaną liczbę.

Ćwiczenie 5

Ile sprawdzeń (porównań dwóch liczb) wystarczy, aby rozstrzygnąć, czy w ciągu komórek z rysunku 9.9 znajduje się liczba 80? Zapisz indeksy oraz wartości komórek, które kolejno będą sprawdzane.

9.4. Algorytmy sortowania

Spójrzmy na porządkowanie jako na problem algorytmiczny i programistyczny. W ostatnich kilkudziesięciu latach odkryto i opisano co najmniej kilkanaście użytecznych algorytmów sortowania danych. Okazuje się, że nie ma algorytmu idealnego o ogólnym przeznaczeniu, najszybszego w każdej sytuacji. Najbardziej uniwersalne są metody hybrydowe, łączące zalety dwóch lub trzech algorytmów sortowania, stosowanych na różnych etapach procesu porządkowania.

Ćwiczenie 6

Przygotuj 8 kartek wielkości połowy kartki z zeszytu. Na każdej z nich zapisz imię oraz datę imienin (dzień i miesiąc). Imiona nie mogą się powtarzać. Następnie odwróć kartki i ułóż je w losowej kolejności.

- a. Zaproponuj metodę sortowania informacji na kartkach (alfabetycznie lub chronologicznie). Przyjmij, że kartki trzeba porównywać parami i nie wolno odkryć więcej niż dwóch naraz.
- b. W grupie dwu- lub trzyosobowej przedyskutujcie zastosowane wcześniej metody porządkowania kartek. Porównajcie czas sortowania 8 i 16 kartek wybraną przez was metodą. Ile razy dłużej trwa porządkowanie 16 kartek?

Programy komputerowe mogą porządkować różne typy danych: liczby, zbiory liter i słów, daty i inne. Wszystkie te wartości są zapisywane cyfrowo w pamięci komputera. Dla uproszczenia ograniczymy się więc do porządkowania list liczb całkowitych. Dodatkowo przyjmijmy, że będziemy sortować w porządku niemalejącym.

Warto wiedzieć

Porównanie dwóch liczb jest jedną z podstawowych operacji wykonywanych przez procesor komputerowy. Każdy język programowania musi obsługiwać tę operację.

Warto wiedzieć

Algorytmy sortowania można podzielić na dwie grupy: te, które działają na zasadzie porównywania elementów zbioru parami (np. sortowanie bąbelkowe), oraz nieposługujące się tą operacją (np. sortowanie przez zliczanie).

Oto specyfikacja naszego problemu sortowania:

Specyfikacja

Dane: liczba naturalna n i n -elementowa lista A liczb całkowitych.
Wynik: lista A po uporządkowaniu: $A[0] \leq A[1] \leq \dots \leq A[n-1]$.

9.5. Sortowanie bąbelkowe

Sortowanie bąbelkowe • Jednym z algorytmów porządkowania jest **sortowanie bąbelkowe** (ang. *bubble sort*). W tym algorytmie na każdym etapie porównuje się parę elementów sąsiednich, zaczynając od pierwszej pary na liście. Jeżeli podczas porównywania elementy są w niewłaściwej kolejności, to zamienia się je miejscami.

Operacja prostej zamiany

Operacja prostej zamiany • **prostą zamianę**, rozumianą jako zamianę miejscami sąsiednich elementów na liście.

Podobną operację stosuje się w oprogramowaniu niektórych telewizorów. Pozwala ono za pomocą pilota uszeregować listę kanałów. Są trzy możliwości: zamiana kanałów miejscami (•), przejście w dół listy (▽) i przejście w górę listy (△). Rysunek 9.10 przedstawia zastosowanie operacji prostej zamiany podczas ustawiania listy kanałów.

TV 3	•	TV 0	▽	TV 0		TV 0		TV 0		TV 0
TV 0		TV 3	•	TV 1	▽	TV 1		TV 1		TV 1
TV 1		TV 1		TV 3	▽	TV 3		TV 3	•	TV 2
TV 6		TV 6		TV 6	▽	TV 6	•	TV 2	△	TV 3
TV 7		TV 7		TV 7	•	TV 2	△	TV 6		TV 6
TV 2		TV 2		TV 2		TV 7		TV 7		TV 7
TV 5		TV 5		TV 5		TV 5		TV 5		TV 5
TV 4		TV 4		TV 4		TV 4		TV 4		TV 4

Rys. 9.10. Pierwsze kroki porządkowania kanałów telewizyjnych przez prostą zamianę

A to ciekawe

Sortowanie w tańcu

Na kanale AlgoRythmics w serwisie YouTube można znaleźć filmy objaśniające działanie różnych algorytmów sortowania. Porządkowanie liczb jest zaprezentowane w tańcu przez członków węgierskiego zespołu folkowego Maros Művészegyüttes. Można tam zobaczyć również m.in. wyszukiwanie binarne.



ALGORYTM SORTOWANIA BĄBELKOWEGO

Porządkowanie elementów tą metodą pokażemy na przykładzie pięcioelementowej listy, w której na kolejnych pozycjach znajdują się liczby: 6, 4, 9, 1, 5. Elementy uporządkujemy niemalejąco. Pierwszy etap ilustruje, jak umieścić właściwy element na ostatniej pozycji. W kolejnych etapach postępujemy analogicznie.

Etap 1

- Porównujemy pierwszy element z drugim. $6 > 4$, więc zamieniamy liczby miejscami.
- Porównujemy drugi element z trzecim. $6 < 9$, więc nie zamieniamy liczb miejscami.
- Porównujemy trzeci element z czwartym. $9 > 1$, więc zamieniamy liczby miejscami.
- Porównujemy czwarty element z piątym. $9 > 5$, więc zamieniamy liczby miejscami.
- W wyniku czterech porównań największy element listy (liczba 9) znalazł się na ostatniej pozycji.

6 > 4

9

1

5

4

6 < 9

1

5

4

6

9 > 1

5

4

6

1

9 > 5

4

6

1

5

9

W drugim etapie umieścimy właściwy element na przedostatniej pozycji. Przy porównaniach pomijamy ostatni element listy, ponieważ jest już na właściwym miejscu. W trzecim etapie przy porównaniach pomijamy dwa ostatnie elementy listy, w czwartym – trzy ostatnie itd. Poniżej przedstawione są wyniki otrzymane w kolejnych etapach.

Etap 2

Porównujemy pary liczb: 4 i 6, 6 i 1 oraz 6 i 5. Liczba 6 znalazła się na właściwej, przedostatniej pozycji.

4

1

5

6

9

Etap 3

Porównujemy pary liczb 4 i 1 oraz 4 i 5. W wyniku tych operacji liczba 5 jest na właściwej pozycji.

1

4

5

6

9

Etap 4

Porównujemy parę liczb 1 i 4. Znalezienie drugiego elementu ciągu wskazuje jednocześnie najmniejszy element ciągu.

1

4

5

6

9

Wynikiem działania algorytmu jest lista:

1	4	5	6	9
---	---	---	---	---

Ćwiczenie 7

Dana jest lista $A = [3, 0, 1, 7, 5]$. Sprawdź, ile prostych zamian potrzeba, by uporządkować ją algorytmem sortowania bąbelkowego:

- a. rosnąco,
- b. malejąco.

Algorytm porządkowania metodą bąbelkową można zapisać jako

Pętle zagnieżdżone

pętle zagnieżdżone (ang. *nested loops*), czyli pętle umieszczone wewnątrz innych pętli. To oznacza, że na jeden etap algorytmu (cykl pętli zewnętrznej) może przypadać wiele cykli pętli wewnętrznej.

Każdy etap algorytmu kończy się ustawieniem na docelowym miejscu elementu, który w analizowanym fragmencie był tym o największej wartości. Liczba porównań w kolejnym etapie zmniejsza się więc o 1.

Oto zapis algorytmu w postaci listy kroków:

- 1 Wczytaj listę liczb A.
- 2 Dla wartości i równych kolejno 1, 2, ..., n-1 powtarzaj kroki 3 oraz 4.
- 3 Dla wartości j równych kolejno 0, 1, ..., n-i-1 powtarzaj krok 4.
- 4 Jeśli $A[j] > A[j+1]$, to zamień te elementy miejscami.

Liczba powtórzeń zewnętrznej pętli wynosi n-1. Liczba powtórzeń wewnętrznej pętli (czyli pętli sortującej) jest zależna od wartości zmiennej i, sterującej zewnętrzną pętlą.

Kod źródłowy funkcji o nazwie Babelkowe w programie Sortowanie bąbelkowe może wyglądać następująco:

```
1. def Babelkowe(A):
2.     n = len(A)
3.     for i in range(1, n):
4.         for j in range(0, n-i):
5.             if A[j] > A[j+1]:
6.                 A[j], A[j+1] = A[j+1], A[j]
```

Zauważ, że funkcja Babelkowe jako parametr przyjmuje listę. W ten sposób przekazuje się do funkcji informację o adresie listy w pamięci. Wszystkie operacje są wykonywane bezpośrednio na liście, dla której funkcja została wywołana.

W linii 6. elementy listy są zamieniane miejscami. Zapis umieszczony po prawej stronie znaku = oznacza utworzenie dwuelementowej **krotki** (ang. *tuple*), czyli odpowiednika listy, której nie można modyfikować. Elementowi $A[j+1]$ zostaje przypisana kopia wartości $A[j]$, a $A[j]$ – kopia wartości $A[j+1]$.

Dobra rada

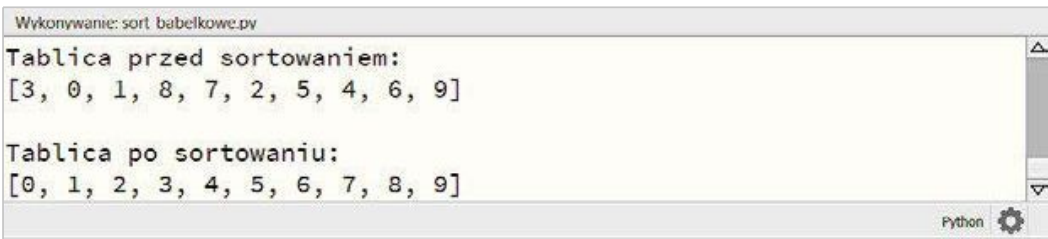
Aby uporządkować elementy nierosnąco, zmień w instrukcji warunkowej znak > na <.

Kod źródłowy funkcji Babelkowe w programie Sortowanie bąbelkowe

Warto wiedzieć

W nagłówku definiowanej funkcji, np. `def Funkcja(A)`, nazwa listy nie musi być identyczna z nazwą listy, dla której wywołamy funkcję. Funkcję można wywołać np. dla listy B i zapisać `Funkcja(B)`.

Rysunek 9.11 przedstawia przykładowe wywołanie programu Sortowanie bąbelkowe.



Rys. 9.11. Przykładowe wywołanie programu Sortowanie bąbelkowe

Ćwiczenie 8

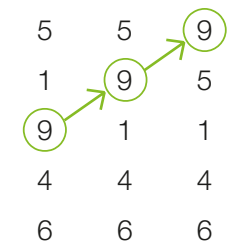
Napisz kod źródłowy programu o nazwie Sortowanie bąbelkowe, w którym wykorzystasz funkcję Babelkowe, a dane do sortowania wprowadzisz bezpośrednio w kodzie źródłowym. Sprawdź działanie programu.

Zapamiętaj

Kluczową operacją w wielu algorytmach sortowania danych jest porównywanie danych parami, by ustalić ich porządek.

Warto wiedzieć

Nazwa sortowanie bąbelkowe powstała przez analogię do zjawiska wypływania na powierzchnię wody bąbelków powietrza: ten o największej objętości wypływa jako pierwszy.



9.6. Sortowanie przez wstawianie

Omówimy teraz kolejny algorytm porządkowania danych – **sortowanie przez wstawianie** (ang. *insertion sort*). Ta metoda polega na przeglądaniu kolejnych elementów listy od lewej i przenoszeniu ich, w razie potrzeby, w odpowiednie miejsce już uporządkowanego fragmentu listy po lewej stronie. Działanie algorytmu przedstawiono na s. 144.

Sortowanie przez wstawianie przypomina układanie kart przez graczy w gry karciane: karta, którą podnosimy ze stołu, jest wstawiana we właściwe miejsce do kart już uporządkowanych (rys. 9.12).



Rys. 9.12. Układanie kart metodą przez wstawianie

Sortowanie przez wstawianie

Warto wiedzieć

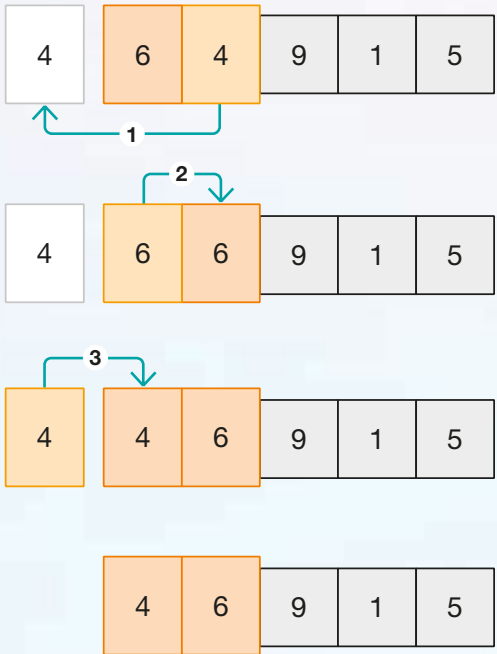
Doświadczeni gracze w pokera lub brydża nie sortują swoich kart. W ten sposób bowiem dostarczyliby przeciwnikom informacji o tym, jakie karty mają w ręku.

ALGORYTM SORTOWANIA PRZEZ WSTAWIANIE

Porządkowanie elementów tą metodą pokażemy na przykładzie pięcioelementowej listy, w której na kolejnych pozycjach znajdują się liczby: 6, 4, 9, 1, 5. Elementy uporządkujemy niemalejąco. Pierwszy etap ilustruje, jak uporządkować niemalejąco dwa elementy listy. Liczby przy strzałkach oznaczają kolejność wykonywania operacji.

Etap 1

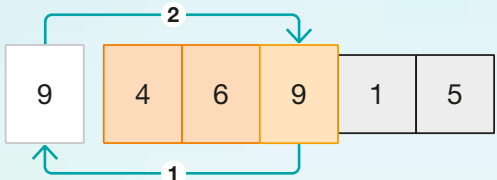
- Przyjmujemy, że pierwszy element listy (liczba 6) jest na właściwym miejscu. Drugi element, liczbę 4, zapamiętujemy w zmiennej pomocniczej.
- Liczba 4 powinna się znaleźć przed liczbą 6, dlatego aby zrobić dla niej miejsce, przesuwamy liczbę 6 o jedną pozycję w prawo.
- Liczbę 4 ze zmiennej pomocniczej zapisujemy jako pierwszy element listy.
- Wynikiem jest uporządkowanie niemalejąco dwóch pierwszych elementów wyjściowej listy.



Postępując analogicznie, w drugim etapie uporządkujemy niemalejąco trzy elementy listy, w następnym – cztery itd. Poniżej znajduje się skrócony opis kolejnych etapów.

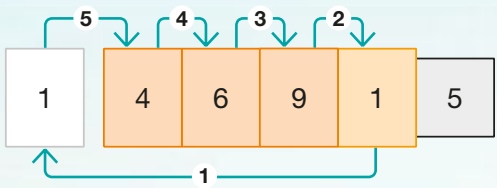
Etap 2

Zapamiętujemy liczbę 9 w zmiennej pomocniczej. Liczba 9 powinna się znaleźć za liczbami 4 i 6, dlatego nie trzeba zmieniać położenia tych liczb, a liczba 9 wraca na swoją pozycję.



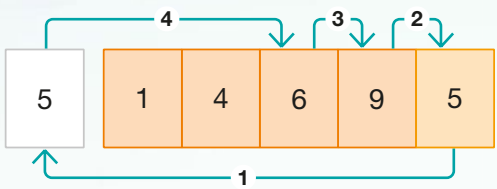
Etap 3

Zapamiętujemy liczbę 1 w zmiennej pomocniczej. Przesuwamy w prawo o jedną pozycję kolejno liczby: 9, 6 i 4. Zapisujemy liczbę 1 na pierwszym miejscu na liście.



Etap 4

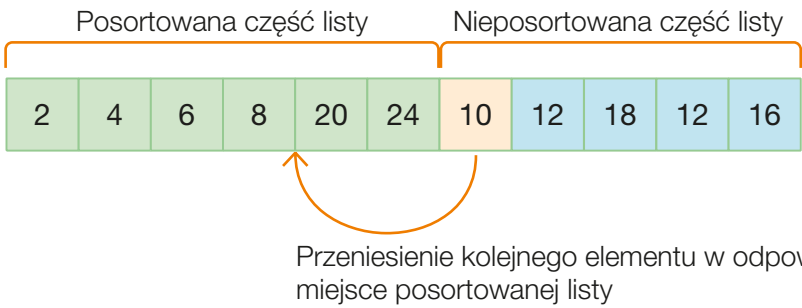
Zapamiętujemy liczbę 5 w zmiennej pomocniczej. Przesuwamy liczbę 9, a następnie liczbę 6 o jedną pozycję w prawo. Wstawiamy liczbę 5 ze zmiennej pomocniczej w zwolnione miejsce.



Wynikiem działania algorytmu jest lista:



Algorytm dzieli listę danych na część posortowaną i nieposortowaną (rys. 9.13). Na każdym etapie algorytmu kolejny element z nieposortowanej części listy trafia na właściwą pozycję w obszarze posortowanym.



Rys. 9.13. Umieszczenie liczby 10 na częściowo uporządkowanej liście

Algorytmy sortowania bąbelkowego i sortowania przez wstawianie są do siebie podobne: pierwszy stopniowo przesuwamy elementy zbioru ku końcowi, podczas gdy drugi przenosi elementy zbioru ku początkowi. Wstawienie elementu listy na właściwą pozycję odbywa się dzięki zwolnieniu miejsca na liście poprzez przesunięcie, krok po kroku, całego bloku elementów o jedną pozycję w prawo.

Przesuwanie elementów jest realizowane przez serię operacji kopiowania. Element, który będzie docelowo wstawiony we właściwe miejsce uporządkowanej części listy, jest tymczasowo przechowywany w zmiennej pomocniczej. Porządkowanie rozpoczynamy od drugiego elementu (o indeksie 1).

Oto zapis algorytmu w postaci listy kroków:

- 1 Wczytaj listę liczb A.
- 2 Dla wartości i równych kolejno 1, 2, ..., n-1 powtarzaj kroki 3-8.
- 3 W zmiennej pom zapamiętaj wartość zapisaną w A[i].
- 4 Oznacz bieżącą wartość i-1 literą j.
- 5 Dopóki $j \geq 0$ i $A[j] > \text{pom}$, wykonuj kroki 6-7.
- 6 Skopiuj A[j] na miejsce A[j+1].
- 7 Zmniejsz wartość j o 1.
- 8 W komórce A[j+1] zapisz wartość pom.

Ćwiczenie 9

Prześledź działanie listy kroków algorytmu sortowania przez wstawianie dla następujących list.

- a. A = [2, 3, 8, 5]
- b. B = [2, 8, 3, 5]
- c. C = [8, 2, 3, 5]

Warto wiedzieć

Sortowanie bąbelkowe i sortowanie przez wstawianie przeprowadzają operacje w miejscu. To oznacza, że poza pamięcią dla sortowanej listy potrzebują bardzo mało dodatkowej pamięci.

Warto wiedzieć

Operację znalezienia właściwego miejsca dla danego elementu w ciągu uporządkowanym można realizować przez przeszukiwanie liniowe lub binarne. Bardziej efektywny jest drugi sposób. Już w latach 50. XX w. analizował go polski matematyk Hugo Steinhaus.

Oto kod źródłowy funkcji o nazwie `Wstawianie` w programie *Sortowanie przez wstawianie*.

Kod źródłowy funkcji  **Wstawianie w programie Sortowanie przez wstawianie**

 Warto wiedzieć

Podany algorytm sortowania przez wstawianie wykonuje operacje kopiowania. Liczba tych operacji jest w przybliżeniu równa liczbie porównań. Kopiowanie nie jest jednak tak czasochłonne jak zamiana, dlatego sortowanie przez wstawianie będzie kilkukrotnie szybsze od sortowania bąbelkowego.

Logika matematyczna i operatory logiczne
→ s. 220–222

Leniwe wartościowanie 

 Warto wiedzieć

Algorytm sortowania przez wstawianie jest szczególnie efektywny przy sortowaniu zbiorów prawie uporządkowanych, np. rankingów aktualizowanych na bieżąco podczas zawodów. Wykorzystuje się go również w rozwiązaniach hybrydowych, m.in. w ostatniej fazie sortowania szybkiego (ang. *quicksort*), gdy do posortowania jest wiele podzbiorów o niewielkiej liczbie elementów.

```
1. def Wstawianie(A):
2.     n = len(A)
3.     for i in range(1, n):
4.         pom = A[i]
5.         j = i - 1
6.         while j >= 0 and A[j] > pom:
7.             A[j+1] = A[j]
8.             j -= 1
9.         A[j+1] = pom
```

Pętla zewnętrzna (linie 3.–9.) przegląda elementy z części nieposortowanej. W kolejnych iteracjach w zmiennej pomocniczej `pom` zapamiętujemy kolejne elementy z tej części (wiersz 4.). Jeżeli element z lewej strony aktualnie analizowanego elementu jest większy ($A[j] > pom$), to przesuwamy elementy posortowanej części ciągu, które powinny się znaleźć za danym elementem, a jego wartość zapamiętaną w zmiennej `pom` wstawiamy na właściwe miejsce (linie 6.–9.).

Warunkiem w wierszu 6. jest **koniunkcja**, czyli złożone wyrażenie logiczne. Program po uruchomieniu będzie działał poprawnie, ponieważ w działaniu interpretera języka Python stosuje się **leniwe wartościowanie** (ang. *lazy evaluation*). W naszym przypadku drugie porównanie się nie wykona, jeśli pierwszy warunek nie jest spełniony, tzn. gdy indeks `j` będzie mniejszy niż indeks pierwszego elementu listy.

W linii 8. zastosowano skróconą formę operacji zmniejszenia wartości zmiennej `j` o 1, równoważną zapisowi `j = j - 1`.

Ćwiczenie 10

Zapisz kod źródłowy funkcji `Wstawianie` i napisz program, w którym tę funkcję zastosujesz. Sprawdź działanie programu.

 Zapamiętaj

Sortowanie przez wstawianie wymaga użycia zmiennej pomocniczej, która tymczasowo przechowuje rozpatrywany element.

 Zadania

- * **1** Napisz program, który wyznaczy najmniejszą liczbę z ciągu liczb całkowitych dodatnich podawanych z klawiatury. Przyjmij, że podanie liczby 0 kończy wprowadzanie liczb.
- * **2** Zmodyfikuj program *Najdłuższy skok* tak, aby trener mógł dodatkowo zliczać liczbę skoków nieważnych (spalonych).
- * **3** Napisz program sprawdzający, czy lista 10 liczb jest uporządkowana.
- * **4** Napisz program, który po wczytaniu 10 liczb dodatnich całkowitych znajdzie różnicę między największą a najmniejszą liczbą.
- * **5** Napisz program, który będzie porządkować malejąco listę 10 różnych liczb metodą:
 - a. bąbelkową,
 - b. przez wstawianie.
- ** **6** Napisz program, który zliczy osobno, ile liczb ujemnych i ile liczb dodatnich wprowadzono z klawiatury. Przyjmij, że wczytanie liczby 0 kończy wczytywanie liczb.
- ** **7** Napisz program, który po wczytaniu 10 liczb wskaże te, które potraktowane jako długości boków utworzą prostokąt o największym polu.
- ** **8** Poszukaj informacji o algorytmie sortowania przez wybór. Napisz program, który będzie sortować liczby całkowite za pomocą tej metody.
- ** **9** Napisz program, który zamieni miejscami elementy znajdujące się w pierwszej i drugiej połowie listy o parzystej liczbie elementów.

Dane wejściowe:

3	1	4	5	2	8	0	3
---	---	---	---	---	---	---	---

Dane wyjściowe:

2	8	0	3	3	1	4	5
---	---	---	---	---	---	---	---

Program nie powinien używać dodatkowej listy pomocniczej.
- *** **10** Dla 6 liczb wprowadzonych z klawiatury zawsze wystarczy 7 porównań, aby wyznaczyć najmniejszą oraz największą z nich. Jak to zrobić? Zapisz algorytm w postaci listy kroków, a następnie jego realizację w języku programowania.
- *** **11** Napisz program sortujący metodą przez wstawianie z użyciem wartownika. Wskazówka: Wartownika umieść jako dodatkowy element w pierwszej komórce listy zawierającej liczby do posortowania.
- *** **12** Napisz program, który po wczytaniu 20 liczb dodatnich całkowitych znajdzie wśród nich parę liczb o najmniejszej różnicy.

Algorytmika i programowanie w języku Python

Wyobraź sobie, że...

...zaczynasz studia na politechnice, ponieważ chcesz zostać inżynierem geodetą. Okazuje się, że na pierwszym roku masz zajęcia z programowania. Na szczęście w szkole poznałaś **podstawowe konstrukcje programistyczne, typy danych i proste struktury danych** oraz zdobyłaś pewne doświadczenie w programowaniu w **środowisku tekstowym**. Radzisz sobie, rozumiesz omawiane algorytmy. Potrafisz je też zaprogramować, mimo że używasz innego języka programowania niż w szkole.



...jesteś inżynierem pożarnictwa i pracujesz w firmie opracowującej zasady bezpieczeństwa dla obiektów biurowych. Wiesz, że instrukcje zawierające sposób postępowania na wypadek pożaru lub innego zagrożenia muszą być bardzo precyzyjne. Jednym ze sposobów ich zapisania jest forma **listy kroków**, z którą zetknąłeś się w szkole podczas nauki programowania. Twoja praca cieszy się dużym uznaniem, ponieważ instrukcje tworzone przez ciebie są dla wszystkich zrozumiałe.

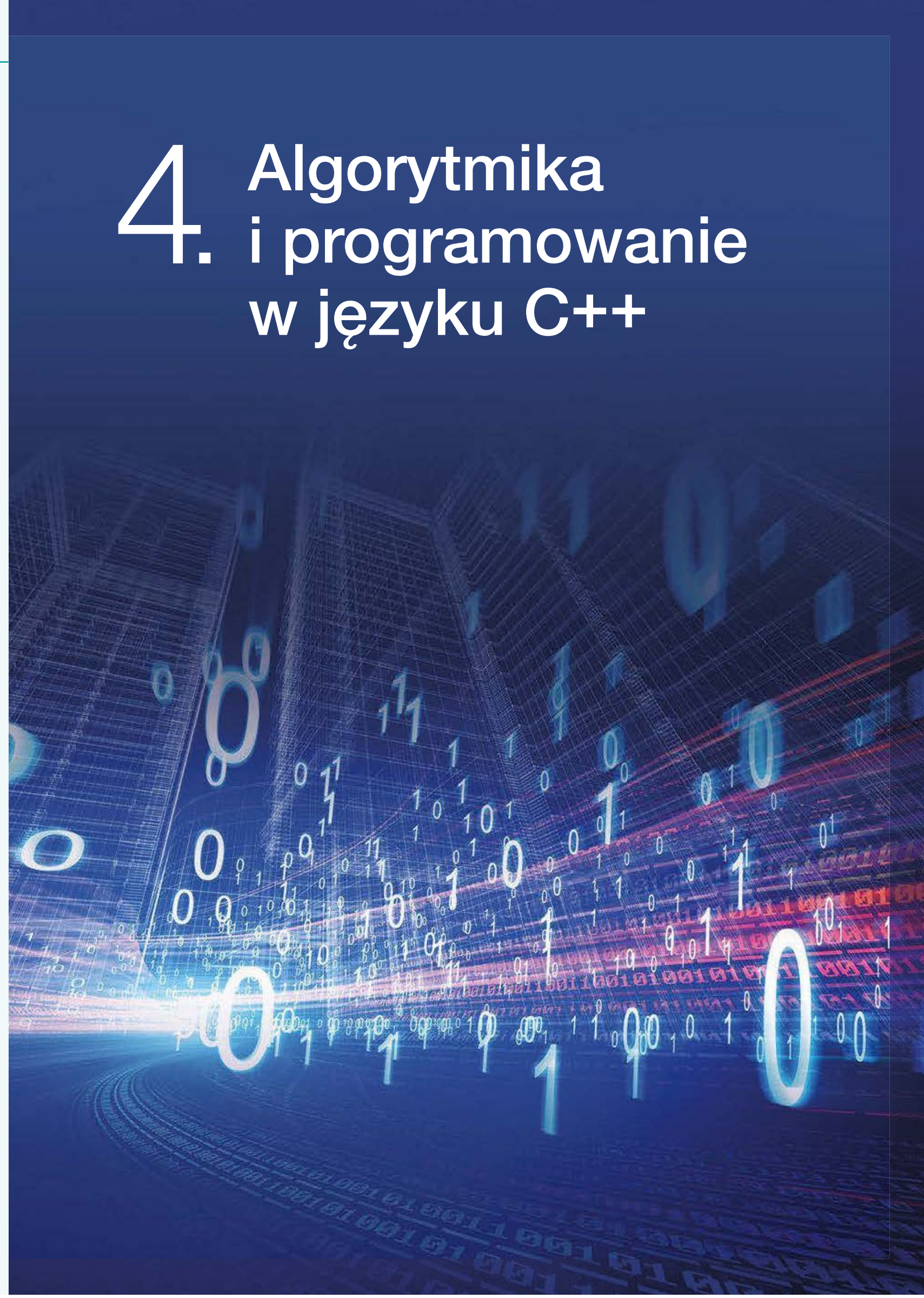
...ukończyłaś studia prawnicze i dostałaś się na staż w znanej kancelarii adwokackiej. Często musisz sięgać po różne akta, które nie są posegregowane. Wiesz, że w uporządkowanym zbiorze łatwiej **wyszukiwać** określone elementy. Przypominasz sobie o algorytmach **sortowania danych** i postanawiasz je wykorzystać. Sortujesz akta według dwóch kryteriów: najpierw chronologicznie, a potem alfabetycznie.



...pracujesz w firmie produkującej meble, która wdraża system wsparcia produkcji wyposażony w sztuczną inteligencję. Na szkoleniu pokazano model sztucznego neuronu wraz z **kodem źródłowym**. Na jego podstawie wyjaśniono podstawę działania całego systemu. Jesteś zachwycony, ponieważ do zrozumienia kodu wystarczyła wiedza o **instrukcji warunkowej** i **pętlach** wyniesiona ze szkoły.



4. Algorytmika i programowanie w języku C++





Algorytmika i programowanie w języku C++

Poniżej znajdziesz podstawowe informacje związane z pisaniem programów w języku C++.

Deklaracja zmiennej jest niezbędną czynnością poprzedzającą jakiekolwiek działania związane z informacjami zapisywanymi w pamięci komputera pod wybraną nazwą (**zmienną**). Ta czynność służy nadaniu nazwy zmiennej oraz określeniu typu informacji, które będą w niej przechowywane.

`int ile;` deklaruje zmienną `ile`, która będzie liczbą całkowitą.

`string imie;` deklaruje zmienną `imie`, która będzie napisem (ciągami znaków).

Instrukcja przypisania (`=`) pozwala na zapisanie w pamięci komputera informacji (danych) w zadeklarowanej zmiennej. Zapisana wartość może się zmieniać w trakcie działania programu.

`ile = 7;` przypisze liczbę 7 zmiennej `ile`.

`imie = "Piotr";` przypisze napis (ciąg znaków) `Piotr` zmiennej `imie`.

Na liczbach i zmiennych, którym przypisano wartości liczbowe, można wykonywać **operacje arytmetyczne**.

`suma = ile + 3;` zwiększy wartość zapisaną w zmiennej `ile` o 3 i zapisze ją w zmiennej `suma`.

`iloczyn = 2 * ile;` podwoi wartość zmiennej `ile` i zapisze ją w zmiennej `iloczyn`.

Instrukcja cin pozwala na wprowadzanie do programu danych z klawiatury.

`cin >> kto;` przypisze zmiennej `kto` tekst wpisany na klawiaturze.

Instrukcja cout pozwala na wyprowadzanie informacji na ekran.

`cout << "Anna";` wypisze na ekranie ciąg znaków `Anna`.

`cout << "Anna i " << imie;` spowoduje wyświetlenie na ekranie ciągu znaków `Anna i Piotr`, jeśli w zmiennej `imie` będzie zapamiętany ciąg znaków `Piotr`.

`cout << 2 * ile;` wypisze na ekranie podwojoną wartość zmiennej `ile`.

Instrukcja warunkowa pozwala wybrać jedną z dróg przebiegu programu, w zależności od spełnienia lub niespełnienia warunku.

```
if (ile > 0)
{
    cout << "Wiecej niż 0";
}
```

wyświetli na ekranie `Wiecej niż 0`, jeśli wartość zmiennej `ile` jest dodatnia.

10

Od problemu do programu

W ostatnim stuleciu rozwój komputerów znacznie ułatwił rozwiązywanie różnego rodzaju problemów. Zadania, których wykonanie zajmuje człowiekowi kilka dni, tygodni, a czasem nawet lat, komputer wykona w zaledwie kilka sekund. Istotę danego problemu możemy przedstawić w odpowiednim, zrozumiałym dla komputera języku, czyli w języku programowania. W tym temacie poznasz podstawy programowania komputerów w języku C++.

Ważne w tym temacie

- Analiza zadania (problemu)
- Tworzenie algorytmu rozwiązania zadania
- Zapis algorytmu w języku programowania
- Testowanie działania programu

Ćwiczenie 1

Z czym kojarzy ci się słowo *algorytm*? Przywołaj sytuacje, w których ktoś użył tego słowa.

10.1. Algorytm – precyzyjna instrukcja

Wiemy, czym są instrukcja, przepis czy procedura. Tymczasem coraz powszechniej używa się technicznego synonimu tych słów, czyli algorytmu. **Algorytm** to precyzyjny opis kolejnych działań, które prowadzą do wykonania zadania (rozwiązania problemu). Pierwotnie matematycy nazywali tak pisemny sposób wykonywania rachunków na liczbach (np. $634 + 87 + 347$).

Z zadaniami algorytmicznymi spotykamy się właściwie na każdym kroku. Wyobraźmy sobie, że na zajęciach wychowania fizycznego każdy uczeń proponuje po jednym ćwiczeniu na rozgrzewkę. Najprościej będzie wtedy ćwiczenie zademonstrować, zamiast o nim mówić (rys. 10.1, s. 152). Gdyby jednak trzeba było zapisać instrukcję kolejnych działań składających się na ćwiczenie fizyczne, można to zrobić, tworząc listę kroków.

Lista kroków to precyzyjny opis postępowania składający się z zapisanych słownie kolejnych pojedynczych instrukcji, nazywanych krokami, które mogą się odwoływać do siebie nawzajem. Lista kroków może się rozgałęziać (mogą być wykonywane różne kroki w zależności od spełnienia lub niespełnienia warunku). Część kroków może też być powtarzana.

Algorytm

Warto wiedzieć

Słowo *procedura* zostało zapożyczone z języka prawniczego, w którym oznacza kolejne kroki procesu, prowadzenia sprawy lub reguły określające te kroki.

Lista kroków



Rys. 10.1. Demonstracja ćwiczenia na rozgrzewkę

Lista kroków algorytmu wykonania ćwiczenia polegającego na wypadach do przodu może wyglądać następująco:

- 1 Stań prosto na płaskiej powierzchni.
- 2 Powtórz 8 razy kroki od 3 do 6 dla obu nóg.
- 3 Pochyl się mocno do przodu, opierając ciężar ciała na jednej nodze.
- 4 Drugą nogę pozostaw prostą.
- 5 Odczekaj 2 sekundy.
- 6 Wróć do pozycji wyjściowej.

Warto zauważyć, jak precyzyjny jest opis kolejnych czynności. W przypadku tych, które należy powtórzyć, podana jest dokładna liczba powtórzeń. Inaczej nie uzyska się oczekiwanego efektu (rozgrzania ciała). Opis ćwiczenia na rozgrzewkę ma **cechy poprawnego algorytmu**, ponieważ:

- ▶ jest skończony – można go zrealizować w skończonej liczbie kroków,
- ▶ jest precyzyjny (dobrze określony i niezawodny) – jednoznacznie definiuje, jakie czynności należy wykonać, aby osiągnąć cel,
- ▶ jest uniwersalny – może być użyteczny w różnych okolicznościach.

Oprócz listy kroków algorytm można **zapisać w inny sposób**, np. w postaci schematu blokowego lub opisu słownego.

Ćwiczenie 2

Przygotuj dwa opisy (w postaci listy kroków) wykonywania zadań z życia codziennego, które mają cechy algorytmu.

Zapamiętaj

Algorytm to opis postępowania prowadzącego do rozwiązania problemu. Znajomość cech poprawnego algorytmu pomaga w tworzeniu dobrych algorytmów.

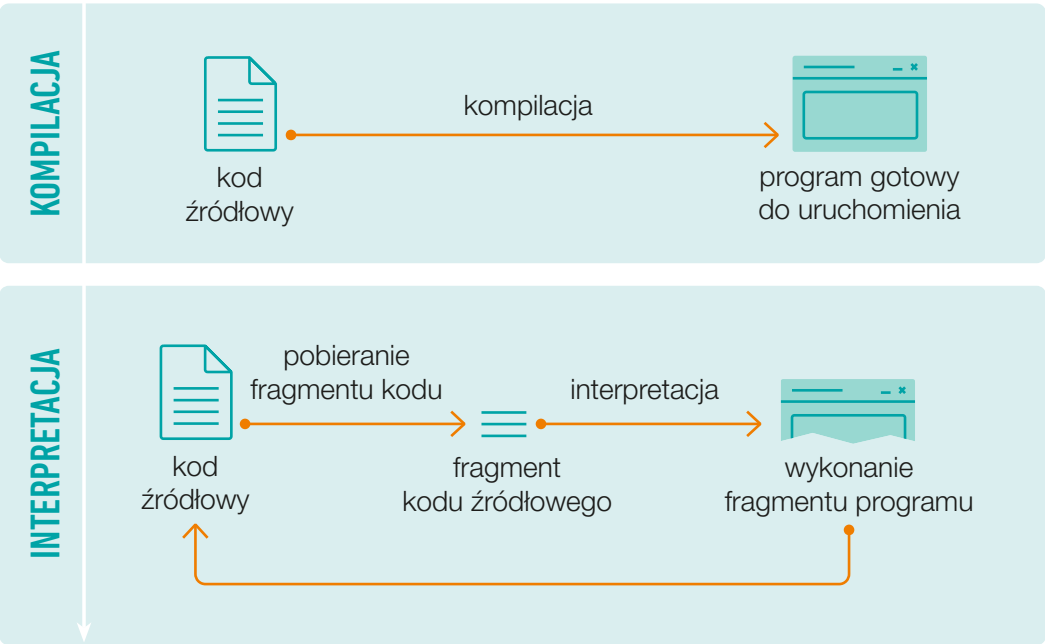
10.2. Od algorytmu do programu komputerowego

Do rozwiązywania problemów z pomocą komputera służą programy komputerowe. W rzeczywistości są one zbiorami algorytmów zapisanych w języku zrozumiałym dla komputera.

Do tworzenia programów komputerowych (oprogramowania) używa się ściśle zdefiniowanych słów i składni **języka programowania**. Zapis algorytmu w takim języku nazywamy **kodem źródłowym** (ang. *source code*) programu komputerowego lub po prostu tekstem programu. Do zapisu kodu źródłowego używa się zwykle specjalnego edytora, często wchodzącego w skład środowiska programistycznego, czyli **IDE**.

Aby można było uruchomić na komputerze program zapisany w kodzie źródłowym, trzeba go najpierw przetłumaczyć **translatorem** na **kod wynikowy (maszynowy)**, czyli taki, który będzie zrozumiały dla procesora.

Translatory, które od razu tworzą kod wynikowy całego programu, nazywamy **kompilatorami**. Drugi typ programów tłumaczących to **interpretery**, które na bieżąco tłumaczą kod źródłowy w trakcie wykonywania programu (rys. 10.2).

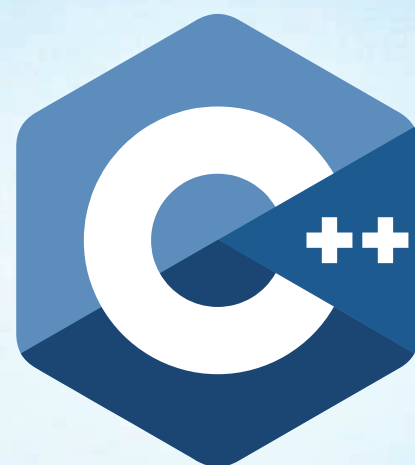


Rys. 10.2. Różnica między kompilacją a interpretacją kodu źródłowego

Trzeba tutaj zaznaczyć, że w przypadku programów zapisanych w językach interpretowanych na komputerze musi być zainstalowany interpreter. Programy skompilowane mogą być uruchamiane bez dodatkowego oprogramowania. Przykładem języka kompilowanego, który omówimy w tym rozdziale, jest **język C++**, a interpretowanego – **język Python**. Przykład prostego kodu źródłowego zapisanego w języku C++ przedstawiamy na infografice na następnej stronie. Znajdują się na niej również najważniejsze zasady, o których należy pamiętać, tworząc programy w tym języku.

ZAPISYWANIE PROGRAMÓW W JĘZYKU C++

Kody źródłowe języka C++ zapisujemy w edytorze kodu połączonym z kompilatorem. Powszechnie korzysta się ze zintegrowanych środowisk programistycznych (IDE). Jednym z nich jest Code::Blocks, który automatycznie koloruje elementy składniowe kodu (np. słowa kluczowe języka, komentarze i liczby).



Przykładowy kod źródłowy programu w języku C++

Te polecenia służą do dołączenia bibliotek. Dzięki nim program będzie mógł wczytywać dane (tekst) z klawiatury i wypisywać wyniki na ekranie.

```
1. #include <iostream>
2. #include <string>
3. using namespace std;
```

To polecenie umożliwi używanie standardowej przestrzeni nazw, co pozwoli na skrócenie zapisu instrukcji (np. `cout` zamiast `std::cout`).

```
4.
5. int main()
6. {
7.     cout << "*** Podaj swoje imie:";
8.     string imie;
9.     cin >> imie;
10.    cout << imie << " ucz się programowania.";
11. }
```

Funkcja `main` to główny blok kodu każdego programu. Początek i koniec bloku oznacza się nawiasami klamrowymi.

Zasady tworzenia kodu w języku C++

Każdą instrukcję kończymy średnikiem.

O typie zmiennej decyduje programista, definiując tę zmienną przed pierwszym użyciem.

Wielkość liter w kodzie ma znaczenie – zmienne `a` i `A` mogą przechowywać różne wartości.

Zaleca się stosowanie wcięć i pustych wierszy, aby zwiększyć czytelność kodu.

Spacje, wcięcia i inne znaki zwane białymi nie mają znaczenia w procesie kompilacji.

Tekst po znakach `//` jest komentarzem.

10.3. Tworzymy własne programy użytkowe

Wykorzystamy język programowania C++. Kody źródłowe programów będziemy pisać w środowisku programistycznym Code::Blocks. Każdy kod będzie się składał z **instrukcji** języka, czyli poleceń zrozumiałych dla kompilatora, który będzie umiał je przetłumaczyć na język zrozumiały dla komputera. Programy będziemy uruchamiać w Wierszu polecenia.

Tworzenie programu krok po kroku:

1. Przygotuj **specyfikację zadania**, czyli określ cel, który chcesz osiągnąć.
2. Zaprojektuj algorytm rozwiązania zadania.
3. Zapisz algorytm w postaci kodu źródłowego.
4. Uruchom i przetestuj program.
5. Popraw ewentualne błędy.

Istotne jest, aby już na etapie tworzenia listy kroków poszczególne czynności (działania) zapisywać zwięźle i jednoznacznie, co uprości proces tworzenia kodu źródłowego.

Program *Powitanie* – interakcja użytkownika z komputerem

Zacniemy od napisania programu, który najpierw zapyta o imię użytkownika, a następnie wyświetli tekst powitalny. Powitanie powinno zawierać imię użytkownika. Najpierw określimy specyfikację zadania. Specyfikacja składa się z danych przekazywanych do programu oraz spodziewanego wyniku działania tego programu.

Specyfikacja

Dane: imię użytkownika wpisane z klawiatury.

Wynik: wyświetlony tekst powitania zawierający imię użytkownika.

Dobra rada

Aktualną wersję programu Code::Blocks możesz pobrać ze strony <http://www.codeblocks.org/>. Jeśli używasz systemu Windows, pobierz wersję oprogramowania, do której dołączono kompilator języka C++ (GCC/G++).

A to ciekawe

Co napisano w C++?

Choć język C++ ma już swoje lata (powstał w 1979 r.), to od dawna utrzymuje się w czołówce najchętniej wykorzystywanych języków programowania. Wiele popularnych programów i gier komputerowych zostało napisanych w tym języku. Wśród gier możemy wymienić World of Warcraft, Wiedźmina, League of Legends, Counter-Strike: Global Offensive, a na liście programów są Adobe Photoshop czy Google Chrome i Mozilla Firefox. Warto tutaj wspomnieć, że znaczna część systemu Windows również została napisana w C++.

Windows 11

Listę kroków algorytmu realizacji powyższego zadania można zapisać następująco:

- 1 Poproś użytkownika o podanie imienia.
- 2 Zapamiętaj podane imię.
- 3 Wypisz na ekranie powitanie zawierające imię użytkownika.

Z kolei kod źródłowy programu realizującego algorytm zapisany w postaci powyższej listy kroków można zapisać tak:

Kod źródłowy programu
Powitanie

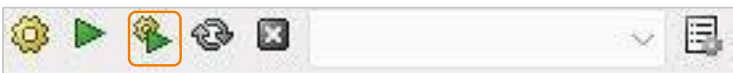
```
1. #include <iostream>
2. #include <string>
3.
4. using namespace std;
5.
6. int main()
7. {
8.     cout << "Podaj swoje imie: " << endl;
9.     string imie;
10.    cin >> imie;
11.    cout << imie << " uczy sie programowania.";
12. }
```

- Instrukcja cout W 8. linijce znajduje się **instrukcja cout**, która wyświetla tekst na ekranie. Mówimy, że jest to **instrukcja wyjścia**, ponieważ dane z komputera są przekazywane do użytkownika. Dane do tej instrukcji są podawane po operatorach << i są określane mianem **argumentów**. W naszym przypadku są dwa argumenty: tekst do wypisania na ekranie oraz **manipulator endl**, który służy do przesunięcia kursora na początek następnego wiersza.
- Zmienna Linijka 9. zaczyna się od zdefiniowania **zmiennej**, czyli miejsca w pamięci, które w trakcie działania programu będzie przechowywało zapisaną wartość. Określamy typ zmiennej (słowo **string** oznacza, że zmienna będzie zawierała tekst) i nadajemy jej nazwę, do której będziemy się odwoływać w kolejnych wierszach kodu.
- Instrukcja cin W wierszu 10. znajduje się **instrukcja cin**, która po wywołaniu poczeka na wpisanie czegoś przez użytkownika i przypisze wpisaną wartość zmiennej imie. Ta instrukcja jest nazywana **instrukcją wejścia**, ponieważ przyjmuje dane od użytkownika.

Dobra rada
Stosuj puste linie do oddzielania logicznych części kodu. Ułatwi to innym programistom analizę tekstu programu.

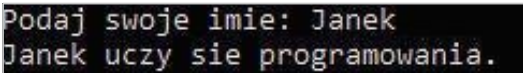
Puste linijki 3. i 5. zostały umieszczone w kodzie ze względów estetycznych. Warto dzielić kod na mniejsze logiczne fragmenty dla lepszej czytelności.
W 11. linijce kodu za pomocą instrukcji **cout** program wyświetli na ekranie zdanie składające się z elementów zapisanych za operatorami <<, czyli wartości zmiennej imie oraz zwrotu **uczy sie programowania**.

Aby rozpocząć tworzenie kodu w środowisku Code::Blocks, wybieramy z menu głównego kolejno **File** → **New** → **Empty file**. Gotowy tekst programu zapisujemy w pliku z rozszerzeniem **cpp**, a następnie kompilujemy go i uruchamiamy, wybierając **Build and run** (rys. 10.3).



Rys. 10.3. Przycisk służący do kompilowania i uruchomienia programu

Efekt uruchomienia programu widzimy na rysunku 10.4.



Rys. 10.4. Wywołanie programu Powitanie

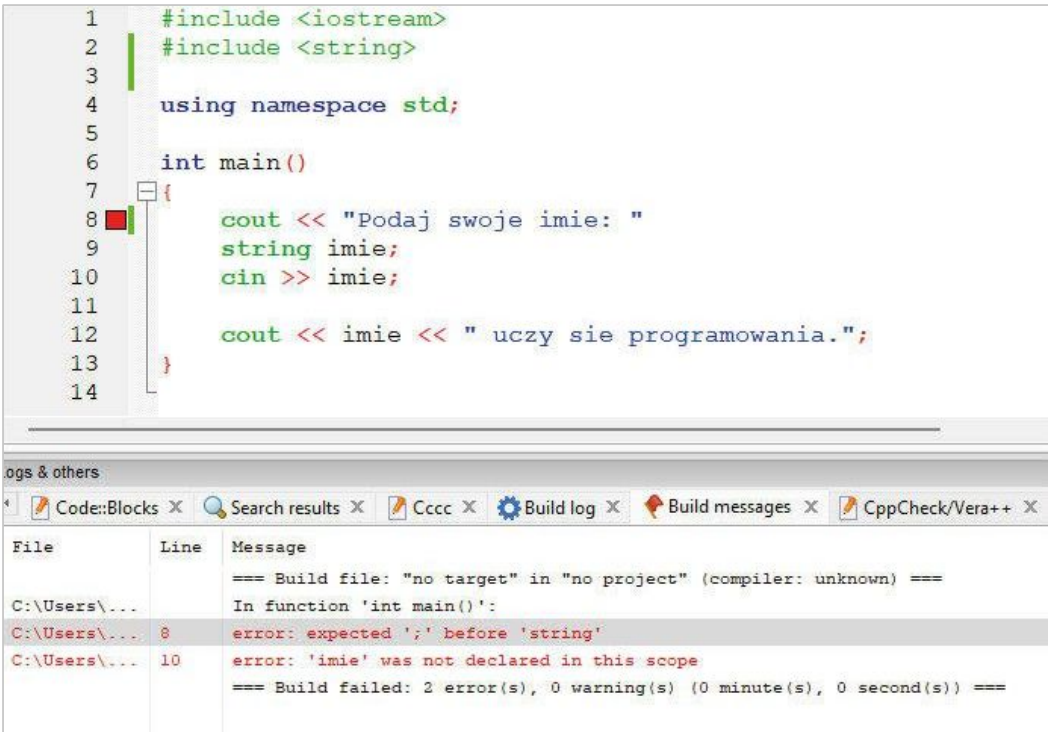
Ćwiczenie 3

Przepisz tekst programu i zapisz go pod nazwą *Powitanie.cpp*. Uruchom program. Jeśli pojawi się informacja o błędzie, popraw tekst programu.

Dobra rada
Możesz też skompilować i uruchomić program, naciskając klawisz F9.

Dobra rada
Edytor Code::Blocks domyślnie zapisuje pliki z rozszerzeniem **.c** (właściwym dla języka C), a więc należy zmienić rozszerzenie na **.cpp**.

Jeżeli w kodzie źródłowym wystąpi **błąd składniowy**, czyli taki, który wynika z zapisania kodu niezgodnie ze **składnią** (czyli z regułami danego języka programowania), to próba skompilowania takiego programu zakończy się wyświetleniem komunikatu o błędzie.
Na przykład jeśli na końcu 8. wiersza nie wstawimy średnika, to komunikat będzie miał taką postać jak na rysunku 10.5.



Rys. 10.5. Błąd składniowy w kodzie źródłowym programu

Błąd polegający na niewstawieniu średnika na końcu instrukcji jest jednym z najczęściej popełnianych błędów przez początkujących programistów. Wynika on z tego, że pisząc kolejne instrukcje w oddzielnych wierszach, można zapomnieć o tym, że to właśnie średnik oddziela te instrukcje od siebie, a nie znak końca wiersza. Można to łatwo sprawdzić, zapisując kilka instrukcji jedna po drugiej w jednym wierszu i oddzielonych tylko średnikami.

Program Szarlotka – korzystamy z instrukcji warunkowej if

Program, który napiszemy, policzy jabłka zebrane przez Anię i Wojtkę w ich przydomowym sadzie. Mamie potrzeba 15 jabłek do upieczenia szarlotki. Po wprowadzeniu liczby jabłek zebranych przez Anię i Wojtkę program zsumuje je i wyświetli informację, czy suma jabłek wystarczy do upieczenia szarlotki.

Specyfikacja

Dane: liczba jabłek zebranych przez Anię i Wojtkę wprowadzona z klawiatury.
Wynik: informacja, czy suma zebranych jabłek wystarczy na szarlotkę, lub komunikat o tym, że liczba jabłek jest niewystarczająca.

Oto propozycja listy kroków algorytmu:

- 1 Zapytaj Anię o to, ile ma jabłek. Zapamiętaj tę liczbę.
- 2 Zapytaj Wojtkę o to, ile ma jabłek. Zapamiętaj tę liczbę.
- 3 Oblicz sumę liczb z kroków 1 i 2. Zapamiętaj tę sumę.
- 4 Jeśli suma jest liczbą mniejszą niż 15, to wypisz informację, że jabłek jest za mało na szarlotkę. W przeciwnym razie wypisz informację, że liczba jabłek jest wystarczająca.

Wykonajmy symulację działania tego algorytmu dla przykładowych liczb, realizując kolejne czynności z listy kroków (tab. 10.1). Na tym etapie można już zaproponować brzmienie tekstu pytań, których użyjemy w programie. Gdyby się okazało, że w projekcie algorytmu jest błąd, to będziemy mogli go usunąć, zanim przystąpimy do zapisywania tekstu programu.

Warto wiedzieć
Błąd algorytmu rozwiązania zadania nazywa się błędem logicznym.

Wyjście	Wejście	Działania
Aniu! Ile masz jabłek?	4	
Wojtku! Ile masz jabłek?	5	
		4 + 5 = 9
		Czy 9 < 15
Jabłek jest za mało.		

Tabela 10.1. Symulacja algorytmu Szarlotka

Oto tekst programu realizującego powyższy algorytm.

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     cout << "Aniu, ile masz jabłek? " << endl;
8.     int ile_Ania;
9.     cin >> ile_Ania;
10.    cout << "Wojtku, ile masz jabłek? " << endl;
11.    int ile_Wojtek;
12.    cin >> ile_Wojtek;
13.
14.    int ile_razem = ile_Ania + ile_Wojtek;
15.    if (ile_razem < 15)
16.    {
17.        cout << "Jabłek jest za mało. " << endl;
18.    }
19.    else
20.    {
21.        cout << "Jabłek wystarczy. " << endl;
22.    }
23. }
```

Kod źródłowy programu Szarlotka

W liniach 8., 11. i 14. są zapisane deklaracje zmiennych z użyciem słowa kluczowego `int`. Dzięki temu kompilator języka C++ potraktuje cyfry wpisane przez użytkownika jako liczbę całkowitą. Wyrażenie `ile_razem = ile_Ania + ile_Wojtek` w 14. linii kompilator zrozumie następująco: oblicz sumę liczb zapamiętanych w zmiennych o podanych nazwach i tę wartość przypisz zmiennej `ile_razem`. W liniach od 15. do 22. zastosowano **instrukcję warunkową `if`**, od której zależy dalszy przebieg programu. Schemat ogólny instrukcji warunkowej wygląda następująco:

```
if (warunek logiczny)
{
    instrukcja na tak;
}
else
{
    instrukcja na nie;
}
```

Warto wiedzieć
Nazwa typu funkcji `int` to skrót angielskiego słowa *integer* oznaczającego liczbę całkowitą.

Instrukcja warunkowa `if`

Jeśli warunek logiczny ma wartość logiczną **prawda**, to jest wykonywana instrukcja na tak, w przeciwnym razie – instrukcja na nie.

Zapis `if (ile_razem) < 15` z 15. linijki mówi o tym, że jeśli wartość zmiennej `ile_razem` jest liczbą mniejszą niż 15, to zostaną wykonane instrukcje zamknięte w nawiasach klamrowych stanowiące

Blok instrukcji ▶ **blok instrukcji** (może być ich wiele). W przeciwnym razie zostaną wykonane instrukcje z bloku znajdującego się po słowie `else`.

Słowa kluczowe ▶ Słowa `if` i `else` to przykłady **słów kluczowych** języka C++, czyli słów zdefiniowanych przez twórcę języka programowania. Programista nie może używać nazw należących do zbioru słów kluczowych w nazwach, które sam tworzy w tekście programu (np. w nazwach zmiennych).

Ćwiczenie 4

Przepisz tekst programu i zapisz go pod nazwą *Szarlotka.cpp*. Skompiluj i uruchom program, a następnie przetestuj go dla różnych danych.

Program *Zbiór jabłek* – wprowadzamy instrukcję powtarzania `while`

Program policzy sumę jabłek zebranych przez Anię i Wojtkę w czasie ich pracy u sadownika. Ania z Wojtkiem powinni zebrać wspólnie 60 kg jabłek, co zdaniem pracodawcy odpowiada 300 jabłkom stosownej wielkości. Mogą oni zbierać jabłka małymi partiami, po kilkanaście lub kilkadziesiąt sztuk. Program będzie tak długo pytał o kolejne wartości, aż suma będzie wynosiła 300 lub więcej.

Specyfikacja

Dane: kolejno wprowadzane liczby jabłek zebranych za każdym razem przez Anię i Wojtkę.

Wynik: sprawdzenie, czy suma zebranych jabłek wynosi przynajmniej 300.

Lista kroków algorytmu może wyglądać tak:

- 1 Ustaw wartość sumy na 0.
- 2 Powtarzaj kroki 3 i 4, dopóki suma jest liczbą mniejszą niż 300.
- 3 Zapytaj o liczbę przyniesionych jabłek. Zapamiętaj tę liczbę.
- 4 Dodaj zapamiętaną liczbę do sumy.
- 5 Wypisz na ekranie informację, że liczba jabłek jest wystarczająca.

Kroki 3 i 4 będą powtarzane tak długo, dopóki warunek określony w kroku 2 będzie spełniony. Symulację działania algorytmu przedstawia tabela 10.2. Odpowiedź „NIE” oznacza, że nastąpiło przejście do 5 kroku algorytmu.

Liczba jabłek przy bieżącym podejściu	Suma jabłek	Czy wartość sumy jest mniejsza niż 300?
–	0	TAK
55	55	TAK
51	106	TAK
63	159	TAK
55	214	TAK
52	266	TAK
50	316	NIE

Tabela 10.2. Symulacja algorytmu *Zbiór jabłek*

Warto zwrócić uwagę, że liczba powtórzeń kroków 3 i 4 zależy od liczby jabłek przynoszonych przez Anię i Wojtkę za każdym razem. Tych powtórzeń może być bardzo dużo, ale może też być tylko jedno. Warto tutaj zaznaczyć, że mogą się też w ogóle nie wykonać, gdybyśmy w pierwszym kroku od razu zadeklarowali wartość sumy jako liczbę większą niż 300.

Poniżej znajduje się ten algorytm w postaci kodu źródłowego języka C++.

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     int ile, suma = 0;
8.     while (suma < 300)
9.     {
10.         cout << "Ile jabłek przyniesliście? ";
11.         cin >> ile;
12.         suma += ile;
13.     }
14.     cout << "Brawo! Uzbieraliście wymagana liczbę jabłek." << endl;
15. }
```

Warto wiedzieć
Mówimy o algorytmie lub programie, że się „zapętlił”, gdy zawiera błąd, którego skutkiem jest wykonywanie w nieskończoność instrukcji tworzących instrukcję powtarzania.

• Kod źródłowy programu *Zbiór jabłek*

W 7. linii zmiennej `suma` przypisujemy początkową wartość 0.

Instrukcja powtarzania (pętla) • Linijka 8. stanowi nagłówek **instrukcji powtarzania**, nazywanej też **pętlą**. W powyższym kodzie tę instrukcję realizujemy dzięki wykorzy-

Pętla while • staniu **pętli while**, która ma następującą budowę:

```
while (warunek)
{
    instrukcja;
}
```

Wszystkie instrukcje znajdujące się w bloku instrukcji w wierszach od 9. do 13. będą powtarzane tak długo, dopóki warunek jest prawdziwy.

Wartość zmiennej `ile` jest przy każdym obrocie pętli podawana przez użytkownika. Wypisaliśmy tekst na ekranie w miejscu, które bezpośrednio poprzedza położenie kursora wskazującego, gdzie ma być wprowadzony tekst z klawiatury.

W 12. linii znajduje się zapis: `suma += ile`. Oznacza on, że poprzednią wartość zmiennej `suma` zwiększamy o aktualną wartość zmiennej `ile` i tę sumę przypisujemy jako nową wartość zmiennej `suma`.

Rysunek 10.6 pokazuje przykładowy przebieg działania programu.

```
Ile jabłek przyniesliscie? 65
Ile jabłek przyniesliscie? 56
Ile jabłek przyniesliscie? 49
Ile jabłek przyniesliscie? 59
Ile jabłek przyniesliscie? 60
Ile jabłek przyniesliscie? 40
Brawo! Uzbieraliscie wymagana liczbe jabłek.
```

Rys. 10.6. Przykładowy przebieg działania programu *Zbiór jabłek*

Ćwiczenie 5

Przepisz kod programu i zapisz go pod nazwą *Zbiór_jabłek.cpp*. Skompiluj i uruchom program i przetestuj go dla różnych danych. Rozpocznij od przykładu danych używanych w tabeli powyżej, podczas symulacji.

Zapamiętaj

Instrukcji warunkowej używa się do wyboru ścieżki przebiegu programu. Pętla służy do wielokrotnego powtarzania bloku instrukcji.

Zadania

- * 1 Przedstaw w postaci listy kroków czynności składające się na:
 - a. wyjście z domu do szkoły (od pobudki do dotarcia do szkoły),
 - b. przygotowanie do wyjazdu na wakacje (pakowanie),
 - c. przygotowanie dowolnej potrawy,
 - d. wstawienie prania.
- * 2 Napisz program, który zapyta użytkownika o imię i ulubiony kolor, a następnie wyświetli zdanie złożone z imienia, słowa *lubi* i nazwy koloru.
- * 3 Napisz program, który wczyta z klawiatury dwie liczby i sprawdzi, czy są one różne. W zależności od tego, czy są różne czy nie, program powinien wyświetlić odpowiedni komunikat.
- * 4 Po opróżnieniu skarbonki Piotr poukładał na stosach monety o nominałach 5 zł, 2 zł i 1 zł oraz banknoty o nominałach 10 zł. Napisz program, który pozwoli obliczyć wartość pieniędzy w skarbonce.
- ** 5 Napisz program, który wyświetli kolejne liczby naturalne od 1 do 10.
- ** 6 Napisz program, który wyświetli w jednym wierszu wszystkie liczby nieparzyste od 1 do 99, a następnie w kolejnym wierszu wyświetli wszystkie liczby parzyste od 2 do 100. Liczby powinny być oddzielone od siebie spacjami.
- ** 7 Ania znalazła taką instrukcję przeliczania lat ludzkich na lata psa: *Wiek 2-letniego psa odpowiada wiekowi 7-letniego dziecka. Każdy kolejny rok ludzki odpowiada 7 latom psim*. Napisz program, który pozwoli Ani przeliczyć wiek dowolnego psa na lata psa.
- ** 8 Napisz program obliczający, ile minut minęło od północy. Użytkownik podaje informację o bieżącym czasie (godzina i minuta) w postaci liczb zapisanych w kolejnych wierszach.
- *** 9 Napisz program, który wczyta z klawiatury liczbę całkowitą dodatnią i sprawdzi, z ilu cyfr składa się ta liczba.
- *** 10 Napisz program, który przeliczy informację o bieżącym czasie z formatu 24-godzinnego na 12-godzinny.
- *** 11 Napisz program, który będzie określać porę dnia (noc, dzień, wieczór) na podstawie informacji o godzinie podanej z dokładnością do minuty. Danymi do programu powinny być dwie liczby określające aktualny czas zegarowy.

11 Systemy liczbowe

Kiedy korzystasz z komputera lub smartfona, wykonujesz wiele różnych czynności, m.in. słuchasz muzyki, przetwarzasz zdjęcia, redagujesz teksty, wysyłasz e-maile i oglądasz transmisje wideo. To wszystko jest możliwe dzięki temu, że zarówno oprogramowanie, jak i różnego typu dane są zapisywane w pamięci komputera w jednolity sposób – cyfrowo, czyli jako liczby w systemie binarnym (dwójkowym).

Ważne w tym temacie

- Zapisywanie liczb w systemie binarnym
- Inne systemy liczbowe
- Cyfrowy zapis informacji
- Algorytmy zamiany reprezentacji liczb
- Korzystanie ze struktury tablicy w języku C++
- Zaprogramowanie algorytmu do zamiany liczby dziesiętnej na liczbę dwójkową

11.1. Liczby dziesiętne i dwójkowe

System dziesiętny (decymalny)

Gdy na co dzień wykorzystujemy dane liczbowe, zazwyczaj posługujemy się **systemem dziesiętnym (decymalnym)**. Dlaczego właśnie ten system się upowszechnił? Zapewne dlatego, że pierwszymi liczydłami były palce u dłoni człowieka. Liczby zapisujemy więc za pomocą dziesięciu cyfr: 0, 1, 2, 3, 4, 5, 6, 7, 8 i 9.

System pozycyjny

Naturalne jest dla nas, że zapis 2350 oznacza liczbę dziesięć razy większą od 235. Wynika to z faktu, że system dziesiętny jest **systemem pozycyjnym**. To oznacza, że z pozycją cyfry (liczoną od prawej strony) wiąże się jej waga – w tym wypadku: 1, 10, 100, 1000 itd. Są to kolejne potęgi liczby 10, która w tym przypadku jest **podstawą systemu liczbowego**. Każdą liczbę przedstawia się w tym systemie jako sumę wielokrotności potęg liczby 10. Na przykład liczba 235 jest sumą dwóch potęg 10^2 , trzech potęg 10^1 i pięciu potęg 10^0 .

Na rysunku 11.1 pokazano zapis liczby 235 w systemie dziesiętnym.

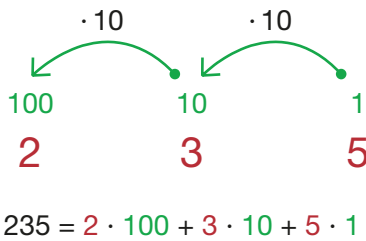
Warto wiedzieć

Słowo *waga* w matematyce oznacza czynnik liczbowy, którym poprzedza się jakąś liczbę, aby zaznaczyć stopień jej ważności (istotności).

Dziesiętny system pozycyjny

Podstawa systemu: 10
Cyfry: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Wagi cyfr na kolejnych pozycjach (liczonych od prawej strony) są kolejnymi potęgami liczby dziesięć: $10^0, 10^1, 10^2, 10^3, \dots$



Rys. 11.1. Zapis dziesiętny liczby 235

W technice komputerowej szczególne znaczenie ma **system dwójkowy (binarny)**, w którym występują tylko dwie cyfry: 0 i 1. Wagi cyfr na kolejnych pozycjach w zapisie liczby w systemie dwójkowym są potęgami liczby 2. Na rysunku 11.2 przedstawiono binarną postać liczby 235.

System dwójkowy (binarny)

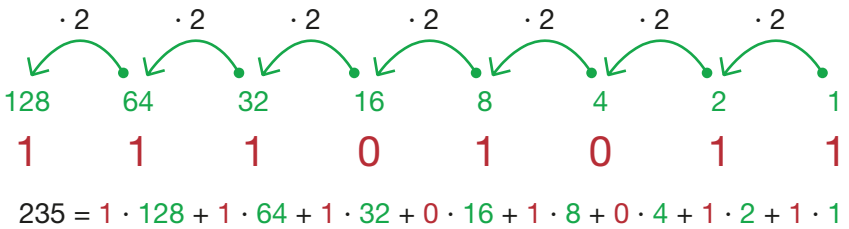
Warto wiedzieć

O liczbie przedstawionej w systemie dwójkowym mówi się w skrócie *liczba dwójkowa*.

Dwójkowy system pozycyjny

Podstawa systemu: 2
Cyfry: 0, 1

Wagi cyfr na kolejnych pozycjach są kolejnymi potęgami liczby dwa: $2^0, 2^1, 2^2, 2^3, \dots$



Rys. 11.2. Zapis dwójkowy liczby 235

Zauważ, że liczba 235 jest sumą $2^7, 2^6, 2^5, 2^3, 2$ oraz 1 i jest – w zapisie dwójkowym – liczbą ośmiocyfrową 11101011.

Tabela 11.1 zawiera zapis binarny liczb od 0 do 15. Na tej podstawie możemy sformułować kilka **podstawowych własności liczb binarnych**:

1. Liczby nieparzyste w zapisie binarnym kończą się cyfrą 1.
2. Liczby binarne kończące się cyfrą 0 dzielą się przez 2, a więc są parzyste.
3. Aby uzyskać liczbę binarną dwa razy większą od wyjściowej, należy na końcu dopisać cyfrę 0. Oznacza to, że jeśli usuniemy ostatnią cyfrę parzystej liczby binarnej, otrzymamy połowę tej liczby.
4. Zapisy dwóch kolejnych liczb binarnych – mniejszej parzystej i większej nieparzystej – różnią się tylko ostatnią cyfrą. Liczba parzysta kończy się cyfrą 0, a nieparzysta – cyfrą 1, np. 12 to 1100, a 13 to 1101.

Takich własności można sformułować więcej i wszystkie można matematycznie udowodnić.

Liczba	Zapis binarny	Uzasadnienie
0	0	0
1	1	2^0
2	10	2^1
3	11	$2^1 + 2^0$
4	100	2^2
5	101	$2^2 + 2^0$
6	110	$2^2 + 2^1$
7	111	$2^2 + 2^1 + 2^0$
8	1000	2^3
9	1001	$2^3 + 2^0$
10	1010	$2^3 + 2^1$
11	1011	$2^3 + 2^1 + 2^0$
12	1100	$2^3 + 2^2$
13	1101	$2^3 + 2^2 + 2^0$
14	1110	$2^3 + 2^2 + 2^1$
15	1111	$2^3 + 2^2 + 2^1 + 2^0$

Tabela 11.1. Zapis binarny liczb od 0 do 15

Podstawowe własności liczb binarnych

Dobra rada

Liczb dwójkowych nie nazywaj tak samo jak liczb w systemie dziesiętnym. Na przykład liczbę zapisaną jako 1010 w systemie dziesiętnym nazywamy „tysiąc dziesięć”, ale w systemie binarnym należy ten zapis odczytać „jeden zero jeden zero”.

Warto wiedzieć

Liczby większe od 1 zapisane w postaci binarnej składają się z większej liczby cyfr niż ich odpowiedniki w postaci dziesiętnej. Z punktu widzenia człowieka to z pewnością wada, dlatego binarnym systemem liczbowym nie posługujemy się na co dzień.

Zapamiętaj

Dwójkowy system liczbowy (podobnie jak system dziesiętny) opiera się na zasadzie zapisu wagowo-pozycyjnego. Cyfry w zapisie liczby znajdują się na ponumerowanych pozycjach. Każdej pozycji (licząc od prawej do lewej) odpowiada pewna waga: 1, 2, 4, 8 itd. Wagi są kolejnymi potęgami liczby 2, którą nazywa się podstawą tego systemu liczbowego.

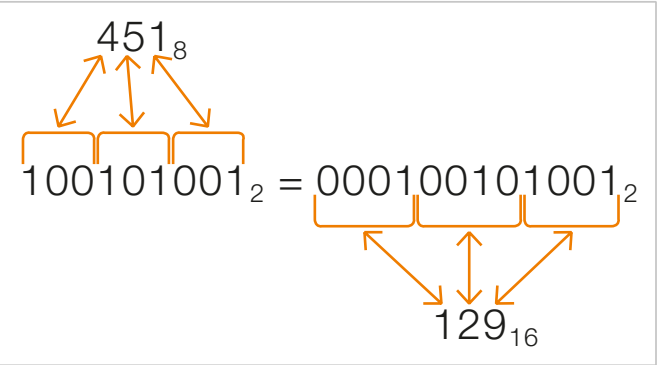
Ćwiczenie 1

Jaką największą liczbę można zapisać w systemie dwójkowym za pomocą ośmiu cyfr?

Inne systemy liczbowe

W informatyce często wykorzystuje się również systemy ósemkowy i szesnastkowy. W **systemie ósemkowym** liczby tworzy się z ośmiu cyfr (od 0 do 7), natomiast w **systemie szesnastkowym** jest ich aż szesnaście: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Do zapisu sześciu ostatnich cyfr wykorzystuje się znaki będące kolejnymi literami alfabetu łacińskiego. Litera F w systemie szesnastkowym odpowiada liczbie 15 w systemie dziesiętnym. Aby oznaczyć, w jakim systemie została zapisana liczba, w indeksie dolnym zapisuje się podstawę systemu (rys. 11.3).

Podstawy systemów szesnastkowego i ósemkowego są potęgami liczby 2, dlatego stosunkowo łatwo można zamienić liczby zapisane w tych systemach na reprezentację w systemie binarnym i odwrotnie. Każdą cyfrę ósemkową można zapisać za pomocą trzech cyfr binarnych, a każdą cyfrę szesnastkową – za pomocą czterech cyfr binarnych, przy czym mniejsze wartości uzupełniamy z lewej strony zerami (rys. 11.3).



Rys. 11.3. Zamiana zapisu liczby pomiędzy systemami dwójkowym, ósemkowym i szesnastkowym

11.2. Cyfrowy zapis informacji

Wszelkiego typu informacje (liczby, teksty, grafika, dźwięk itd.) w systemach komputerowych są przedstawiane (kodowane) jako liczby.

Bit • Najmniejszą, niepodzielną porcją informacji jest **bit** (od ang. *binary digit*), czyli informacja, którą można przedstawić w postaci jednej cyfry dwójkowej: 0 albo 1.

Dobra rada

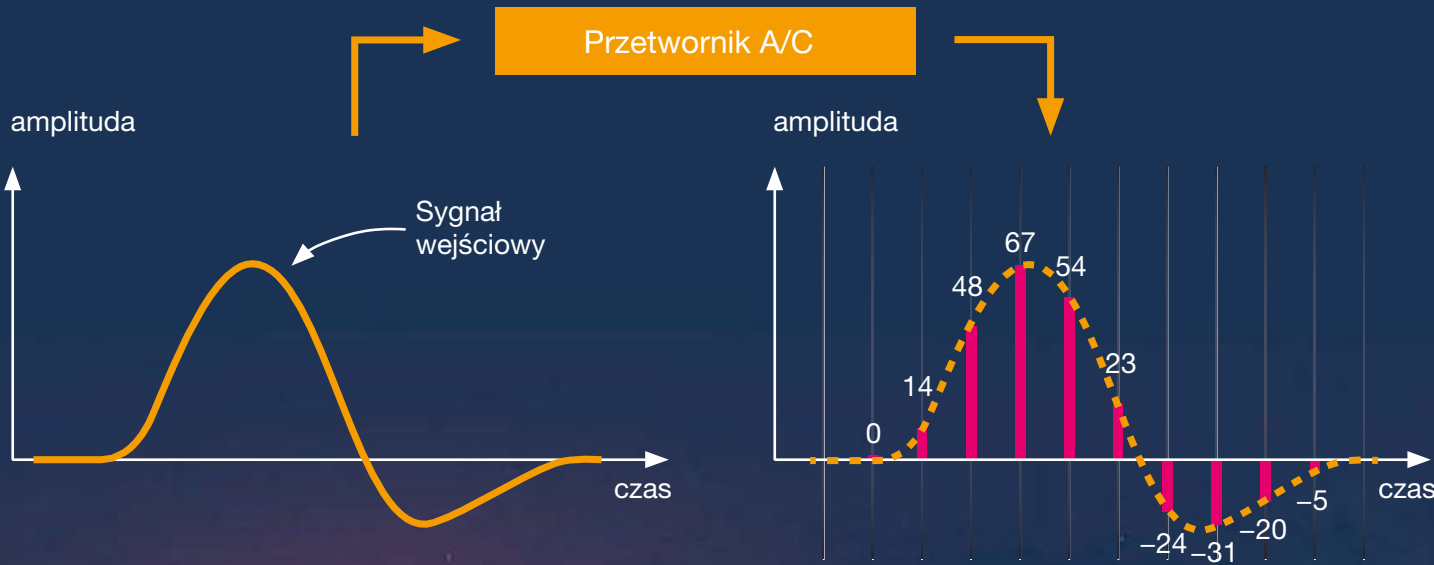
Jeśli posługujesz się jednocześnie przynajmniej dwoma systemami liczbowymi, stosuj dodatkowy indeks dolny do określenia podstawy systemu, w którym liczba została zapisana. Jest to szczególnie ważne w przypadku liczb, których postaci wyglądają tak samo w różnych systemach, ale mają różne wartości, np.: $100_{10} = 1100100_2$, a $100_2 = 4_{10}$.

JAK DANE ANALOGOWE SĄ PRZEKSZTAŁCANE NA CYFROWE?

Podczas nagrywania filmu smartfonem dźwięk i ruchomy obraz są przetwarzane na postać cyfrową. Dzięki temu możemy przechowywać je w pamięci telefonu, a następnie odtwarzać. Jak działają cyfrowe urządzenia rejestrujące dźwięk i obraz? Przeanalizujmy to na przykładzie.

Mikrofon wbudowany w telefon zamienia falę dźwiękową na sygnały elektryczne odpowiadające amplitudzie i częstotliwości dźwięku.

Przetwornik A/C (analogowo-cyfrowy) próbuje amplitudy sygnałów elektrycznych, czyli mierzy je tysiące razy w ciągu jednej sekundy i zamienia na ciągi liczb binarnych. Aby zapis cyfrowy był zgodny z dźwiękiem, który słyszy ludzkie ucho, próbkowanie musi się odbywać z odpowiednio wysoką częstotliwością i rozdzielczością. Na przykład dla płyt CD standardem są częstotliwość 44,1 kHz i rozdzielczość 16 bitów na próbkę, czyli $2^{16} = 65\,536$ możliwych wartości amplitudy fali dźwiękowej.



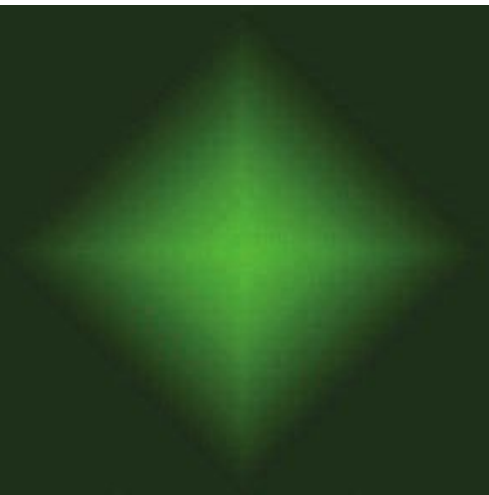
Jak w komputerze zapisuje się obrazy cyfrowe?

Aby wyjaśnić dokładniej istotę cyfrowego zapisu informacji, posłużymy się przykładem z dziedziny grafiki komputerowej. Zobaczymy, jakie informacje może zawierać jeden piksel obrazu.

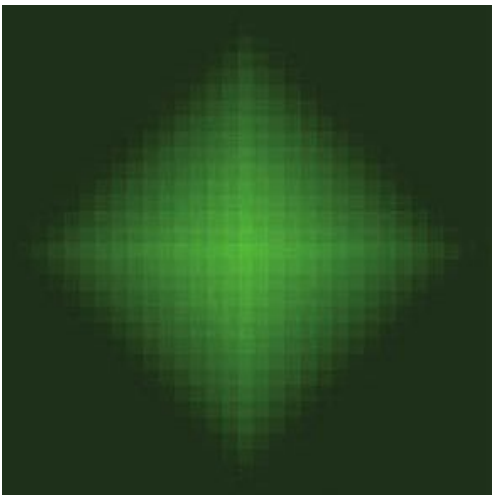
Rysunek 11.4 to przykład gradientu, czyli wypełnienia obszaru równomiernym przejściem z jednego odcienia tej samej barwy do innego.

Rysunek 11.5 powstał z rysunku 11.4 w wyniku zmniejszenia rozdzielczości, czego efektem jest mniejsza liczba pikseli.

Warto wiedzieć
Graficy komputerowi i osoby zajmujące się projektowaniem wizualnych aspektów stron internetowych zwykle zapisują informacje o kolorach za pomocą systemu szesnastkowego.



Rys. 11.4. Gradient



Rys. 11.5. Gradient po zmniejszeniu rozdzielczości obrazu

Rozdzielczość obrazu
podręcznik NOWA Informatyka na czasie 1. Zakres podstawowy s. 46

Dzięki zastosowaniu odpowiedniej **rozdzielczości** mamy wrażenie, że obraz z rysunku 11.4 jest ciągły – kolory płynnie przechodzą jeden w drugi. Obraz z rysunku 11.5 sprawia wrażenie zbioru kwadratów o różnych odcieniach zieleni, które są ułożone schodkowo.

Przyjrzyjmy się dokładnie rysunkowi 11.5. W poziomie i w pionie mamy po 31 pikseli. Na rysunku 11.6 pokazano środkową linię tego obrazu w powiększeniu:



Rys. 11.6. Środkowa linia obrazu w powiększeniu

Warto wiedzieć
Czasami zamiast słowa *piksel* używa się słowa *pel* (ang. *picture element*). Stosuje się je zwłaszcza wtedy, gdy mówimy o elementach obrazu wydrukowanego.



Rys. 11.7. Środkowa linia obrazu z ponumerowanymi odcieniami

W cyfrowej reprezentacji każda informacja, również informacja o obrazie, musi mieć postać liczbową. Każdemu z odcieni można przypisać liczbę całkowitą, począwszy od 0.

Jeśli przyjąć, że odcień o najmniejszej intensywności (jasności) opiszemy liczbą 0, a ten o największej – 15, to piksele tworzące tę linię obrazu opiszemy ciągiem 31 liczb. Przedstawiono to na rysunku 11.7.

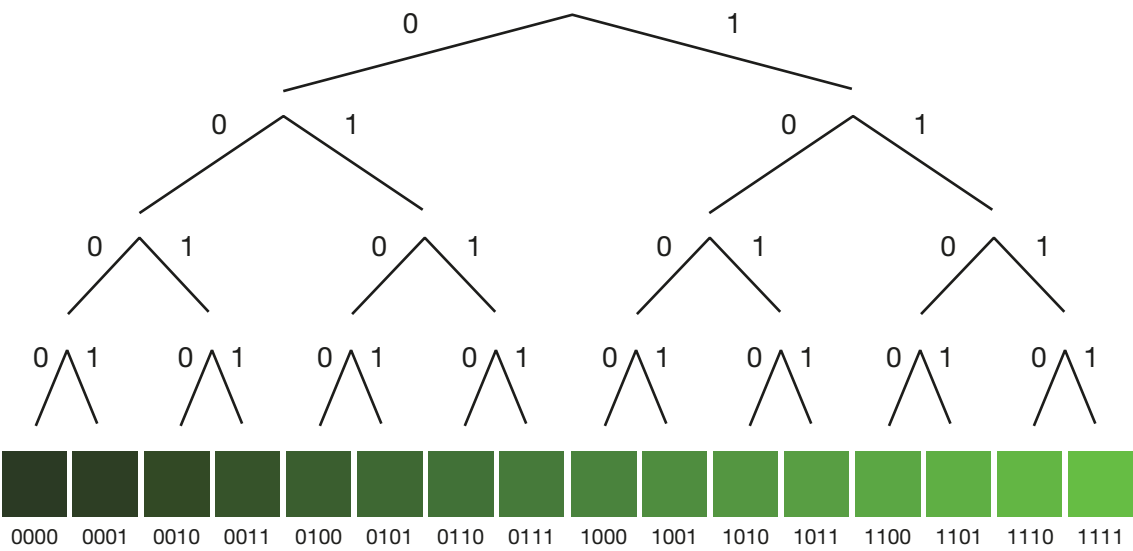
Każda z tych liczb dziesiętnych w pamięci komputera lub na nośniku danych jest w rzeczywistości zapisywana jako czterocyfrowa liczba dwójkowa. Aby zrozumieć, w jaki sposób są tworzone kody kolorów, przeanalizujmy diagram z rysunku 11.8.

Różnym odcieniom koloru zielonego przyporządkowano liczby binarne od 0000 do 1111. **Bit najbardziej znaczący** – reprezentowany przez pierwszą od lewej cyfrę liczby binarnej – decyduje o tym, czy dany odcień jest w grupie ciemniejszych czy jaśniejszych.

Diagram z rysunku 11.8 ma formę **drzewa decyzyjnego** – czytamy je od góry. Jeśli postawilibyśmy pytanie „Czy dany odcień jest jednym z jaśniejszych?”, to odpowiedź „tak” jest reprezentowana przez cyfrę 1, a odpowiedź „nie” – przez 0. Dlatego wszystkie odcienie bliższe czerni mają pierwszy bit o wartości 0.

Drugi bit określa, czy odcień jest wśród $\frac{1}{4}$ najciemniejszych czy wśród $\frac{1}{4}$ średnio ciemnych (lub analogicznie $\frac{1}{4}$ najjaśniejszych czy w $\frac{1}{4}$ średnio jasnych).

Po wyznaczeniu trzeciego bitu wybór organicza się do dwóch odcieni, a **bit najmniej znaczący** – reprezentowany przez ostatnią cyfrę liczby binarnej – pozwala jednoznacznie określić odcień docelowy.



Rys. 11.8. Diagram ilustrujący tworzenie kodów odcieni w obrazie

Każdy odcień koloru zielonego na obrazie można więc jednoznacznie przedstawić jako czterocyfrową liczbę binarną. Do opisanego jednego piksela obrazu z rysunku 11.7 ze s. 168 potrzeba zatem 4 bitów.

Zapamiętaj
Wszelkiego typu informacje przechowywane i przetwarzane w systemach komputerowych są przedstawiane (kodowane) jako liczby binarne, w postaci ciągu cyfr 0 i 1.

Warto wiedzieć
W latach 40. XX w. w pierwszych komputerach nie korzystano z systemu binarnego. W słynnym komputerze ENIAC zastosowano system dwupiętkowy.

11.3. Algorytmy zamiany reprezentacji liczb

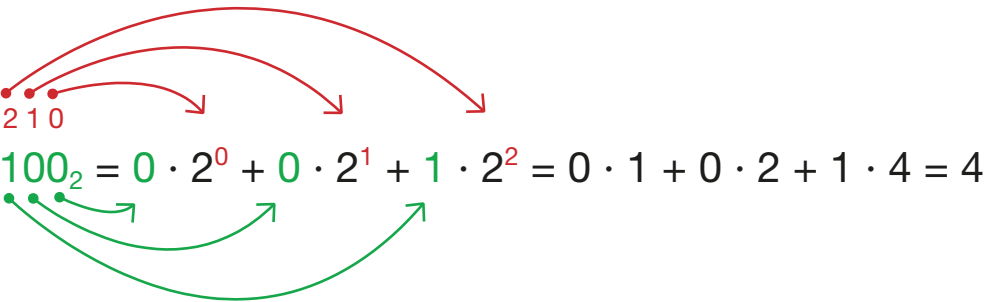
W komputerze liczby zapisuje się w systemie dwójkowym, jednak użytkownicy są przyzwyczajeni do używania systemu dziesiętnego. Dlatego np. podczas pracy z kodem HTML, tworzenia grafiki komputerowej czy konfiguracji sieci pojawia się potrzeba konwersji między systemami liczbowymi.

Zamiana liczby binarnej na liczbę dziesiętną

Zapis dziesiętny liczby binarnej można wyznaczyć, sumując kolejne potęgi liczby 2 odpowiadające wagom kolejnych cyfr. Algorytm można zapisać następująco:

- 1 Ponumeruj cyfry liczby binarnej od prawej do lewej, zaczynając od 0.
- 2 Każdą cyfrę potraktuj odpowiednio jako liczbę dziesiętną 0 albo 1 i pomnóż przez liczbę 2 podniesioną do takiej potęgi, jaki jest numer cyfry.
- 3 Zsumuj otrzymane iloczyny.

Zamianę liczby binarnej 100₂ na postać dziesiętną ilustruje rysunek 11.9.



Rys. 11.9. Schemat zamiany liczby binarnej na dziesiętną

Ćwiczenie 2

Zapisz poniższe liczby binarne w systemie dziesiętnym.

- a. 111₂
- b. 10001₂
- c. 1101101₂

Zamiana liczby dziesiętnej na liczbę binarną

Założmy, że chcemy zapisać w postaci binarnej liczbę dziesiętną 123. Aby wyznaczyć kolejne cyfry (bity) zapisu binarnego, musimy wskazać potęgi liczby 2, które w sumie dają tę liczbę.

Zacznijmy od wyznaczenia najmniej znaczącego, czyli ostatniego bitu. Z podanych wcześniej **własności liczb binarnych** wiemy, że wszystkie liczby nieparzyste kończą się cyfrą 1 (wynika to z własności 1), dlatego zapis binarny liczby 123 będzie się kończyć cyfrą 1.

Liczby 122 i 123 to kolejne liczby całkowite, więc z własności 4. wiemy, że wszystkie bity poza ostatnim będą takie same jak bity liczby 122. Przedostatni bit liczby 122 (a więc i 123) będzie taki jak ostatni bit liczby o połowę mniejszej, a więc 61 (wynika to z własności 3.). Ta liczba jest nieparzysta, więc jej postać binarna będzie się kończyć cyfrą 1. To oznacza, że zapis binarny liczby 122 kończy się parą cyfr 10 (jeden, zero), a liczby 123 – parą 11 (jeden, jeden).

Jak znaleźć kolejną cyfrę (trzecią od prawej) zapisu binarnego liczby 123? Wystarczy powtórzyć powyższe rozumowanie: określić parzystość liczby (61 – 1)/2. Wyznaczanie parzystości kolejnych ilorazów należy powtarzać do momentu, gdy otrzymamy iloraz równy 0.

Przebieg tego algorytmu można przedstawić w tabelarycznej formie (tab. 11.2).

Liczba	Iloraz z dzielenia przez 2	Reszta z dzielenia przez 2
123	61	1
61	30	1
30	15	0
15	7	1
7	3	1
3	1	1
1	0	1

Tabela 11.2. Przebieg algorytmu zapisu liczby 123 w postaci binarnej

Ostatnią kolumnę czyta się od dołu i zapisuje cyfry od lewej do prawej. Liczba 123₁₀ w zapisie binarnym to 1111011₂.

Dobra rada

Kolejne etapy realizacji danego algorytmu możesz w wygodny sposób umieścić w tabeli. Może ona pomóc w zrozumieniu trudniejszego algorytmu oraz pozwala znaleźć ewentualne błędy.

A to ciekawe

Komputery analogowe

Maszyny analogowe, inaczej niż urządzenia cyfrowe, przetwarzały dane w postaci analogowej, tj. za pomocą wielkości fizycznych zmieniających się w sposób ciągły, takich jak napięcie elektryczne. Jednym z pierwszych komputerów analogowych na świecie był analizator AKAT-1, skonstruowany w 1957 r. przez zespół pod kierunkiem inż. Jacka Karpińskiego. Komputer pomagał naukowcom w analizie dynamicznie zmieniających się procesów, takich jak przepływy cieczy, wymiana ciepła, resorowanie pojazdów. Ergonomiczną obudowę komputera zaprojektowano na warszawskiej Akademii Sztuk Pięknych.



Podstawowe własności liczb binarnych → s. 165

 Dobra rada

Jeśli chcesz wyznaczyć postać binarną wielu liczb, możesz skorzystać z funkcji `DZIES.NA.DWÓJK` lub `DEC2BIN` w arkuszu kalkulacyjnym.

Ćwiczenie 3

Wyznacz postać binarną liczby dziesiętnej 117. Zapisz przebieg algorytmu w tabeli. Jego poprawność możesz sprawdzić np. w kalkulatorze systemu Windows (tryb programisty) albo korzystając z odpowiedniej aplikacji w telefonie lub w serwisie internetowym.

11.4. Program zamieniający liczbę dziesiętną na binarną

Przygotujemy się teraz do stworzenia programu, który po wpisaniu z klawiatury liczby naturalnej, nie większej niż 255, wypisze ciąg zer i jedynek przedstawiający binarną reprezentację komputerową tej liczby. Ograniczymy się do liczb, które da się zapisać w jednym

Bajt  **bajcie** (ang. *byte*), czyli jako liczby binarne złożone z ośmiu bitów. Zapiszmy specyfikację problemu algorytmicznego.

 Warto wiedzieć

Jeden bajt reprezentuje aż $2^8 = 256$ różnych liczb binarnych. Tę właściwość wykorzystano w kodzie ASCII, w którym za pomocą jednobajtowych kodów przedstawia się litery alfabetu łacińskiego (języka angielskiego), cyfry, znaki przestankowe, symbole i inne znaki służące do komunikacji tekstowej.

Specyfikacja

Dane: liczba całkowita (nieujemna) nie większa niż 255.

Wynik: zapis binarny liczby całkowitej.

Aby wyznaczyć postać binarną liczby dziesiętnej, możemy postępować w sposób przedstawiony w tabeli 11.2. Wystarczy zapamiętywać kolejne ilorazy z dzielenia przez 2 oraz otrzymywane reszty z dzielenia. Dokładny zapis działania algorytmu przedstawia poniższa lista kroków.

- 1 Podziel liczbę przez 2, zapisz resztę z dzielenia i zapamiętaj iloraz (część całkowitą).
- 2 Dopóki ostatnio otrzymany iloraz jest większy od 0, powtarzaj kroki 3, 4 i 5.
- 3 Wyznacz resztę z dzielenia ostatnio zapamiętanego ilorazu przez 2.
- 4 Zapisz resztę z dzielenia przed ostatnią, wcześniej zapisaną resztą.
- 5 Podziel iloraz przez 2 i zapamiętaj część całkowitą wyniku jako nowy iloraz.

Po otrzymaniu ilorazu zerowego postać binarną tworzą zapisane reszty.

Do zapisania algorytmu w języku C++ potrzebna będzie nam nowa wiedza, ponieważ będziemy wyznaczać reszty z dzielenia oraz zapamiętywać kolejne ilorazy (części całkowite). Do tego celu wykorzystamy operator wyznaczania reszty z dzielenia oraz strukturę tablicy.

Dzielenie całkowite i reszta z dzielenia

W programowaniu rozróżnia się dwie operacje dzielenia, gdy dzielna i dzielnik są liczbami całkowitymi: dzielenie w sensie znanym ze szkoły podstawowej (np. operacja $123/2$ daje iloraz równy 61,5) i dzielenie całkowite (operacja $123/2$ daje iloraz całkowity 61 i resztę 1).

W języku C++ w obu przypadkach stosuje się operator `/`. W przypadku gdy dzielimy liczbę całkowitą przez liczbę całkowitą, to domyślnie zwracanym wynikiem jest iloraz całkowity. Resztę z dzielenia wyznaczamy z wykorzystaniem operatora `%`, zwanego **modulo**. Na przykład wartością zmiennej `iloraz` po deklaracji zmiennej i jednoczesnym przypisaniu `int iloraz = 123/2` będzie 61, a wartością zmiennej `reszta` po deklaracji i przypisaniu `reszta = 123%2` będzie 1.

 **Modulo** to operacja wyznaczania reszty z dzielenia jednej liczby przez drugą.

Ćwiczenie 4

Sprawdź efekt działania programu, którego funkcja `main` ma postać:

```
int iloraz = 123/2;
int reszta = 123%2;
cout << iloraz;
cout << reszta;
```

 Warto wiedzieć

Dostępny na standardowej klawiaturze symbol procentu `%` wybrano jako operator wyznaczania reszty z dzielenia dlatego, że symbol `%` zawiera w sobie znak dzielenia `/` (procent – 1/100 część).

Wykorzystanie pętli `while` do zamiany liczby dziesiętnej na binarną

Kolejne reszty z dzielenia liczby dziesiętnej przez 2 możemy wyznaczyć, wykorzystując pętlę `while`.

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     int liczba;
8.     cout << "Podaj liczbę: ";
9.     cin >> liczba;
10.
11.     while (liczba > 0)
12.     {
13.         cout << liczba % 2 << " ";
14.         liczba = liczba / 2;
15.     }
16. }
```

W 8. wierszu program prosi o podanie liczby dziesiętnej z klawiatury, a następnie zapisuje ją w zmiennej `liczba`. W wierszach 11.–15. powtarzamy w pętli wyznaczanie reszty z dzielenia liczby przez 2 i zapamiętywanie ilorazu całkowitego w zmiennej `liczba`.

Reszta z dzielenia jest za każdym razem wypisywana na ekranie. Kroki pętli są powtarzane, dopóki iloraz jest większy od zera.

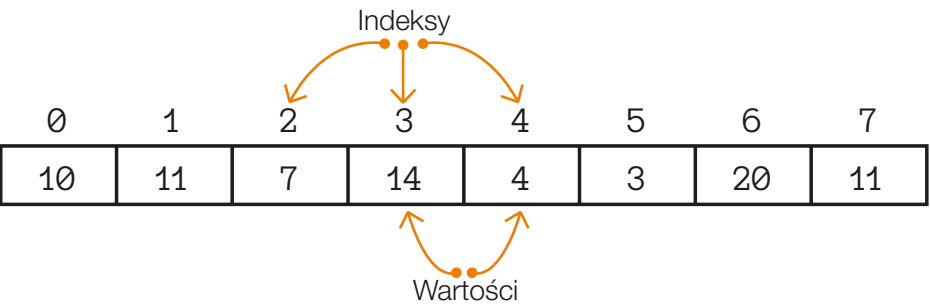
Powyższy program ma jedną wadę: reszty są wypisywane w kolejności ich wyznaczania, a powinny być zapisane w odwróconej kolejności. Rozwiązaniem tego problemu jest wprowadzenie nowej struktury danych do zapisania kolejnych reszt. Dzięki niej program będzie miał dostęp do zapamiętanych reszt i będzie mógł wyświetlić je w odwróconej kolejności.

Tablica – struktura danych

Aby zapisać w pamięci i przetwarzać w programie większą liczbę danych, np. liczby całkowite, bez konieczności tworzenia osobnej zmiennej dla każdej z nich, można posłużyć się strukturą danych nazywaną **tablicą** (ang. *array*).

Tablica jest uporządkowana, tzn. każda zapisana w niej wartość ma określone miejsce. Tablicę liczb można porównać do wiersza tabeli, w której w kolejnych komórkach są zapisane kolejne liczby. Każda

wartość zapisana w tablicy ma swoją współrzędną, zwaną **indeksem**. Tablice są indeksowane od 0 (rys. 11.10).



Rys. 11.10. Interpretacja graficzna tablicy o 8 wartościach liczbowych

Aby odwołać się do jakiegoś elementu tablicy, wystarczy podać jego współrzędną (indeks).

Instrukcje:

```
tablica[0] = 10;
tablica[7] = 11;
```

powodują przypisanie wartości 10 pierwszemu elementowi tablicy (o indeksie 0) i przypisanie wartości 11 ostatniemu elementowi (o indeksie 7).

Aby korzystać z tablicy, trzeba ją wcześniej zadeklarować, określając typ przechowywanych informacji. Między nawiasami wstawia się kolejne wartości, oddzielone przecinkami. Aby utworzyć tablicę o nazwie `tablica` zawierającą osiem konkretnych wartości całkowitych, należy napisać: `int tablica[8] = {1, 11, 7, 14, 4, 3, 20, 10};`

Warto wiedzieć

Przypisanie wartości do wybranego elementu tablicy oznacza zastąpienie wartości dotychczas przechowywanej nową wartością.

Tablice – wczytywanie i wypisywanie danych

Napiszemy program o nazwie *Cyfry wspak*, który po wczytaniu z klawiatury ośmiu cyfr (liczb jednocyfrowych) wyświetli je w odwróconej kolejności.

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     int tablica[8] = {};
8.     for (int i=0; i<=7; i++)
9.     {
10.         cin >> tablica[i];
11.     }
12.     for (int i=7; i>=0; i--)
13.     {
14.         cout << tablica[i] << " ";
15.     }
16. }
```

Kod źródłowy programu *Cyfry wspak*

Kod w wierszu 7. pozwala na utworzenie tablicy składającej się z ośmiu elementów typu `int`. W nawiasach klamrowych można od razu zainicjować kolejne elementy tablicy, oddzielając je od siebie przecinkiem. Jeśli tego nie zrobimy, program domyślnie uzupełni tablicę zerami. W dalszej części programu będziemy zamieniać te wartości (zera w tablicy) na wartości, które wprowadzi użytkownik. W wierszach 8.–11. zapisano kod pętli, która służy do wprowadzenia docelowych ośmiu liczb z klawiatury: kolejne liczby będą zapisywane na liście w miejsce zer jako elementy o indeksach: 0, 1, 2, ..., 7.

Wykorzystaliśmy tutaj **pętlę `for`**, która w przeciwieństwie do pętli `while` nie czeka na spełnienie określonego warunku, ale ma z góry ustaloną liczbę powtórzeń. W naszym kodzie ta liczba jest określona w nagłówku pętli, czyli poprzez trzy instrukcje: `int i=0; i<=7; i++;`, dzięki którym liczba powtórzeń pętli jest powiązana ze zmianą wartości zmiennej `i` od 0 aż do 7. Tę pomocniczą zmienną nazywa się **licznikiem pętli**.

Do wskazania położenia liczby w tablicy użyto zmiennej pomocniczej `i`, trzeba też usunąć przecinek. Ponieważ liczby mają być przez program wypisane w odwróconej kolejności (od elementu z indeksem 7 do elementu z indeksem 0), w pętli z wierszy 12.–15. nagłówek pętli zapisano tak, by wartość licznika pętli zmieniała się od 7 do 0. Taki zapis umożliwia dostęp do elementów listy w kolejności od ostatniego.

Dobra rada

Tworząc tablicę, możesz zainicjować część jej elementów. Zapis `tablica[8] = {1, 2}` utworzy tablicę o ośmiu elementach, z czego pierwszy będzie miał wartość 1, drugi 2, a kolejne będą zerami.

Pętla `for`

Ćwiczenie 5

- Przepisz kod programu *Cyfry wspak* i sprawdź jego działanie dla różnych zestawów danych.
- Przyjmij, że liczby na liście od razu powinny być zapamiętane w kolejności odwróconej. Napisz program realizujący takie założenie. Kod programu zapisz w pliku *cyfry_wprost.cpp*.

Program Zamiana systemów liczbowych

Teraz możemy wykorzystać tablice do napisania programu *Zamiana*, realizującego algorytm zamiany liczby dziesiętnej na liczbę dwójkową. Nasz program będzie mógł zapisać liczbę mieszczącą się w 1 bajcie pamięci. Wynika to z faktu, że tablica będzie miała z góry zadeklarowaną wielkość 8 elementów.

Oto kod programu znajdujący się w pliku *zamiana.cpp*:

Kod źródłowy programu
Zamiana

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     int tablica[8] = {0};
8.     int liczba;
9.     cout << "Podaj liczbę: ";
10.    cin >> liczba;
11.    int iloraz = liczba;
12.
13.    int i=0;
14.    while (iloraz > 0)
15.    {
16.        tablica[i] = iloraz % 2;
17.        iloraz = iloraz / 2;
18.        i++;
19.    }
20.    cout << "Zapis binarny " << liczba << " to: ";
21.    for (int i=7; i>=0; i--)
22.    {
23.        cout << tablica[i] << " ";
24.    }
25. }
```

W wierszu 9. program prosi o podanie liczby do zamiany i w wierszu 10. zapisuje ją w zmiennej *liczba*. W wierszu 11. tę samą liczbę przypisuje zmiennej *iloraz*, której wartość będzie się zmieniała w pętli. W wierszach 14.–19. za pomocą pętli *while* do listy są dodawane kolejne reszty z dzielenia liczby przez 2. Z kolei w zmiennej *iloraz* jest zapisywany iloraz całkowity z tego dzielenia.

W wierszach 21.–24. podana wcześniej liczba w systemie dziesiętnym jest wypisywana na ekranie w postaci binarnej. Do wypisania kolejnych elementów tablicy jest wykorzystywana pętla *for*. Przykładowe wykonanie programu widoczne jest na rysunku 11.11.

```
Podaj liczbę: 131
Zapis binarny 131 to: 1 0 0 0 0 0 1 1
```

Rys. 11.11. Działanie programu *Zamiana systemów liczbowych*

Ćwiczenie 6

- Zapisz kod źródłowy programu *zamiana.cpp* i uruchom program dla różnych liczb (mniejszych niż 256) wpisanych z klawiatury. Zaczynaj od sprawdzenia liczby 123.
- Zmień kod źródłowy programu *zamiana.cpp* tak, aby program pozwalał zapisywać liczby za pomocą dwóch bajtów (16 bitów).

💡 Zapamiętaj

Tablica to struktura programistyczna służąca do przechowywania wielu danych w uporządkowany sposób. Dane są przechowywane w ponumerowanych (indeksowanych) komórkach pamięci. W tablicy *n*-elementowej pierwszy element ma indeks 0, a ostatni *n* – 1.

 Zadania

- * **1** Najprostszym systemem liczbowym jest kod jedynekowy (unarny). Poszukaj informacji na temat tego systemu liczbowego. Podaj przykład jego praktycznego zastosowania w życiu codziennym.
- * **2** Napisz program, który będzie określać parzystość liczby wprowadzonej z klawiatury. W przypadku liczby parzystej powinien się wyświetlić komunikat „Liczba jest parzysta”, natomiast w przypadku liczby nieparzystej – komunikat „Liczba jest nieparzysta”.
- * **3** Napisz program, który zapisze w tablicy dziesięć kolejnych liczb nieparzystych, począwszy od 1, i wyświetli zawartość tablicy.
- * **4** Napisz program, który wczyta cztery liczby z klawiatury i wypisze je w odwróconej kolejności.
- * **5** Napisz program, który dla podanych dwóch liczb dwucyfrowych wypisze ich postać binarną, a następnie postać binarną ich sumy.
- ** **6** Napisz program, który dla liczby wprowadzonej z klawiatury wyznaczy największą potęgę liczby 2, nie większą od tej liczby.
- ** **7** Napisz program, który wczyta 8 liczb do tablicy, a następnie każdą z nich zmniejszy o 1.
- ** **8** Napisz program, który wczyta liczbę z klawiatury i wyznaczy jej dzielniki mniejsze od tej liczby. Na przykład po wczytaniu liczby 18 na ekranie powinny zostać wypisane liczby 1, 2, 3, 6 i 9.
- ** **9** W słynnym komputerze ENIAC stosowano kodowanie oparte na systemie dwupiętkowym, znane wcześniej z japońskich liczydeł, tzw. sorobanów. Przygotuj krótką prezentację lub artykuł na stronę internetową szkoły na temat tego systemu liczbowego i jego zastosowania do kodowania liczb w komputerze ENIAC.
- ** **10** Napisz program, który dla liczby wprowadzonej z klawiatury wypisze po przecinku wszystkie jej dzielniki jednocyfrowe (łącznie z liczbą 1).
- ** **11** Napisz program odwracający kolejność cyfr w liczbie całkowitej. Program może zapamiętywać cyfry w strukturze tablicy.
- *** **12** Napisz program, który będzie określać parzystość liczby wprowadzonej z klawiatury. Program nie powinien korzystać z operatora %.
- *** **13** Napisz program tworzący zestawienie liczb binarnych odpowiadających liczbom dziesiętnym od 0 do 255. Efektem działania programu powinny być dwie kolumny: po lewej – liczby dziesiętne, a po prawej – liczby binarne.
- *** **14** Napisz program odzwierciedlający kolejność cyfr w liczbie całkowitej. Program nie może zapamiętywać cyfr w strukturze tablicy.

12 **Liczby pierwsze**

Liczby pierwsze znasz z lekcji matematyki w szkole podstawowej. Choć odkryto je już w starożytności, przez lata twierdzenia dotyczące liczb pierwszych uchodziły wśród matematyków za piękne, ale bezużyteczne. Dopiero w drugiej połowie XX w. znalazły praktyczne zastosowanie w informatyce, zwłaszcza w kryptografii. Te twierdzenia są podstawą działania niektórych protokołów szyfrowania, czyli algorytmów, które zapewniają poufność komunikacji w Internecie i bezpieczeństwo transakcji bankowych. W tym temacie dowiesz się więcej o liczbach pierwszych i poznasz ich znaczenie.

Ważne w tym temacie

- Liczby złożone i liczby pierwsze
- Podzielność liczb całkowitych
- Algorytmy sprawdzające, czy liczba jest pierwsza
- Funkcje w języku C++
- Funkcja `sqrt` z biblioteki `cmath`

12.1. Liczby złożone i liczby pierwsze

Dodatknie liczby naturalne, które można przedstawić w postaci iloczynu liczb naturalnych mniejszych od nich, nazywamy **liczbami złożonymi**.  **Liczby złożone** Ich przykładami są: $9 = 3 \cdot 3$, $10 = 2 \cdot 5$, $120 = 2 \cdot 3 \cdot 4 \cdot 5$, $1115 = 3 \cdot 5 \cdot 7 \cdot 11$.

Pozostałe liczby, z wyjątkiem liczby 1, są liczbami pierwszymi. **Liczbą pierwszą** nazywamy więc liczbę większą od 1, która jest podzielna wyłącznie przez 1 i przez samą siebie. Wśród liczb mniejszych od 100 jest dwadzieścia pięć liczb pierwszych: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97. Na rysunku 12.1 oznaczono je pomarańczowym tłem.

Jedno z najważniejszych twierdzeń matematycznych dotyczących liczb pierwszych nazywane jest **podstawowym twierdzeniem arytmetyki**  **Podstawowe twierdzenie arytmetyki** i brzmi następująco: każda liczba naturalna (większa od 1) jest albo liczbą pierwszą, albo iloczynem liczb pierwszych.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	

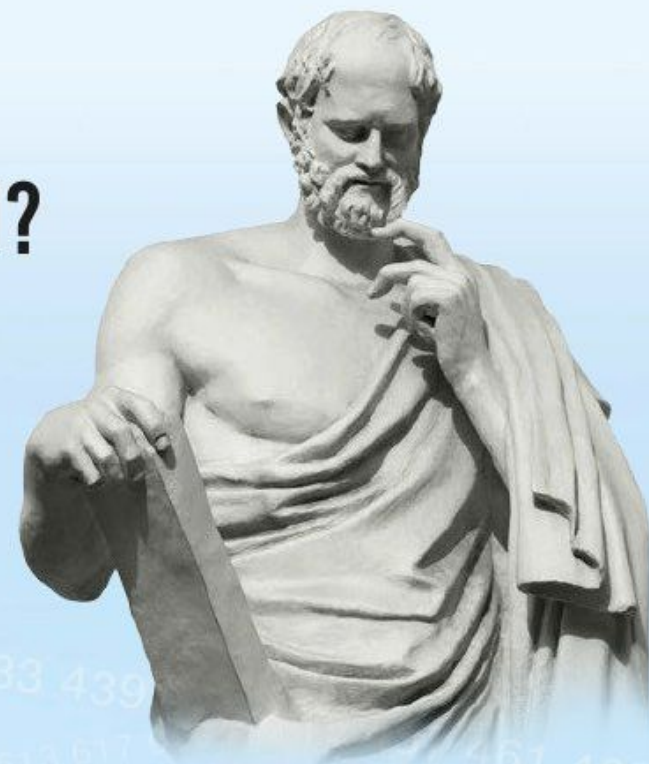
Rys. 12.1. Liczby złożone i liczby pierwsze mniejsze niż 100

 **Liczba pierwsza**

 Warto wiedzieć

Liczba 1 jest wyjątkowa. Przez setki lat uważano ją za liczbę pierwszą. Dziś przyjmuje się, że nie jest ani pierwsza, ani złożona.

CO WARTO WIEDZIEĆ O LICZBACH PIERWSZYCH?



Ile jest liczb pierwszych?

Liczyby pierwsze odkryto i badano już w starożytności. W IV w. p.n.e. Euklides udowodnił, że jest ich nieskończenie wiele. Dotąd nie odkryto jednak wzoru, według którego można byłoby dla konkretnej liczby n wyznaczyć n -tą liczbę pierwszą.

2 to najmniejsza i jedyna parzysta liczba pierwsza.

Liczyby postaci $2^n - 1$, gdzie n jest liczbą naturalną, nazywamy liczbami Mersenne'a. Wiele z nich jest liczbami pierwszymi.

Liczyby pierwsze większe od 5 kończą się cyframi 1, 3, 7, 9.



Liczyby czworaczce to cztery liczby pierwsze mające postać: $p, p+2, p+6, p+8$.

Liczyby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.

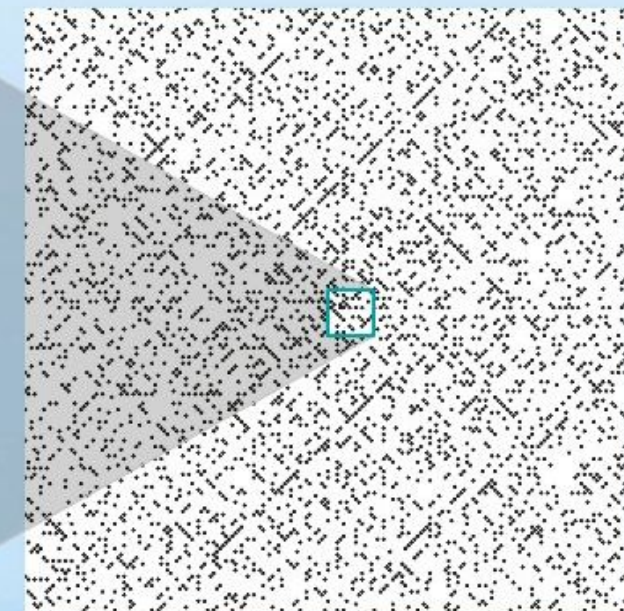
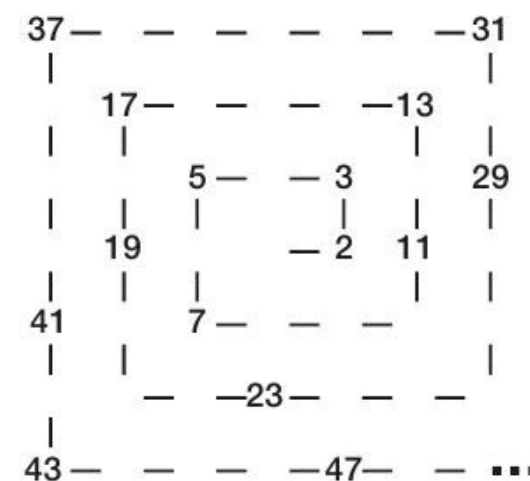
Liczyby pierwsze w kryptografii

Liczbę utworzoną z iloczynu dwóch dużych liczb pierwszych bardzo trudno w rozsądnym czasie rozłożyć na czynniki pierwsze, nawet z użyciem najszybszych komputerów. Wykorzystali to twórcy algorytmu RSA, który stosuje się do szyfrowania komunikacji w Internecie oraz w bankowości elektronicznej. Liczyby pierwsze używane w tym algorytmie mają nawet 2048 bitów, czyli składają się z 2048 cyfr binarnych.



Spirala Ulama

W 1963 r. polski matematyk Stanisław Ulam przedstawił graficznie rozmieszczenie liczb pierwszych. Na kwadratowej tablicy spiralnie wypisał kolejne liczby naturalne. Po naniesieniu bardzo wielu liczb okazało się, że na niektórych liniach liczby pierwsze grupują się częściej niż na innych. Do tej pory nie wyjaśniono, dlaczego tak się dzieje.



Jak liczby pierwsze pomagają rozpoznawać anagramy?

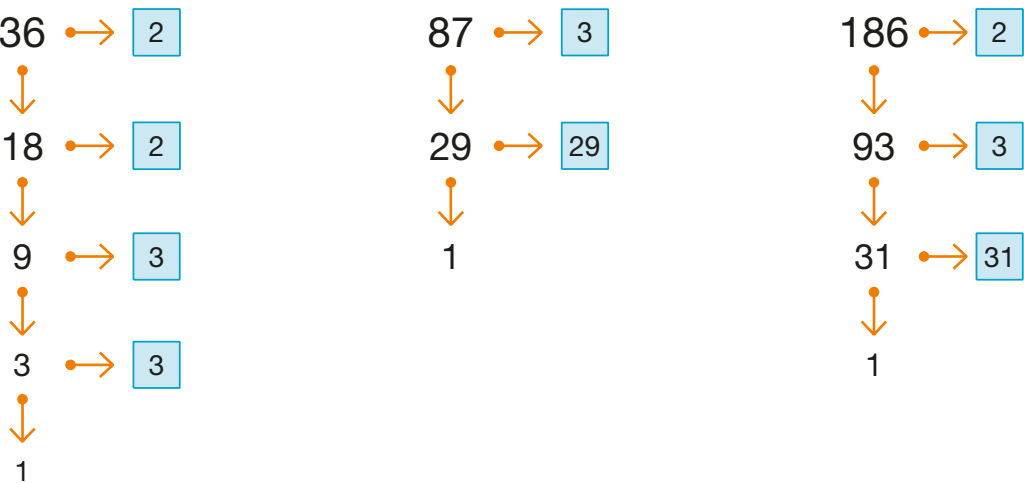
Rozkład dowolnej liczby na czynniki pierwsze jest jednoznaczny (np. dla 30 jest to wyłącznie rozkład $2 \cdot 3 \cdot 5$). Ta własność jest podstawą działania pewnego algorytmu sprawdzającego, czy dwa wyrazy są anagramami (składają się z tych samych liter). Kolejnym literom alfabetu przypisuje się kolejne liczby pierwsze. Jeśli iloczyny liczb odpowiadających literom w wyrazach są równe, wyrazy są anagramami.

$$\begin{array}{ccc} 3 & 2 & 5 \\ | & | & | \\ \text{b} & \text{a} & \text{c} \end{array} \quad 3 \cdot 2 \cdot 5 = 30$$

$$\begin{array}{ccc} 2 & 3 & 5 \\ | & | & | \\ \text{a} & \text{b} & \text{c} \end{array} \quad 2 \cdot 3 \cdot 5 = 30$$

Rozkład na czynniki pierwsze

Iloczyn liczb pierwszych, który tworzy daną liczbę, nazywamy **rozkładem na czynniki pierwsze**. Jest on jedyny w swoim rodzaju, tzn. każda liczba ma tylko jedną unikatową formułę rozkładu, która do niej przynależy. Na rysunku 12.2 przedstawiono rozkład na czynniki pierwsze liczb 36, 87 i 186. Te liczby możemy zapisać następująco: $36 = 2 \cdot 2 \cdot 3 \cdot 3$; $87 = 3 \cdot 29$; $186 = 2 \cdot 3 \cdot 31$.



Rys. 12.2. Rozkład liczb 36, 87 i 186 na czynniki pierwsze

Warto wiedzieć

Stosunkowo łatwo sprawdzić, czy podana liczba nieparzysta jest podzielna przez 5, ponieważ kończy się ona cyfrą 5.

Okazuje się, że im większa liczba, tym trudniej sprawdzić, czy jest pierwsza czy złożona, a więc czy da się ją rozłożyć na czynniki pierwsze. W przypadku dużych liczb rozstrzygnięcie tego w rozsądnym czasie stanowi zadanie praktycznie niewykonalne, nawet dla współczesnych komputerów. Ta trudność jest podstawą bezpieczeństwa działania niektórych algorytmów kryptograficznych (np. algorytmu RSA).

Liczba 2 jest jedyną parzystą liczbą pierwszą, więc w praktyce poszukiwania liczb pierwszych wymagają sprawdzania tylko liczb nieparzystych.

A to ciekawe

Szybki test pierwszości

Dla bardzo dużych liczb bardzo trudno ustalić, czy są one pierwsze czy złożone. Powszechnie stosowanym i w praktyce najszybszym testem pierwszości jest test Millera–Rabina. Algorytm stworzył w 1976 r. amerykański profesor informatyki Gary Miller, a w 1980 r. ulepszył go urodzony we Wrocławiu Michael O. Rabin (na zdjęciu). Dzięki temu algorytmowi można przeprowadzić test pierwszości nawet dla ogromnych liczb, rzędu 150 cyfr. Ponieważ jest to test probabilistyczny, nie wskazuje on żadnych dzielników testowanej liczby. Oznacza to, że liczba może zostać błędnie rozpoznana jako pierwsza. Prawdopodobieństwo takiego zdarzenia jest jednak znikome – wynosi ok. 1 do 10^{12} .



12.2. Podzielność liczb

Zanim stworzymy program sprawdzający, czy podana liczba jest pierwsza, napiszemy prostsze programy związane z podzielnością. Wprowadzimy przy tej okazji nową konstrukcję programistyczną – funkcje.

Funkcje w języku C++

Aby rozwiązać bardziej złożony problem algorytmiczny, najlepiej rozłożyć go na prostsze podproblemy. W praktyce oznacza to wyodrębnianie fragmentów programów (podprogramów), które w języku C++ nazywa się **funkcjami**.

Każda funkcja musi mieć nadaną nazwę. Po nazwie zawsze występują nawiasy okrągłe, w których podaje się **parametry funkcji**.

Funkcja

Parametr funkcji

```
typ nazwa_funkcji(typ parametr1, typ parametr2, ...)
{
    //blok instrukcji tworzących treść funkcji
}
```

Definicja funkcji składa się z nagłówka funkcji i bloku instrukcji, które stanowią jej treść. Nagłówek zawiera typ wartości zwracanej przez funkcję, nazwę funkcji oraz listę parametrów.

Efekt działania funkcji albo wyświetla się na ekranie, albo za pomocą **instrukcji return** jest przekazywany (zwracany) z powrotem do instrukcji programu, która funkcję wywołała.

Instrukcja return

Dzięki konsekwentnemu stosowaniu funkcji stworzymy kod bardziej przejrzysty i łatwiejszy do modyfikowania w przyszłości. Unikamy także powtórzeń, ponieważ tę samą funkcję można wywołać wiele razy dla różnych parametrów.

Badanie podzielności przez 3

Aby zobaczyć, jak w praktyce korzysta się z funkcji, napiszemy program, który będzie sprawdzał, czy podana przez użytkownika liczba jest podzielna przez 3.

Sprawdzenie podzielności danej liczby przez 3 sprowadza się do ustalenia, czy reszta z dzielenia tej liczby przez 3 jest równa zero.

Oto specyfikacja problemu:

Specyfikacja

Dane: dwie liczby naturalne a i b .

Wynik: reszta z dzielenia liczby a przez liczbę b .

Stworzymy funkcję o nazwie `ResztaZDzielenia`, zwracającą resztę z dzielenia liczby a przez liczbę b , a wywołamy ją dla podanej przez użytkownika liczby i b równego 3.

Poniżej znajduje się kod źródłowy programu.

Kod źródłowy programu sprawdzającego podzielność liczby przez 3

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int ResztaZDzielenia(int a, int b)
6. {
7.     return a % b;
8. }
9. int main()
10. {
11.     int liczba;
12.     cout << "Podaj liczbe: "; cin >> liczba;
13.     cout << "Reszta z dzielenia " << liczba;
14.     cout << " przez 3 wynosi: ";
15.     cout << ResztaZDzielenia(liczba, 3) << endl;
16. }
```

Dobra rada

Staraj się tworzyć funkcje o możliwie szerokim zastosowaniu. Dzięki temu będzie można je wywołać dla różnych parametrów. Tak tworzone są również funkcje w arkuszach kalkulacyjnych.

W linii 15. znajduje się wywołanie funkcji ResztaZDzielenia przy użyciu parametrów liczba i 3. W ten sposób sprawdzimy podzielność podanej liczby przez 3. Definicja tej funkcji znajduje się w wierszach 5.–8., czyli przed kodem funkcji main. W wierszu 5. przed nazwą funkcji umieszczono zapis int, który określa typ wartości zwracanej przez funkcję. W nawiasach znajduje się lista parametrów funkcji oraz informacja o ich typie. Te parametry służą do przekazywania wartości z programu głównego (odpowiednio: wartość zmiennej liczba oraz liczbę 3). Wiersz 7. to treść funkcji, która wyznacza resztę z dzielenia (z wykorzystaniem operatora %) i zwraca rezultat, stąd słowo kluczowe return.

Zauważ, że tak zdefiniowana funkcja jest ogólna i może być zastosowana do sprawdzania podzielności dowolnych liczb.

Na rysunku 12.3 przedstawiono przykładowe wywołanie programu wyznaczającego resztę z dzielenia przez 3.



Rys. 12.3. Wywołanie programu sprawdzającego podzielność przez 3

Parametry określone w momencie definiowania nowej funkcji nazywa się parametrami formalnymi. Natomiast parametry, dla których funkcja wywołana jest w programie głównym, to parametry aktualne. Parametr aktualny przekazuje więc faktyczną wartość (np. podaną przez użytkownika) do funkcji, która została wcześniej zdefiniowana.

Parametry formalny i aktualny funkcji

Ćwiczenie 1

- a. Przepisz kod źródłowy programu sprawdzającego podzielność liczby przez 3 i sprawdź jego działanie na różnych przykładach.
- b. Zmodyfikuj treść programu tak, aby sprawdzał on podzielność podanej liczby przez 17.

Badanie podzielności przez kilka liczb jednocześnie

Zmodyfikujemy kod naszego programu tak, aby wyznaczał reszty z dzielenia podanej liczby przez cztery kolejne liczby pierwsze: 2, 3, 5, 7.

Wynikiem działania programu powinny być cztery liczby: reszta z dzielenia przez 2, reszta z dzielenia przez 3, reszta z dzielenia przez 5 i reszta z dzielenia przez 7.

Dzięki temu, że zdefiniowaliśmy wcześniej funkcję ResztaZDzielenia, modyfikacja programu polega wyłącznie na dopisaniu do programu głównego odpowiednich wywołań funkcji.

Oto kod źródłowy funkcji ResztaZDzielenia i jej wywołanie w funkcji main:

```
1. int ResztaZDzielenia(int a, int b)
2. {
3.     return a % b;
4. }
5.
6. int main()
7. {
8.     int liczba;
9.     cout << "Podaj liczbe: "; cin >> liczba;
10.    cout << "Reszta z dzielenia liczby " << liczba;
11.    cout << " przez 2, 3, 5, 7: ";
12.    cout << "wynosi odpowiednio:" << endl;
13.    cout << ResztaZDzielenia(liczba, 2) << endl;
14.    cout << ResztaZDzielenia(liczba, 3) << endl;
15.    cout << ResztaZDzielenia(liczba, 5) << endl;
16.    cout << ResztaZDzielenia(liczba, 7) << endl;
17. }
```

Fragment kodu źródłowego programu obliczającego reszty z dzielenia przez 2, 3, 5, 7

Ćwiczenie 2

Zapisz kod źródłowy programu obliczającego reszty z dzielenia przez 2, 3, 5, 7 i sprawdź jego działanie dla różnych liczb. Instrukcję z linii 1.–4. zapisz w jednej linii.

 Zapamiętaj

Funkcje definiujemy, gdy chcemy nazwać pewną grupę instrukcji i oddzielić fragment kodu od reszty programu – zwykle po to, aby go wielokrotnie wykorzystać.

12.3. Test pierwszości liczby

Matematycy od wieków zastanawiali się, jak sprawdzić, czy liczba jest pierwsza. Dziś naukowcy szukają przede wszystkim szybkich algorytmów, które będą to rozstrzygać.

Test pierwszości – najprostszy algorytm

Aby stwierdzić, czy jakaś liczba n jest pierwsza czy złożona, należy sprawdzić, czy ma ona jakikolwiek dzielnik różny od 1 i od niej samej. Można to zrobić, dzieląc n kolejno przez 2, przez 3 itd., aż do $n - 1$.

Przed napisaniem programu komputerowego zapiszemy specyfikację.

Specyfikacja

Dane: liczba naturalna większa od 1.

Wynik: komunikat „Liczba pierwsza”, jeśli liczba jest liczbą pierwszą, komunikat „Liczba złożona” – w przeciwnym wypadku.

Poniżej znajduje się kod źródłowy funkcji CzyPierwsza i jej wywołanie w funkcji main.

Fragment kodu
źródłowego programu
Test pierwszości

```
1. bool CzyPierwsza(int n)
2. {
3.     for (int i=2; i<n; i++)
4.         if (n%i==0) return false;
5.     return true;
6. }
7.
8. int main()
9. {
10.    int liczba;
11.    cout << "Podaj liczbę: "; cin >> liczba;
12.    if (CzyPierwsza(liczba))
13.        cout << "Liczba pierwsza" << endl;
14.    else
15.        cout << "Liczba złożona" << endl;
16. }
```

W liniach 1.–6. znajduje się definicja funkcji CzyPierwsza, która jest funkcją typu `bool`, tzn. wynikiem jej działania będzie wartość typu logicznego. Zadaniem funkcji jest sprawdzenie podzielności podanej liczby przez liczby od 2 do $n - 1$. Gdy zostanie znaleziony dzielnik, wynikiem funkcji (instrukcja `return`) jest wartość `false` (fałsz). Wówczas funkcja kończy swoje działanie i nie wykonuje kolejnych dzielen. Jeśli nie zostanie znaleziony dzielnik, to wynikiem funkcji jest wartość `true` (prawda). Funkcja kończy wtedy swoje działanie (linia 5.).

W programie głównym w linii 12. wywoływana jest funkcja CzyPierwsza. Argumentem tej funkcji jest zmienna liczba, której wartość użytkownik wpisuje wcześniej z klawiatury. W zależności od wartości zwróconej przez funkcję program wyświetla komunikat, czy liczba jest pierwsza czy złożona.

Ćwiczenie 3

- a. Przepisz kod źródłowy programu i sprawdź jego działanie.
- b. Przygotuj tabelę, w której rozpiszesz kolejne etapy działania funkcji CzyPierwsza dla $n = 9$ oraz $n = 11$.

Program *Test pierwszości*, choć daje poprawne wyniki, wykonuje wiele niepotrzebnych operacji wyznaczania reszt z dzielenia. Liczbę tych operacji możemy stosunkowo łatwo ograniczyć.

Test pierwszości – ulepszony algorytm

Sprawdzanie podzielności podanej liczby przez liczby parzyste większe od 2 nie ma sensu. Wystarczy więc zbadać podzielność przez 2, a potem sprawdzać wyłącznie liczby nieparzyste.

Liczbę wykonywanych dzielen można ograniczyć jeszcze bardziej. Oto rozkłady liczb 36, 100 i 42 na czynniki:

$36 = 2 \cdot 18$	$100 = 2 \cdot 50$	$42 = 2 \cdot 21$
$36 = 3 \cdot 12$	$100 = 4 \cdot 25$	$42 = 3 \cdot 14$
$36 = 4 \cdot 9$	$100 = 5 \cdot 20$	$42 = 6 \cdot 7$
$36 = 6 \cdot 6$	$100 = 10 \cdot 10$	

Powyższe przykłady pokazują, że gdy liczbę złożoną n przedstawiamy w postaci iloczynu dwóch czynników większych od 1, to przynajmniej jeden z tych czynników musi być mniejszy lub równy $\sqrt{n}$.

Dla liczby 36 są to odpowiednio liczby 2, 3, 4 i 6, przy czym $6 = \sqrt{36}$. Podobnie dla pozostałych liczb – mniejszy z dzielników liczby nie jest większy od pierwiastka z tej liczby. Zatem najmniejszy dzielnik właściwy liczby złożonej n musi być mniejszy bądź równy $\sqrt{n}$.

 Warto wiedzieć

W latach 90. XX w. pewien producent procesorów był zmuszony wymienić jeden z modeli, ponieważ wykonywał on operacje dzielenia z mniejszą dokładnością, niż zakładano. Wynosiła ona 9 zamiast 19 miejsc po przecinku. Ta wada ujawniła się przy szacowaniu stałej Bruna, która dotyczy liczb bliźniaczych.

 Warto wiedzieć

Dzielnik właściwy liczby to każdy dzielnik mniejszy od tej liczby.

Operacja pierwiastkowania **Operacja pierwiastkowania** nie jest standardowym działaniem matematycznym w języku C++, dlatego w kodzie musimy dołączyć bibliotekę `cmath`. Zawiera ona zdefiniowane funkcje matematyczne, m.in. trygonometryczne.

Kod źródłowy programu
Ulepszony test pierwszośc

```
1. #include <iostream>
2. #include <cmath>
3.
4. using namespace std;
5.
6. bool CzyPierwsza(int n)
7. {
8.     if (n==2) return true;
9.     if (n%2==0) return false;
10.    int pierwiastek = sqrt(n);
11.    for (int i=3; i<=pierwiastek; i=i+2)
12.        if (n%i==0) return false;
13.    return true;
14. }
15.
16. int main()
17. {
18.     int liczba;
19.     cout << "Podaj liczbe: "; cin >> liczba;
20.     if (CzyPierwsza(liczba))
21.         cout << "Liczba pierwsza" << endl;
22.     else
23.         cout << "Liczba zlozona" << endl;
24. }
```

Ulepszenia wprowadzono w definicji funkcji `CzyPierwsza`. W liniach 8.–9. nadajemy status liczby pierwszej liczbie 2 (instrukcja `return true`) oraz status liczby złożonej wszystkim liczbom podzielny przez 2 większym od 2 (instrukcja `return false`). W linii 10. wprowadzono zmienną `pierwiastek`, której wartość stanowi przybliżenie n otrzymane dzięki funkcji `sqrt` znajdującej się w bibliotece `cmath` (linia 2.). W pętli modyfikujemy liczbę wykonań – ograniczamy ją do n oraz testujemy tylko liczby nieparzyste (instrukcja `i=i+2`).

Ćwiczenie 4

a. Przepisz kod źródłowy programu i sprawdź jego działanie.

b. Przygotuj tabelę, w której rozpiszesz kolejne etapy działania funkcji `CzyPierwsza` dla $n = 9$ oraz $n = 11$.

 Zadania

- * 1 Napisz program wyznaczający wszystkie dzielniki liczby 80.
- * 2 Napisz program znajdujący najmniejszą i największą trzycyfrową liczbę pierwszą.
- * 3 Napisz funkcję, która będzie badała podzielność podanej liczby jednocześnie przez 3 i 7. W teście użyj pojedynczej instrukcji logicznej (bez operatora `and`).
- * 4 Napisz program, który dla dwóch podanych przez użytkownika liczb wyświetli informację o tym, czy ich suma jest liczbą pierwszą.
- ** 5 Napisz program wyświetlający liczby, których pierwiastki mieszczą się w podanym przez użytkownika przedziale. Na przykład dla przedziału [5; 6] są to liczby: 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36. W kodzie wykorzystaj moduł `math` i funkcję `sqrt`. Sprawdź działanie programu dla przedziału [14; 19].
- ** 6 Stwórz program, który wyznaczy wszystkie trzycyfrowe liczby pierwsze.
- ** 7 Przygotuj w arkuszu kalkulacyjnym skoroszyt, który pozwoli rozstrzygnąć, czy prawdziwe jest następujące twierdzenie: każda liczba złożona mniejsza od 100 dzieli się przynajmniej przez jedną z liczb: 2, 3, 5, 7. Skorzystaj z funkcji `MOD`.
- ** 8 Napisz program wyznaczający wszystkie dzielniki pierwsze liczby 80.
- ** 9 Napisz w języku C++ prostą grę, w której dwóch użytkowników podaje po jednej liczbie sześciocyfrowej. Wygrywa ten, kto wskazał liczbę, która ma więcej dzielników właściwych.
- *** 10 Znajdź pary dwucyfrowych i trzycyfrowych liczb bliźniaczych. Uwaga: Liczby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.
- *** 11 Liczba 362 881 ma tę własność, że przy dzieleniu przez 2, 3, 4, 5, 6, 7, 8, 9 daje resztę 1. Znajdź najmniejszą liczbę o tej własności.
- *** 12 Napisz program, który wyznaczy najmniejszą z trzycyfrowych liczb doskonałych. Liczba doskonała to liczba równa sumie swych dzielników różnych od niej samej, np. $28 = 1 + 2 + 4 + 7 + 14$.
- *** 13 Napisz program, który wypisze wszystkie pary liczb bliźniaczych z przedziału [2; 3000]. Uwaga: Liczby bliźniacze to dwie liczby pierwsze, których różnica wynosi 2.

13 Sortowanie liczb i wyszukiwanie

Porządkowanie jest ważną częścią ludzkiej aktywności. Zwykle wysiłek w nie włożony się opłaca, bo później wiele zadań można wykonać szybciej i łatwiej. Jeśli np. szukamy pokoju w hotelu, to docenimy fakt, że pokoje są ponumerowane, a numery często zawierają informację o piętrze. W świecie oprogramowania porządkowanie odbywa się dzięki algorytmom sortowania. W tym temacie przyjrzymy im się bliżej.

Ważne w tym temacie

- Poszukiwanie największej liczby
- Porządkowanie danych
- Algorytmy wyszukiwania sekwencyjnego i binarnego
- Algorytmy sortowania bąbelkowego i przez wstawianie
- Tablice jako argumenty funkcji w języku C++

Możemy wyszukiwać dane w zbiorze nieuporządkowanych danych. Jeśli ten zbiór jest niewielki, dość szybko znajdziemy interesujący nas element. Ręczne przeszukiwanie większego zbioru wymagałoby od nas dużo więcej czasu. Wtedy warto wykorzystać do pomocy komputer. Znacznie szybciej od człowieka znajdzie on pożądaną przez nas informację nawet w ogromnej ilości danych

13.1. Szukamy największej liczby

Rozważmy następujący problem: trener skoku w dal mierzy podczas treningu długości kolejnych skoków zawodnika i wpisuje wyniki na bieżąco do programu komputerowego. Chcemy stworzyć program, który na zakończenie treningu wyświetli długość najdłuższego skoku. Zakładamy, że wyniki podawane są w metrach, z dokładnością do 0,01 m, czyli 1 cm.

Nie wiemy, ile skoków lekkoatleta wykona podczas treningu. Nawet jeśli trener zaplanuje jakąś ich liczbę, to i tak niektóre skoki będą prawdopodobnie spalone (nieważne) i nie będą brane pod uwagę w obliczeniach.



Rys. 13.1. Skok w dal

Warto wiedzieć

Określenie jednostek, w których są wyrażone różne wielkości, jest dla programisty bardzo ważne, ponieważ musi on odpowiednio wybrać typ danych. Na przykład jeśli dokładność pomiaru skoku w dal wynosi 1 cm, to wyniki podawane w centymetrach będą liczbami całkowitymi. Jeśli jednak wyniki podajemy w metrach, mogą być ułamkami.

Program komputerowy trzeba więc napisać tak, aby trener mógł poinformować komputer o tym, że trening został zakończony. Jak to zrobić? Jednym ze sposobów może być wpisanie przez trenera liczby 0, która zakończy ciąg podawanych liczb. Wtedy program komputerowy powinien wypisać długość najdłuższego skoku.

Specyfikacja

Dane: ciąg liczb nieujemnych, zapisanych z dokładnością do 0,01, zakończony liczbą 0.

Wynik: największa z podanych liczb.

Aby wykonać to zadanie, program musi porównać długości skoków. Komputer nie potrafi jednak jednocześnie porównywać wielu liczb, nawet gdyby miał je wszystkie zapisane w pamięci. Pracuje on sekwencyjnie i nie może jak człowiek objąć wzrokiem kilku liczb, aby wskazać największą. W jednej operacji może brać pod uwagę tylko dwie liczby. Pozostałe są niedostępne.

Na poniższym schemacie zaprezentowano, jaką strategię wyboru największej spośród czterech podanych liczb stosuje człowiek, a jaką – komputer.

Jak widzi komputer?

Zadanie:

Która z podanych liczb jest największa?

6,40 6,53 6,34 6,55



Człowiek

Przy małej liczbie danych człowiekowi wystarczy jedno spojrzenie, aby wskazać największą liczbę:

6,40 6,53 6,34 6,55



Komputer

Komputer, nawet przy małej liczbie danych, etapami porównuje kolejne pary liczb i wskazuje większą liczbę z pary:

Etap 1	6,40	6,53		
Etap 2		6,53	6,34	
Etap 3		6,53		6,55

Dobra rada

Ciąg potraktuj jako układ liczb, w którym występują one w określonej kolejności.

Warto wiedzieć

W żargonie programistów specjalną wartość (np. 0), która wskazuje koniec ciągu wczytywanych do programu wartości, nazywa się wartownikiem (ang. *sentinel*). Wartość ta musi być unikatowa, aby nie można było jej pomylić z wartościami podanymi przez użytkownika programu.

Aby program mógł wyznaczyć najdłuższy z podanych skoków, będziemy porównywać ich długości parami: porównamy pierwszy skok z drugim, dłuższy z nich porównamy z trzecim skokiem, dłuższy z nich z czwartym itd. Te operacje program będzie wykonywał do momentu, aż użytkownik wpisze liczbę 0.

Dzięki takiemu rozwiązaniu program będzie musiał zapamiętywać tylko dwie wartości: aktualnie podaną przez użytkownika i największą z dotychczas podanych. W naszej sytuacji pamiętanie pozostałych wartości nie jest potrzebne.

Lista kroków algorytmu wyznaczania największej liczby z ciągu liczb mogłaby wyglądać następująco:

 **Warto wiedzieć**

Procesor porównujący ze sobą dwie liczby używa sprzętowej instrukcji porównywania. Jej działanie polega na sprawdzeniu, czy różnica porównywanych liczb jest dodatnia, ujemna czy równa 0.

- 1 Jako wartość aktualnie największą (maksimum) przyjmij 0 i wczytaj pierwszą liczbę.
- 2 Dopóki ostatnia wczytana liczba jest różna od 0, wykonuj krok 3 i krok 4.
- 3 Jeśli ostatnia wczytana liczba jest większa od maksimum, to podstaw ją za maksimum.
- 4 Wczytaj kolejną liczbę z klawiatury.
- 5 Wypisz maksimum na ekranie.

Przeanalizujemy teraz działanie algorytmu na przykładowych danych. Wygodnie będzie zrobić to w tabeli.

 **Dobra rada**

Jeżeli trudno ci zrozumieć, jak działa algorytm, możesz utworzyć tabelę zawierającą pośrednie wartości zmiennych i warunków logicznych.

Wartość aktualnie największa (maksimum)	Wczytana liczba	Czy ostatnia wczytana liczba jest różna od 0?	Czy ostatnia wczytana liczba jest większa od aktualnego maksimum?
0	6,20	tak	tak
6,20	6,23	tak	tak
6,23	6,34	tak	tak
6,34	6,30	tak	nie
6,34	6,41	tak	tak
6,41	6,14	tak	nie
6,41	6,34	tak	nie
6,41	6,27	tak	nie
6,41	6,10	tak	nie
6,41	0	nie	–

Tabela 13.1. Działanie algorytmu wyznaczania największej liczby

Tabelę uzupełniamy wierszami od lewej do prawej, według kolejnych kroków algorytmu. Kolorem zielonym zaznaczono pewien etap analizy działania algorytmu: wczytano w tym momencie liczbę 6,41. Kolejnymi operacjami będą: sprawdzenie, czy liczba 6,41 jest różna od 0 (krok 2), a następnie – czy jest większa od ostatniej największej wartości równej 6,34 (krok 3). Ponieważ oba warunki są spełnione, liczba 6,41 staje się aktualnie największą. W kolejnych iteracjach podawane są liczby od niej mniejsze, dlatego w momencie podania liczby 0 liczba aktualnie największa staje się największą spośród wszystkich podanych liczb.

Ćwiczenie 1

Zastanówcie się w grupie, czy jeśli na liście kroków ze s. 192 zamienimy miejscami kroki 3 i 4, to lista nadal będzie przedstawiała poprawny algorytm. W jaki sposób można to sprawdzić?

Zauważ, że program wyznaczający największą liczbę z ciągu powinien powtarzać następujące czynności:

- 1. sprawdzać, czy liczba wprowadzona przez użytkownika jest liczbą 0, której podanie kończy działanie programu,
- 2. sprawdzać, czy wprowadzona liczba jest większa od aktualnie największej,
- 3. wczytywać kolejną wprowadzoną liczbę.

Możemy więc do rozwiązania tego zadania wykorzystać pętlę while, a kod źródłowy programu *Najdłuższy skok* może wyglądać następująco:

Pętla while
→ s. 162

```
1. #include <iostream>
2.
3. using namespace std;
4.
5. int main()
6. {
7.     float a, maks;
8.     maks = 0;
9.     cout << "Podaj kolejne dlugosci skokow" << endl;
10.    cin >> a;
11.    while (a!=0)
12.    {
13.        if (a>maks) maks = a;
14.        cin >> a;
15.    }
16.    cout << "\nNajdluzszy skok: " << maks << " m" << endl;
17. }
```

 **Kod źródłowy programu**
Najdłuższy skok

 **Warto wiedzieć**

Zmiennym zadeklarowanym w funkcji main kompilator nie przypisuje sam żadnej wartości. Domyślnie zmienna ma wartość przypadkową.

Warto wiedzieć

Od lewego ukośnika zapisuje się stałe znakowe znaków niegraficznych kodu ASCII (np. `\n` dla nowego wiersza, `\t` dla tabulacji poziomej) lub znaków specjalnych (np. `\"` dla znaku `"`, gdy chcemy go wyświetlić na ekranie).

Warto wiedzieć

Zapisywanie liczb ułamkowych z kropką zamiast przecinka jest zwyczajem anglosaskim. To dlatego na klawiaturze numerycznej znajduje się właśnie znak kropki.

Dobra rada

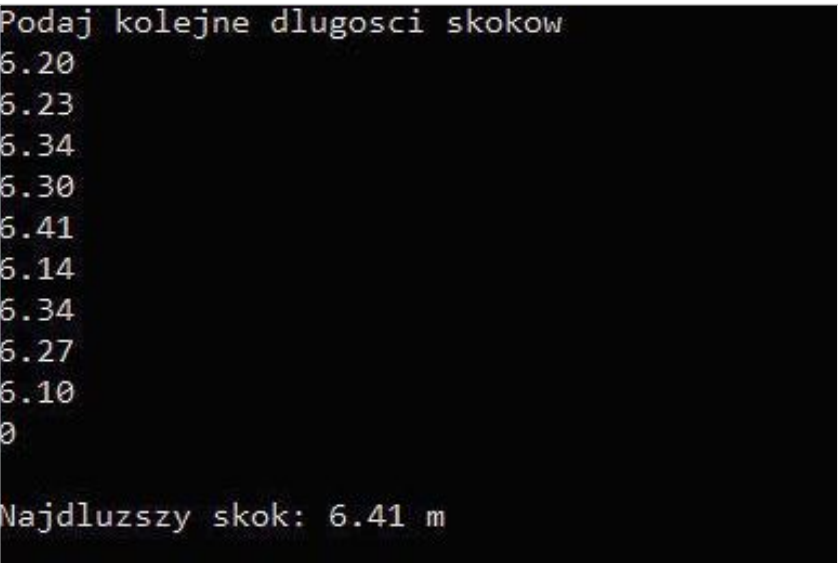
Aby przyspieszyć wprowadzanie wielu danych do programu, możesz wpisać je w jednym wierszu, oddzielając spacjami, np. `6.11 6.24 6 6.01 6.12 0`, a następnie nacisnąć Enter.

Funkcja float W wierszu 7. deklarujemy dwie zmienne typu `float` (`a` oraz `maks`), w których można zapisywać liczby całkowite i niecałkowite (z częścią ułamkową). W zmiennej `a` będzie przechowywana liczba jako ostatnia wczytana z klawiatury, a w zmiennej `maks` – aktualnie największa z wczytanych liczb. Na początku przypisujemy tej zmiennej wartość 0.

W wierszu 10. wczytujemy wartość zmiennej `a`. W wierszach 11.–15. pojawia się pętla `while`, która najpierw sprawdza, czy wartość zmiennej `a` jest różna od 0. Jeżeli tak, to program wykonuje grupę instrukcji w nawiasach klamrowych. Instrukcje wykonywane są tyle razy, ile razy użytkownik podał liczbę różną od zera.

W wierszu 13. sprawdzamy, czy podana przez użytkownika liczba jest większa od wartości zmiennej `maks`. Jeśli tak, to do zmiennej `maks` jest przypisywana aktualna liczba. Po zakończeniu wprowadzania liczb (czyli po wpisaniu przez użytkownika liczby 0) aktualnie największa liczba staje się liczbą największą dla tego zestawu danych. Warto zauważyć, że cała instrukcja `if` jest zapisana w jednym wierszu, ponieważ jest w niej wykonywana tylko jedna operacja. W linii 16. użyto zapisu `\n`, który odpowiada za przeniesienie kursora do kolejnego wiersza.

Okno programu *Najdłuższy skok* przedstawiono na rysunku 13.2. Zwróć uwagę na to, że w programie liczby ułamkowe podaje się z kropką zamiast przecinka.



Rys. 13.2. Wywołanie programu *Najdłuższy skok*

Ćwiczenie 2

Utwórz plik `najdluzszy_skok.cpp` i zapisz w nim kod źródłowy programu *Najdłuższy skok*. Sprawdź kod, skompiluj go i uruchom program kilkakrotnie dla różnych danych wpisanych z klawiatury.

13.2. Do czego służy sortowanie danych?

Sortowanie danych (ang. *sorting*), zwane też **porządkowaniem danych**, to ustawianie tych danych w zadanej kolejności, według określonego kryterium.

Zwykle gdy mówimy o sortowaniu liczb, mamy na myśli uporządkowanie ich w kolejności malejącej albo rosnącej. W rzeczywistości najczęściej jest to kolejność nierosnąca lub niemalejąca, ponieważ sortowane wartości mogą się powtarzać. Wyrazy z reguły ustawiamy w kolejności alfabetycznej (od A do Z) albo odwróconej kolejności alfabetycznej (od Z do A). Z kolei daty i godziny porządkuje się zwykle chronologicznie, czyli od najstarszych do najnowszych (ewentualnie w chronologii odwróconej – od najnowszych do najstarszych). Przykłady posortowanych danych są widoczne na rysunku 13.3.

2003	butelka	01.01.2001
2147	długopis	15.03.2005
2290	klucze	22.07.2010
2435	kubek	09.09.2012
2589	kurtka	30.04.2015
2741	laptop	18.08.2017
2896	ładowarka	05.11.2018
3050	notatnik	14.02.2020
3204	okulary	23.06.2021
3358	parasol	11.10.2022
3512	plecak	07.05.2023
3667	portfel	19.12.2024
3821	słuchawki	03.02.2026

Rys. 13.3. Posortowane liczby, wyrazy i daty

Uporządkowane informacje, zwłaszcza obszerne zbiory danych, o wiele łatwiej czytać, analizować oraz przeszukiwać.

Rysunek 13.4 przedstawia przykładową tabliczkę przystankową. Zauważ, że dane są uporządkowane na dwóch poziomach: kierunki – alfabetycznie, a godziny dla każdego kierunku – chronologicznie.

LGOTA WIELKA				
ROZKŁAD ODJAZDÓW				
Kierunek	Przez	Godziny odjazdu		
BOGUMIŁOWICE	SULMIERZYCE	14:50 S		
ŁÓDŹ	BEŁCHATÓW	6:58 dnw	12:08 dnw	17:03 dnw
PAJĘCZNO	SULMIERZYCE	09:20 Dm		
RADOMSKO		05:11 6s	07:01 S	07:09 S
		07:11 S	08:15 S	08:15 Dm
		12:05 Dm	14:35 dnw	18:05 dnw
		23:18 dnw		

Rys. 13.4. Przykład tabliczki przystankowej

PORZĄDEK WOKÓŁ NAS

Zwykle nie zwracamy uwagi na to, jak wiele obiektów wokół nas jest uporządkowanych. Ważne jest jednak nie tylko samo uporządkowanie, lecz także jego funkcjonalność, która znacznie ułatwia nam codzienne życie.

Kody pocztowe

Wprowadzenie kodów pocztowych znacznie usprawniło sortowanie i skróciło czas dostarczania listów. Wcześniej pracownicy poczty musieli sami odszyfrowywać nazwy miejscowości i sprawdzać ich lokalizację.

**Wąbrzeźno**

87

200

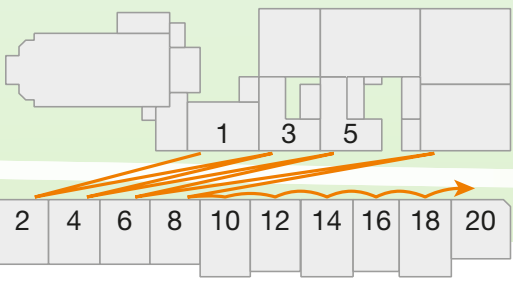
Okręg i strefa kodowa

Sektor kodowy i placówka pocztowa



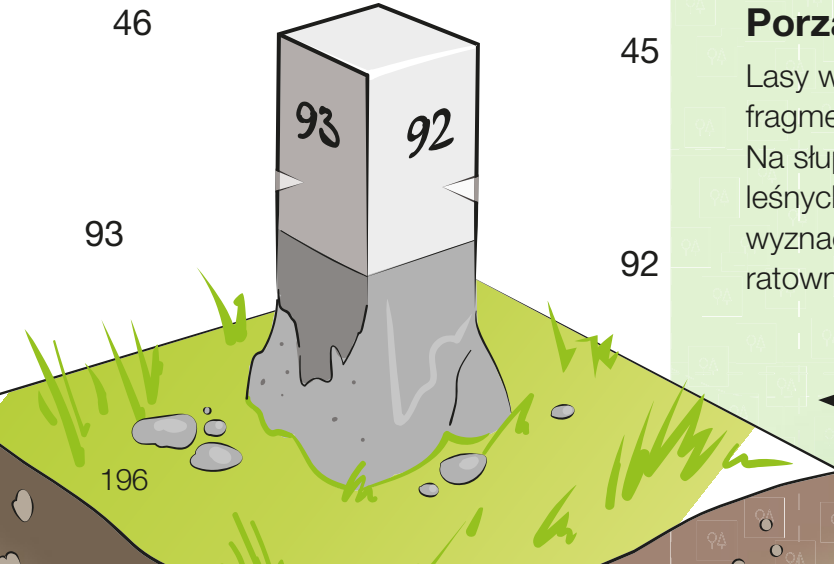
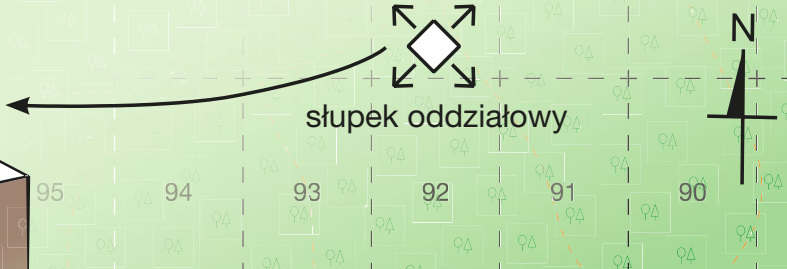
Numeracja domów

Domy znajdujące się w centrach miast są numerowane w uporządkowany sposób. Numery domów położonych wzdłuż głównych ulic rosną w kierunku od centrum miejscowości ku jej krańcom: po lewej stronie są numery nieparzyste, po prawej – parzyste.



Porządek w lesie

Lasy w Polsce są podzielone na oddziały (prostokątne fragmenty), które są identyfikowane przez słupki oddziałowe. Na słupkach znajdują się liczby oznaczające numery oddziałów leśnych. Słupki oddziałowe pomagają określić położenie, wyznaczyć kierunek oraz ułatwiają prowadzenie akcji ratowniczej.



Ćwiczenie 3

Z jakimi uporządkowanymi zbiorami danych masz do czynienia na co dzień? Wymień co najmniej pięć konkretnych przykładów.

Szukanie hasła w słowniku lub skorowidzu

W encyklopediach, słownikach i leksykonach obowiązuje porządek alfabetyczny. Gdy korzystasz z tradycyjnego słownika, np. polsko-angielskiego, zwykle nie szukasz na chybił trafił, lecz intuicyjnie stosujesz bardzo rozsądną strategię poszukiwania słowa. Podobnie postępujesz podczas szukania hasła w indeksie (skorowidzu) umieszczonym zwykle na końcu książki (rys. 13.5).

A	Grafika informacyjna (infografika) 74
Adres IP (adres logiczny) 30	— rastrowa (bitmapowa) 46
— MAC (adres fizyczny, adres sprzętowy) 27	— wektorowa 61
— opisowy (mnemoniczny) 34	
— URL 38	H
Atak socjotechniczny 127	Hejt 123
	Host 30
B	
Bezprzewodowa sieć lokalna (WLAN) 25	I
Big data 95	Identyfikator 108
Botnet 128	Importowanie danych 188
Brama 24	Indeksowanie plików 19
	Influencer 124

Rys. 13.5. Fragment indeksu podręcznika *NOWA Informatyka na czasie 1*

Jeśli chcesz znaleźć hasło *Adres IP*, to prawdopodobnie szukasz go wśród początkowych haseł, a jeśli *Influencer* – mniej więcej w środku. Następnie szukasz wyrazów zawierających dwie pierwsze litery – w naszej sytuacji będą to odpowiednio *Ad* i *In*. Za każdym razem zawężasz obszar poszukiwań, pomijając strony, na których dane hasło na pewno nie wystąpi. Posortowanie haseł przez twórców słownika pozwala więc znacznie skrócić czas przeszukiwania.

Warto wiedzieć

Porządek alfabetyczny nazywa się też porządkiem leksykograficznym.

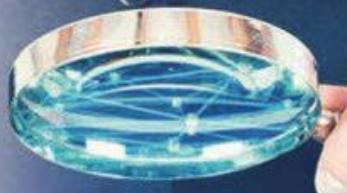
Warto wiedzieć

Metodę wyszukiwania, którą stosujemy intuicyjnie podczas korzystania ze słownika, nazywa się wyszukiwaniem interpolacyjnym.

A to ciekawe

Algorytmy wyszukiwania

Znalezienie w sieci informacji, których potrzebujemy, byłoby praktycznie niemożliwe bez ich uporządkowania. Systemy rankingowe wyszukiwarek internetowych przeglądają indeksy wyszukiwania, stosując wiele różnych algorytmów. Analizują takie aspekty, jak treść zapytania, przydatność stron, wiarygodność źródła, czas zamieszczenia informacji, lokalizacja użytkownika. Gdyby wykorzystywane algorytmy nie działały bardzo szybko dla dużych zestawów danych, na wyniki dla pojedynczego zapytania trzeba by było czekać nawet kilkanaście minut lub dłużej.



Klasyfikacja zawodników w pięcioboju nowoczesnym

Przyjrzyjmy się bliżej zasadom ustalania kolejności w pięcioboju nowoczesnym. Składa się on z szermierki, pływania, jazdy konnej oraz konkurencji łączonej – biegu na przełaj i strzelania z pistoletu. Wyniki w poszczególnych konkurencjach są przeliczane na punkty wielobojo-we. W tabeli 13.2 przedstawiono wyniki pierwszych dziesięciu zawod-niczek w klasyfikacji końcowej Letnich Igrzysk Olimpijskich w Paryżu w 2024 r. W rywalizacji brało udział 36 zawodniczek.

Lista zawodniczek			Szermierka		Pływanie			Jazda konna		Bieg ze strzelaniem			Suma
			punkty	miejsce	czas	miejsce	punkty	miejsce	punkty	czas	miejsce	punkty	
1.	Michelle Gulyas	HUN	245	2	2:12.44	4	286	1	300	11:10.01	7	630	1461
2.	Élodie Clouvel	FRA	264	1	2:11.64	3	287	9	293	11:32.35	14	608	1452
3.	Seungmin Seol	KOR	225	7	2:11.47	2	288	2	300	11:12.87	8	628	1441
4.	Blanka Guzi	HUN	200	17	2:16.25	7	278	8	300	10:45.48	1	655	1433
5.	Elena Micheli	ITA	232	3	2:12.47	5	286	14	293	11:27.86	12	613	1424
6.	Ilke Ozyuksel	TUR	205	15	2:15.48	6	280	12	293	10:58.20	3	642	1420
7.	Gintare Venckauskaite	LTU	205	16	2:18.16	12	274	5	300	11:00.31	4	640	1419
8.	Sunwoo Kim	KOR	222	8	2:17.67	11	275	16	286	11:13.92	9	627	1410
9.	Kerenza Bryson	GBR	230	4	2:21.77	15	267	15	286	11:19.12	10	621	1404
10.	Lucie Hlavackova	CZE	209	13	2:20.54	13	269	11	293	11:08.44	6	632	1403

Tabela 13.2. Wyniki pięcioboju nowoczesnego kobiet – igrzyska w Paryżu w 2024 r.

Ćwiczenie 4

a.

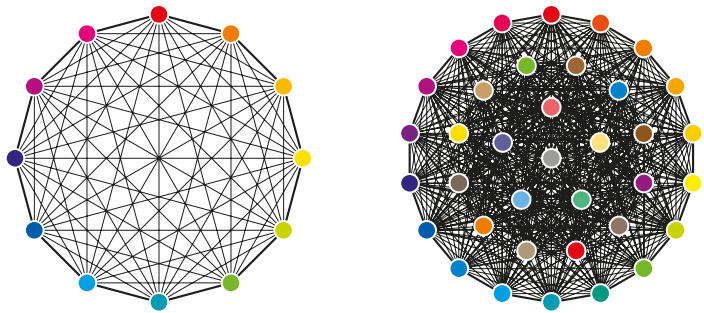
Według jakiego kryterium uporządkowano zawodniczki w tabeli 13.2?

b.

Czy klasyfikacje dla każdej pojedynczej konkurencji są tworzone na podstawie wyników uporządkowanych malejąco?

Ustalenie kolejności zawodników podczas zawodów w wieloboju jest szczególnie ważne przed rozegraniem ostatniej konkurencji (strze-lanie i bieg). Różnice punktowe w klasyfikacji są bowiem przeliczane na sekundy. Lider startuje jako pierwszy, a kolejni – z opóźnieniem wyliczonym na podstawie dotychczasowych wyników.

Systemy wyłaniania zwycięzców wcześniejszych konkurencji muszą być sprawiedliwe. Na przykład w szermierce stosuje się system „każdy z każdym”, zwany systemem kołowym. Przy dużej liczbie rywalizują-cych jest on jednak bardzo czasochłonny. Dla 36 zawodniczek (każda walczy 35 razy) należy przeprowadzić aż $\frac{1}{2} \cdot 36 \cdot 35 = 630$ pojedynków. Dla porównania – przy 12 zawodniczkach odbyłoby się 66 walk. Tę róż-nicę pokazuje rysunek 13.6.



Rys. 13.6. Ilustracja liczby pojedynków w rywalizacji „każdy z każdym” dla 12 i 36 zawodniczek

Sortowanie danych w komputerze

Sortowanie zbiorów danych jest jedną z operacji najczęściej wykony-wanych przez komputery podczas przetwarzania informacji. Oto kilka przykładów zastosowań sortowania, od prostszych do mniej oczywi-stych:

- ▶ wyświetlanie listy plików w katalogu (folderze) na dysku – przed wyświetleniem zawartości katalogu system operacyjny sortuje pliki zgodnie z określonym kryterium,
- ▶ przeglądanie danych z arkusza kalkulacyjnego lub bazy danych – często stosujemy sortowanie, aby ułatwić czytanie danych zawartych w tabelach (nadać im odpowiednie znaczenie),
- ▶ wyszukiwarki internetowe – zasoby zgromadzone na serwerach wy-szukiwarek internetowych muszą być zindeksowane, aby wyszukiwarka mogła zwrócić wynik możliwie szybko,
- ▶ grafika komputerowa – obiekty np. w grze komputerowej lub w symula-torze lotu są ułożone na warstwach, więc aby program mógł wyświetlić je na ekranie w dobrej kolejności, może wymagać ich wcześniejszego posortowania (względem odległości od obserwatora).

13.3. Sortowanie a wyszukiwanie

Uporządkowanie danych ma zasadnicze znaczenie podczas prze-szukiwania ich zbioru. Prześledźmy to na przykładzie wyszukiwania w pamięci komputera.

Przyjmijmy, że po uruchomieniu jakiegoś programu w pamięci komputera mamy zapisaną listę jako uporządkowany ciąg komórek pamięci RAM, zaadresowanych kolejnymi liczbami całkowitymi.

Warto wiedzieć

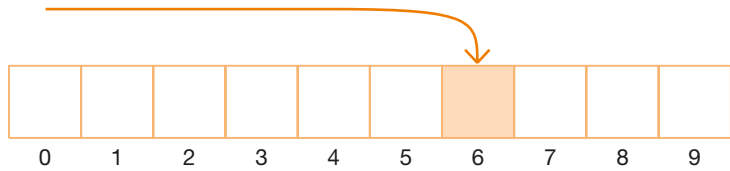
W języku angielskim system rywalizacji „każdy z każdym” nazywa się *round-robin tournament* (dosł. turniej karuzelowy).

Warto wiedzieć

Przykłady problemów, do których rozwiązania jest potrzebne sortowanie danych:

- ▶ usuwanie duplikatów,
- ▶ sprawdzanie, czy wszystkie elementy zbioru danych są różne,
- ▶ znajdowanie dwóch wartości o najmniejszej różnicy,
- ▶ wyznaczanie mediany, dominanty i innych statystyk,
- ▶ odtwarzanie pierwotnego porządku elementów zbioru danych.

Jeśli procesor zna położenie danych w pamięci, to odwołuje się do nich bezpośrednio, poprzez adres komórki, w której się znajdują (rys. 13.7).



Rys. 13.7. Dostęp bezpośredni

Warto wiedzieć

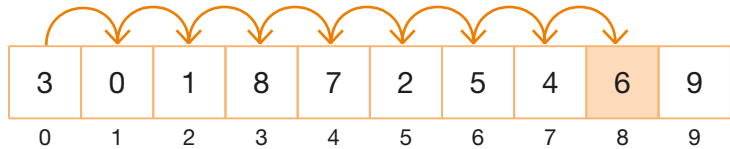
Czas dostępu do danych zależy nie tylko od ich uporządkowania, lecz także od rodzaju pamięci. Różny czas osiągają rejestry procesora, pamięć podręczna, pamięć RAM czy pamięć zewnętrzna (np. flash, dysk twardy).

Aby odczytać dane zapisane w siódmej komórce (indeks 6) fragmentu pamięci, procesor nie odczytuje wszystkich danych w kolejności, w której zostały zapisane, lecz pobiera dane z właściwej komórki (rys. 13.7).

Współczesne algorytmy komputerowe często wymagają, aby czas dostępu do danych z pamięci operacyjnej był jak najkrótszy. Wtedy kluczowe znaczenie – oprócz aspektu technicznego (budowy pamięci) – ma właśnie uporządkowanie danych. Jego brak znacznie spowalnia wyszukiwanie informacji, nawet jeśli korzysta się z szybkich procesorów.

Wyszukiwanie sekwencyjne

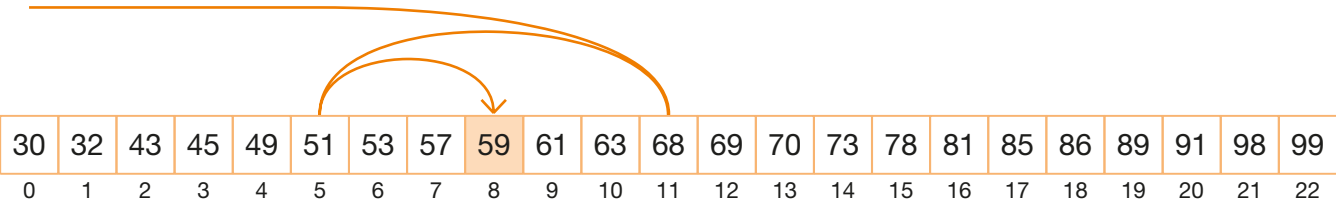
Kiedy szukamy informacji w zbiorach danych nieuporządkowanych, nie mamy wielkiego wyboru: albo zastosujemy metodę „na chybił trafił”, albo algorytm **wyszukiwania sekwencyjnego** (ang. *sequential search*), tj. przeglądanie zbioru danych w kolejności, w której są zapisane (np. według adresów w pamięci). Rysunek 13.8 pokazuje sekwencyjne wyszukiwanie liczby 6 w zbiorze nieuporządkowanym.



Rys. 13.8. Wyszukiwanie sekwencyjne

Wyszukiwanie binarne

Wyszukiwanie informacji można znacznie przyspieszyć, gdy zbiór danych jest uporządkowany. Stosujemy wówczas strategię połowienia zbioru, czyli zmniejszania liczby jego elementów o połowę w każdej kolejnej próbie. Tę strategię nazywamy **wyszukiwaniem binarnym (połówkowym)**. Przykładowy algorytm zastosowany do poszukiwania liczby 59 w zbiorze 23 elementów przedstawiono na rysunku 13.9.



Rys. 13.9. Wyszukiwanie binarne

Wyszukiwanie liczby 59 zaczęło się od sprawdzenia środkowej komórki obszaru pamięci. Jest to komórka o indeksie 11, przechowująca liczbę 68, która jest większa od 59. Ponieważ wiemy, że dane w pamięci są uporządkowane, mamy pewność, że przeszukiwanie komórek o indeksach większych od 11 można pominąć. W drugiej próbie program wyszukujący powinien sprawdzić komórkę o indeksie 5. Znajduje się w niej liczba 51, dlatego wartości należy szukać w komórkach o indeksach od 6 do 10. W kolejnym kroku sprawdzamy komórkę o adresie 8 – znajdziemy w niej poszukiwaną liczbę.

Ćwiczenie 5

Ile sprawdzeń (porównań dwóch liczb) wystarczy, aby rozstrzygnąć, czy w ciągu komórek z rysunku 13.9 znajduje się liczba 80? Zapisz indeksy oraz wartości komórek, które kolejno będą sprawdzane.

13.4. Algorytmy sortowania

Spójrzmy na porządkowanie jako na problem algorytmiczny i programistyczny. W ostatnich kilkudziesięciu latach odkryto i opisano co najmniej kilkanaście użytecznych algorytmów sortowania danych. Okazuje się, że nie ma algorytmu idealnego o ogólnym przeznaczeniu, najszybszego w każdej sytuacji. Najbardziej uniwersalne są metody hybrydowe, łączące zalety dwóch lub trzech algorytmów sortowania, stosowanych na różnych etapach procesu porządkowania.

Ćwiczenie 6

Przygotuj 8 kartek wielkości połowy kartki z zeszytu. Na każdej z nich zapisz imię oraz datę imienin (dzień i miesiąc). Imiona nie mogą się powtarzać. Następnie odwróć kartki i ułóż je w losowej kolejności.

- a. Zaproponuj metodę sortowania informacji na kartkach (alfabetycznie lub chronologicznie). Przyjmij, że kartki trzeba porównywać parami i nie wolno odkryć więcej niż dwóch naraz.
- b. W grupie dwu- lub trzyosobowej przedyskutujcie zastosowane wcześniej metody porządkowania kartek. Porównajcie czas sortowania 8 i 16 kartek wybraną przez was metodą. Ile razy dłużej trwa porządkowanie 16 kartek?

Programy komputerowe można wykorzystać do porządkowania różnych typów danych: liczb, zbiorów liter i słów, dat i innych. Wszystkie te wartości są zapisywane cyfrowo w pamięci komputera. Dla uproszczenia ograniczymy się więc do porządkowania liczb całkowitych. Dodatkowo przyjmijmy, że będziemy sortować w porządku niemalejącym.

Warto wiedzieć

Porównanie dwóch liczb jest jedną z podstawowych operacji wykonywanych przez procesor komputerowy. Każdy język programowania musi obsługiwać tę operację.

Warto wiedzieć

Algorytmy sortowania można podzielić na dwie grupy: te, które działają na zasadzie porównywania elementów zbioru parami (np. sortowanie bąbelkowe), oraz nieposługujące się tą operacją (np. sortowanie przez zliczanie).

Oto specyfikacja naszego problemu sortowania:

Specyfikacja

Dane: liczba naturalna n i n -elementowa tablica A liczb całkowitych.
Wynik: tablica A po uporządkowaniu: $A[0] \leq A[1] \leq \dots \leq A[n-1]$.

13.5. Sortowanie bąbelkowe

Sortowanie bąbelkowe • Jednym z algorytmów porządkowania jest **sortowanie bąbelkowe** (ang. *bubble sort*). W tym algorytmie na każdym etapie porównuje się parę sąsiednich elementów, zaczynając od pierwszej pary na liście. Jeżeli podczas porównywania elementy są w niewłaściwej kolejności, to zamienia się je miejscami.

Operacja prostej zamiany

Operacja prostej zamiany • **prostą zamianę** rozumianą jako zamianę miejscami sąsiednich elementów tablicy.

Podobną operację stosuje się w oprogramowaniu niektórych telewizorów. Pozwala ono za pomocą pilota uszeregować listę kanałów. Są trzy możliwości: zamiana kanałów miejscami (•), przejście w dół listy (▽) i przejście w górę listy (△). Rysunek 13.10 przedstawia zastosowanie operacji prostej zamiany podczas ustawiania listy kanałów.

TV 3	•	TV 0	▽	TV 0		TV 0		TV 0		TV 0
TV 0		TV 3	•	TV 1	▽	TV 1		TV 1		TV 1
TV 1		TV 1		TV 3	▽	TV 3		TV 3	•	TV 2
TV 6		TV 6		TV 6	▽	TV 6	•	TV 2	△	TV 3
TV 7		TV 7		TV 7	•	TV 2	△	TV 6		TV 6
TV 2		TV 2		TV 2		TV 7		TV 7		TV 7
TV 5		TV 5		TV 5		TV 5		TV 5		TV 5
TV 4		TV 4		TV 4		TV 4		TV 4		TV 4

Rys. 13.10. Pierwsze kroki porządkowania kanałów telewizyjnych przez prostą zamianę

A to ciekawe

Sortowanie w tańcu

Na kanale AlgoRythmics w serwisie YouTube można znaleźć filmy objaśniające działanie różnych algorytmów sortowania. Porządkowanie liczb jest zaprezentowane w tańcu przez członków węgierskiego zespołu folkowego Maros Művészegyüttes. Można tam zobaczyć również m.in. wyszukiwanie binarne.



ALGORYTM SORTOWANIA BĄBELKOWEGO

Porządkowanie elementów tą metodą pokażemy na przykładzie pięcioelementowej tablicy, w której na kolejnych pozycjach znajdują się liczby: 6, 4, 9, 1, 5. Elementy uporządkujemy niemalejąco. Pierwszy etap ilustruje, jak umieścić właściwy element na ostatniej pozycji. W kolejnych etapach postępujemy analogicznie.

Etap 1

- Porównujemy pierwszy element z drugim. $6 > 4$, więc zamieniamy liczby miejscami.
- Porównujemy drugi element z trzecim. $6 < 9$, więc nie zamieniamy liczb miejscami.
- Porównujemy trzeci element z czwartym. $9 > 1$, więc zamieniamy liczby miejscami.
- Porównujemy czwarty element z piątym. $9 > 5$, więc zamieniamy liczby miejscami.
- W wyniku czterech porównań największy element tablicy (liczba 9) znalazł się na ostatniej pozycji.

6 > 4

9

1

5

4

6 < 9

1

5

4

6

9 > 1

5

4

6

1

9 > 5

4

6

1

5

9

W drugim etapie umieścimy właściwy element na przedostatniej pozycji. Przy porównaniach pomijamy ostatni element tablicy, ponieważ jest już na właściwym miejscu. W trzecim etapie przy porównaniach pomijamy dwa ostatnie elementy tablicy, w czwartym – trzy ostatnie itd. Poniżej przedstawione są wyniki otrzymane w kolejnych etapach.

Etap 2

Porównujemy pary liczb: 4 i 6, 6 i 1 oraz 6 i 5. Liczba 6 znalazła się na właściwej, przedostatniej pozycji.

4

1

5

6

9

Etap 3

Porównujemy pary liczb 4 i 1 oraz 4 i 5. W wyniku tych operacji liczba 5 jest na właściwej pozycji.

1

4

5

6

9

Etap 4

Porównujemy parę liczb 1 i 4. Znalezienie drugiego elementu ciągu wskazuje jednocześnie najmniejszy element ciągu.

1

4

5

6

9

Wynikiem działania algorytmu jest tablica:

1	4	5	6	9
---	---	---	---	---

Ćwiczenie 7

Dana jest tablica $A = [3, 0, 1, 7, 5]$. Sprawdź, ile prostych zamian potrzeba, by uporządkować ją algorytmem sortowania bąbelkowego:

- a. rosnąco,
- b. malejąco.

Algorytm porządkowania metodą bąbelkową można zapisać jako

Pętle zagnieżdżone

pętle zagnieżdżone (ang. *nested loops*), czyli pętle umieszczone wewnątrz innych pętli. To oznacza, że na jeden etap algorytmu (cykl pętli zewnętrznej) może przypadać wiele cykli pętli wewnętrznej.

Każdy etap algorytmu kończy się ustawieniem na docelowym miejscu elementu, który w analizowanym fragmencie był tym o największej wartości. Liczba porównań w kolejnym etapie zmniejsza się więc o 1.

Oto zapis algorytmu w postaci listy kroków:

- 1 Wczytaj tablicę liczb A.
- 2 Dla wartości i równych kolejno 1, 2, ..., n-1 powtarzaj kroki 3 oraz 4.
- 3 Dla wartości j równych kolejno 0, 1, ..., n-i-1 powtarzaj krok 4.
- 4 Jeśli $A[j] > A[j+1]$, to zamień te elementy miejscami.

Liczba powtórzeń zewnętrznej pętli wynosi n-1. Liczba powtórzeń wewnętrznej pętli (czyli pętli sortującej) jest zależna od wartości zmiennej i, sterującej zewnętrzną pętlą.

Kod źródłowy funkcji o nazwie Babelkowe w programie Sortowanie bąbelkowe może wyglądać następująco:

```
1. void Babelkowe(int A[])
2. {
3.     for (int i=1; i<N; i++)
4.         for (int j=0; j<N-i; j++)
5.             if (A[j]>A[j+1])
6.                 swap(A[j],A[j+1]);
7. }
```

Dobra rada

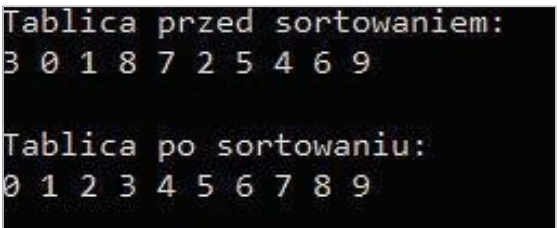
Aby uporządkować elementy nierosnąco, zmień w instrukcji warunkowej znak > na <.

Kod źródłowy funkcji Babelkowe w programie Sortowanie bąbelkowe

Warto wiedzieć

W nagłówku definiowanej funkcji, np. `void Funkcja(int A[])`, nazwa tablicy nie musi być identyczna z nazwą tablicy, dla której wywołamy funkcję. Funkcję można wywołać np. dla tablicy B i zapisać `Funkcja(B)`.

Rysunek 13.11 przedstawia przykładowe wywołanie programu Sortowanie bąbelkowe.



Rys. 13.11. Przykładowe wywołanie programu Sortowanie bąbelkowe

Ćwiczenie 8

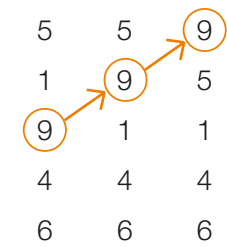
Napisz kod źródłowy programu o nazwie Sortowanie bąbelkowe, w którym wykorzystasz funkcję Babelkowe, a dane do sortowania wprowadzisz bezpośrednio w kodzie źródłowym. Sprawdź działanie programu.

Zapamiętaj

Kluczową operacją w wielu algorytmach sortowania danych jest porównywanie danych parami, by ustalić ich porządek.

Warto wiedzieć

Nazwa sortowanie bąbelkowe powstała przez analogię do zjawiska wypływania na powierzchnię wody bąbelków powietrza: ten o największej objętości wypływa jako pierwszy.



13.6. Sortowanie przez wstawianie

Omówimy teraz kolejny algorytm porządkowania danych – **sortowanie przez wstawianie** (ang. *insertion sort*). Ta metoda polega na przeglądaniu kolejnych elementów tablicy od lewej i przenoszeniu ich, w razie potrzeby, w odpowiednie miejsce już uporządkowanego fragmentu tablicy po lewej stronie. Działanie algorytmu przedstawiono na s. 206.

Sortowanie przez wstawianie przypomina układanie kart przez graczy w gry karciane: karta, którą podnosimy ze stołu, jest wstawiana we właściwe miejsce do kart już uporządkowanych (rys. 13.12).



Rys. 13.12. Układanie kart metodą przez wstawianie

Sortowanie przez wstawianie

Warto wiedzieć

Doświadczeni gracze w pokera lub brydża nie sortują swoich kart. W ten sposób bowiem dostarcziliby przeciwnikom informacji o tym, jakie karty mają w ręku.

ALGORYTM SORTOWANIA PRZEZ WSTAWIANIE

Porządkowanie elementów tą metodą pokażemy na przykładzie pięcioelementowej tablicy, w której na kolejnych pozycjach znajdują się liczby: 6, 4, 9, 1, 5. Elementy uporządkujemy niemalejąco. Pierwszy etap ilustruje, jak uporządkować niemalejąco dwa elementy tablicy. Liczby przy strzałkach oznaczają kolejność wykonywania operacji.

Etap 1

- Przyjmujemy, że pierwszy element tablicy (liczba 6) jest na właściwym miejscu. Drugi element, liczbę 4, zapamiętujemy w zmiennej pomocniczej.
- Liczba 4 powinna się znaleźć przed liczbą 6, dlatego aby zrobić dla niej miejsce, przesuwamy liczbę 6 o jedną pozycję w prawo.
- Liczbę 4 ze zmiennej pomocniczej zapisujemy jako pierwszy element tablicy.
- Wynikiem jest uporządkowanie niemalejąco dwóch pierwszych elementów wyjściowej tablicy.

Postępując analogicznie, w drugim etapie uporządkujemy niemalejąco trzy elementy tablicy, w następnym – cztery itd. Poniżej znajduje się skrócony opis kolejnych etapów.

Etap 2

Zapamiętujemy liczbę 9 w zmiennej pomocniczej. Liczba 9 powinna się znaleźć za liczbami 4 i 6, dlatego nie trzeba zmieniać położenia tych liczb, a liczba 9 wraca na swoją pozycję.

Etap 3

Zapamiętujemy liczbę 1 w zmiennej pomocniczej. Przesuwamy w prawo o jedną pozycję kolejno liczby: 9, 6 i 4. Zapisujemy liczbę 1 na pierwszym miejscu w tablicy.

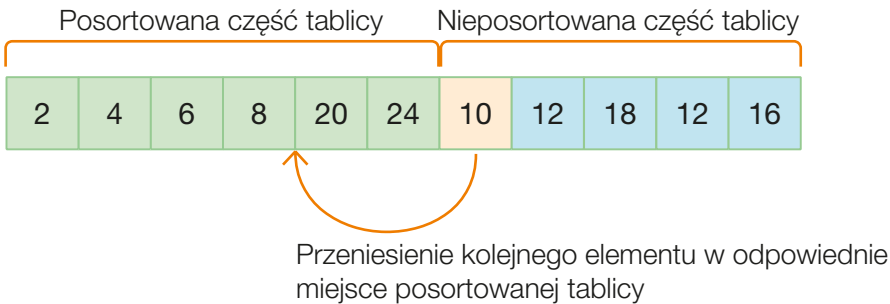
Etap 4

Zapamiętujemy liczbę 5 w zmiennej pomocniczej. Przesuwamy liczbę 9, a następnie liczbę 6 o jedną pozycję w prawo. Wstawiamy liczbę 5 ze zmiennej pomocniczej w zwolnione miejsce.

Wynikiem działania algorytmu jest tablica:

1	4	5	6	9
---	---	---	---	---

Algorytm dzieli tablicę danych na część posortowaną i nieposortowaną (rys. 13.13). Na każdym etapie algorytmu kolejny element z nieposortowanej części tablicy trafia na właściwą pozycję w obszarze posortowanym.



Rys. 13.13. Umieszczenie liczby 10 w częściowo uporządkowanej tablicy

Algorytmy sortowania bąbelkowego i sortowania przez wstawianie są do siebie podobne: pierwszy stopniowo przesuwamy elementy zbioru ku końcowi, podczas gdy drugi przenosi elementy zbioru ku początkowi. Wstawienie elementu tablicy na właściwą pozycję odbywa się dzięki zwolnieniu miejsca w tablicy poprzez przesunięcie, krok po kroku, całego bloku elementów o jedną pozycję w prawo.

Przesuwanie elementów jest realizowane przez serię operacji kopiowania. Element, który będzie docelowo wstawiony we właściwe miejsce uporządkowanej części tablicy, jest tymczasowo przechowywany w zmiennej pomocniczej. Porządkowanie rozpoczynamy od drugiego elementu (o indeksie 1).

Oto zapis algorytmu w postaci listy kroków:

- 1

Wczytaj tablicę liczb A.
- 2

Dla wartości i równych kolejno 1, 2, ..., n-1 powtarzaj kroki 3-8.
- 3

W zmiennej pom zapamiętaj wartość zapisaną w A[i].
- 4

Oznacz bieżącą wartość i-1 literą j.
- 5

Dopóki j ≥ 0 i A[j] > pom, wykonuj kroki 6-7.
- 6

Skopiuj A[j] na miejsce A[j+1].
- 7

Zmniejsz wartość j o 1.
- 8

W komórce A[j+1] zapisz wartość pom.

Ćwiczenie 9

Prześledź działanie listy kroków algorytmu sortowania przez wstawianie dla następujących tablic.

- a. A = [2, 3, 8, 5]
- b. B = [2, 8, 3, 5]
- c. C = [8, 2, 3, 5]

Warto wiedzieć

Sortowanie bąbelkowe i sortowanie przez wstawianie przeprowadzają operacje w miejscu. To oznacza, że poza pamięcią dla sortowanej tablicy potrzebują bardzo mało dodatkowej pamięci.

Warto wiedzieć

Operację znalezienia właściwego miejsca dla danego elementu w ciągu uporządkowanym można realizować przez przeszukiwanie liniowe lub binarne. Bardziej efektywny jest drugi sposób. Już w latach 50. XX w. analizował go polski matematyk Hugo Steinhaus.

Oto kod źródłowy funkcji o nazwie `Wstawianie` w programie *Sortowanie przez wstawianie*.

Kod źródłowy funkcji  **Wstawianie w programie**
Sortowanie przez wstawianie

 Warto wiedzieć

Algorytm sortowania przez wstawianie jest szczególnie efektywny przy sortowaniu zbiorów prawie uporządkowanych, np. rankingów aktualizowanych na bieżąco podczas zawodów. Wykorzystuje się go również w rozwiązaniach hybrydowych, m.in. w ostatniej fazie sortowania szybkiego (ang. *quicksort*), gdy do posortowania jest wiele podzbiorów o niewielkiej liczbie elementów.

Logika matematyczna i operatory logiczne
→ s. 220–222

Leniwe wartościowanie 

```
1. void Wstawianie(int A[])
2. {
3.     int j, pom;
4.     for (int i=1; i<N; i++)
5.     {
6.         pom = A[i];
7.         j = i - 1;
8.         while (j>=0 && A[j]>pom)
9.         {
10.            A[j+1] = A[j];
11.            j--;
12.        }
13.        A[j+1] = pom;
14.    }
15. }
```

Pętla zewnętrzna (linie 4.–14.) przegląda elementy z części nieposortowanej. W kolejnych iteracjach w zmiennej pomocniczej `pom` zapamiętujemy kolejne elementy z tej części (wiersz 6.). Jeżeli element z lewej strony aktualnie analizowanego elementu jest większy ($A[j]>pom$), to przesuwamy elementy posortowanej części ciągu, które powinny się znaleźć za danym elementem, a jego wartość zapamiętaną w zmiennej `pom` wstawiamy na właściwe miejsce (linie 8.–13.).

Warunkiem w wierszu 8. jest **koniunkcja**, czyli złożone wyrażenie logiczne. Program po uruchomieniu będzie działał poprawnie, ponieważ w działaniu kompilatora języka C++ stosuje się **leniwe wartościowanie** (ang. *lazy evaluation*). W naszym przypadku drugie porównanie się nie wykona, jeśli pierwszy warunek nie jest spełniony, tzn. gdy indeks `j` będzie mniejszy niż indeks pierwszego elementu tablicy.

Ćwiczenie 10

Zapisz kod źródłowy funkcji `Wstawianie` i napisz program, w którym tę funkcję zastosujesz. Sprawdź działanie programu.

 Zapamiętaj

Sortowanie przez wstawianie wymaga użycia zmiennej pomocniczej, która tymczasowo przechowuje rozpatrywany element.

 Zadania

- * **1** Napisz program, który wyznaczy najmniejszą liczbę z ciągu liczb całkowitych dodatnich podawanych z klawiatury. Przyjmij, że podanie liczby 0 kończy wprowadzanie liczb.
- * **2** Zmodyfikuj program *Najdłuższy skok* tak, aby trener mógł dodatkowo zliczać liczbę skoków nieważnych (spalonych).
- * **3** Napisz program sprawdzający, czy tablica 10 liczb jest uporządkowana.
- * **4** Napisz program, który po wczytaniu 10 liczb dodatnich całkowitych znajdzie różnicę między największą a najmniejszą liczbą.
- * **5** Napisz program, który będzie porządkować malejąco tablicę 10 różnych liczb metodą:
 - a. bąbelkową,
 - b. przez wstawianie.
- ** **6** Napisz program, który zliczy osobno, ile liczb ujemnych i ile liczb dodatnich wprowadzono z klawiatury. Przyjmij, że wczytanie liczby 0 kończy wczytywanie liczb.
- ** **7** Napisz program, który po wczytaniu 10 liczb wskaże te, które potraktowane jako długości boków utworzą prostokąt o największym polu.
- ** **8** Poszukaj informacji o algorytmie sortowania przez wybór. Napisz program, który będzie sortować liczby całkowite za pomocą tej metody.
- ** **9** Napisz program, który zamieni miejscami elementy znajdujące się w pierwszej i drugiej połowie tablicy o parzystej liczbie elementów.

Dane wejściowe:

3	1	4	5	2	8	0	3
---	---	---	---	---	---	---	---

Dane wyjściowe:

2	8	0	3	3	1	4	5
---	---	---	---	---	---	---	---

Program nie powinien używać dodatkowej tablicy pomocniczej.
- *** **10** Dla 6 liczb wprowadzonych z klawiatury zawsze wystarczy 7 porównań, aby wyznaczyć najmniejszą oraz największą z nich. Jak to zrobić? Zapisz algorytm w postaci listy kroków, a następnie jego realizację w języku programowania.
- *** **11** Napisz program sortujący metodą przez wstawianie z użyciem wartownika. Wskazówka: Wartownika umieść jako dodatkowy element w pierwszej komórce tablicy zawierającej liczby do posortowania.
- *** **12** Napisz program, który po wczytaniu 20 liczb dodatnich całkowitych znajdzie wśród nich parę liczb o najmniejszej różnicy.

Algorytmika i programowanie w języku C++

Wyobraź sobie, że...

...zaczynasz studia na politechnice, ponieważ chcesz zostać inżynierem geodetą. Okazuje się, że na pierwszym roku masz zajęcia z programowania. Na szczęście w szkole poznałaś **podstawowe konstrukcje programistyczne, typy danych i proste struktury danych** oraz zdobyłaś pewne doświadczenie w programowaniu w **środowisku tekstowym**. Radzisz sobie, rozumiesz omawiane algorytmy. Potrafisz je też zaprogramować, mimo że używasz innego języka programowania niż w szkole.



...jesteś inżynierem pożarnictwa i pracujesz w firmie opracowującej zasady bezpieczeństwa dla obiektów biurowych. Wiesz, że instrukcje zawierające sposób postępowania na wypadek pożaru lub innego zagrożenia muszą być bardzo precyzyjne. Jednym ze sposobów ich zapisania jest forma **listy kroków**, z którą zetknąłeś się w szkole podczas nauki programowania. Twoja praca cieszy się dużym uznaniem, ponieważ instrukcje tworzone przez ciebie są dla wszystkich zrozumiałe.



...ukończyłaś studia prawnicze i dostałaś się na staż w znanej kancelarii adwokackiej. Często musisz sięgać po różne akta, które nie są posegregowane. Wiesz, że w uporządkowanym zbiorze łatwiej **wyszukiwać** określone elementy. Przypominasz sobie o algorytmach **sortowania danych** i postanawiasz je wykorzystać. Sortujesz akta według dwóch kryteriów: najpierw chronologicznie, a potem alfabetycznie.



...pracujesz w firmie produkującej meble, która wdraża system wsparcia produkcji wyposażony w sztuczną inteligencję. Na szkoleniu pokazano model sztucznego neuronu wraz z **kodem źródłowym**. Na jego podstawie wyjaśniono podstawę działania całego systemu. Jesteś zachwycony, ponieważ do zrozumienia kodu wystarczyła wiedza o **instrukcji warunkowej** i **pętlach** wyniesiona ze szkoły.



Projekt zespołowy

Tworzenie strony WWW z wykorzystaniem narzędzi AI

Strony internetowe wykorzystuje się dziś w niemal każdej sferze działalności człowieka. Dzięki istniejącym narzędziom sztucznej inteligencji w zasadzie każdy może stworzyć taką stronę. W tym projekcie utworzycie stronę internetową, wykorzystując narzędzia AI.



Cel projektu

Wykonanie i opublikowanie w Internecie responsywnej strony internetowej z wykorzystaniem narzędzi AI.

To trzeba wiedzieć

Hosting

Stronę internetową należy umieścić na serwerze, który będzie ją obsługiwał. Jeśli nie mamy dostępu do własnego serwera, możemy znaleźć dostawcę takiej usługi w Internecie.

Responsywność

Jest to cecha stron internetowych, dzięki której układ i wielkość strony dostosowują się do urządzenia, na którym jest wyświetlana. Responsywność strony WWW jest brana pod uwagę przez algorytmy wyszukiwarek przy ustalaniu wyników wyszukiwania.

Narzędzia AI do generowania stron WWW

To ogólnodostępne, często darmowe narzędzia pozwalające wygenerować kod HTML strony internetowej, style CSS, a nawet skrypty takich języków, jak JavaScript i PHP. Nie wymagają dużej znajomości tych języków, lecz precyzyjnego opisu naszych oczekiwań. Dobrze jednak znać podstawy używanych języków, aby samodzielnie wprowadzać modyfikacje w kodach.

Adres strony

Każda strona internetowa ma swój unikalny adres. Dostępność nazw dla popularnych domen komercyjnych można sprawdzić np. na stronie <https://www.dns.pl>.

Propozycje tematów

1. Historia sztucznej inteligencji
2. Mój kawałek wszechświata (lokalne ciekawostki, miejsca, historia itp.)
3. Kuchnie świata (ciekawe dania i przepisy z różnych stron świata)
4. Matematyczne ciekawostki
5. Niewyjaśnione zagadki starożytności

Etapy projektu

1. Wybierzcie temat projektu strony WWW – najlepiej taki, który odpowiada waszym zainteresowaniom (możecie skorzystać z podanych wyżej propozycji). Temat dobierzcie tak, aby treść, którą zaprezentujecie, była bogata merytorycznie i wizualnie.
2. Przemyślcie i przygotujcie konspekt opisujący:
 - a. strukturę (menu) witryny,
 - b. kolorystykę strony i wygląd graficzny,
 - c. tematykę strony,
 - d. tematykę poszczególnych wpisów na stronie,
 - e. podział ról w zespole tworzącym stronę – należy uwzględnić role: projektanta strony, administratora, autora wpisów, redaktora, grafika.
3. Utwórzcie konto na stronie <https://chatgpt.com> i korzystając z przygotowanego wcześniej konspektu, wykonajcie następujące czynności:
 - a. po zalogowaniu, z lewej strony głównego ekranu wybierzcie możliwość korzystania z innych modeli GPT – wybranie tej opcji pozwoli wam użyć narzędzia Website Generator,
 - b. zaznaczcie **Build and host a website**, aby po stworzeniu strony od razu udostępnić ją w Internecie,
 - c. nadajcie tworzonej stronie nazwę i szczegółowo opiszcie, czego oczekujecie.
4. Zadbajcie o precyzyjne opisy (zapytania, prompty) dla chata GPT, a zwrócone kody od razu zapiszcie na swoich komputerach.
5. Wygenerowany projekt od razu zostanie umieszczony w serwisie <https://b12.io>. Załóżcie darmowe konto w tym serwisie, aby móc wprowadzać zmiany na swojej stronie.
6. Przetestujcie opcje wyświetlania serwisu na różnych urządzeniach i wymieńcie się spostrzeżeniami. Nanieście poprawki na podstawie uwag, które zbierzecie.
7. Wypełnijcie serwis treścią.
8. Kiedy strona będzie gotowa, udostępnicie jej adres nauczycielowi.

Przykładowe narzędzia

ChatGPT, Microsoft Copilot, Google Gemini to popularne chatboty oparte na modelach sztucznej inteligencji, które potrafią odpowiedzieć na niemal każde pytanie i wygenerować wiele różnych treści. Pamiętajmy jednak, że nie są one doskonałe i bezbłędne. Dane, które się z nich uzyskuje, lepiej zweryfikować w dodatkowych źródłach, a wygenerowane kody przejrzeć i dostosować do własnych potrzeb.

Dodatki

1. Podstawowe znaczniki języka HTML

Znacznik	Opis	Przykład
<!DOCTYPE>	Określa typ dokumentu.	<pre><!DOCTYPE html> <html lang="pl"> <head> <meta charset="UTF-8"> <title>Tytuł strony WWW</title> </head> <body> </body> </html></pre>
<html></html>	To znacznik główny strony internetowej.	
<head></head>	Zawiera metadane o dokumencie HTML.	
<body></body>	Zawiera elementy widoczne na stronie WWW.	
<meta>	Pozwala sprecyzować konkretne metadane strony WWW, np. sposób kodowania znaków na stronie internetowej.	
<link>	Pozwala podlinkować plik, np. ze stylami CSS.	<pre><link rel="stylesheet" type="text/css" href="style.css"></pre>
	To znacznik nagłówek pierwszego poziomu. Jest jednym z sześciu znaczników nagłówkowych: od <h1> do <h6>.	<pre><h1>Tekst nagłówka</h1></pre>
<a>	Pozwala definiować linki na stronach WWW.	<pre>tekst linku</pre>
<title></title>	Wyświetla tytuł strony WWW.	<pre><title>Tytuł strony</title></pre>
	Znacznik akapitu.	<pre><p>Rozpoczynamy nowy akapit tekstu</p></pre>
	Pogrubienie tekstu.	<pre><p> Liczbę podzieloną przez 2 bez reszty nazywamy parzystą. </p></pre>
	Pogrubienie tekstu i wskazanie jego ważności urządzeniom takim jak czytniki ekranu.	<pre><p>Liczbę podzieloną przez 2 bez reszty nazywamy parzystą.</p></pre>
<i></i>	Pochylenie tekstu.	<pre><p> Adam Mickiewicz, <i>Dziady</i> </p></pre>
>	Pochylenie tekstu i wskazanie jego ważności urządzeniom takim jak czytniki ekranu.	<pre><p>Adam Mickiewicz, Dziady</p></pre>
 	Pozwala na złamanie linii lub wstawienie pustej linii.	<pre><p>Tekst po znaczniku przenosimy
 do nowej linii</p></pre>

	Wstawia obrazy w dokumencie HTML.	<pre></pre>
<picture></picture>	Pozwala uzależnić wybór wyświetlanego obrazu np. od urządzenia, rozdzielczości ekranu.	<pre><picture> <source media="(min-width:750px)" srcset="obraz1.jpg"> <source media="(min-width:450px)" srcset="obraz2.jpg"> </picture></pre>
<source>	Definiuje własności obrazów wewnątrz znacznika <picture>.	
<table></table>	To główny znacznik tabeli.	<pre><table> <tr> <th>Imię</th> <th>Nazwisko</th> </tr> <tr> <td>Jan</td> <td>Kowalski</td> </tr> <tr> <td>Anna</td> <td>Nowak</td> </tr> </table></pre>
<tr></tr>	Opisuje wiersz w tabeli.	
<td></td>	Opisuje komórkę w wierszu tabeli.	
<th></th>	Pozwala na dodanie nagłówka tabeli (domyślnie tekst nagłówka jest pogrubiony).	
	Pozwala opisać element listy uporządkowanej i nieuporządkowanej.	<pre>element listy</pre>
	Lista uporządkowana (numerowanie).	<pre> Pierwszy element listy Drugi element listy Trzeci element listy </pre>
	Lista nieuporządkowana (wypunktowanie).	<pre> Element listy Element listy Element listy </pre>
<hr>	Linia pozioma. Jest stosowana przy oddzielaniu różnych elementów strony internetowej.	<pre><h2>Moja strona</h2> <hr> <p>Witaj na mojej stronie</p></pre>
<!-- -->	Wstawia komentarz w kodzie HTML.	<pre><!-- tekst komentarza --></pre>
	Pozwala zdefiniować fragment strony, który może być jednolicie sformatowany. Wykorzystuje się go do tworzenia szkieletu strony.	<pre><div style="color: #6d7177"> <h3>Zabytki</h3> <p>Opis</p> </div></pre>

2. Typy danych i maski w programie Microsoft Access

Wybrane typy danych

Typ danych	Opis
Krótki tekst	Pozwala na przechowywanie 255 znaków alfanumerycznych (np. liter, liczb, znaków przestankowych). Wykorzystywany jest do zapisywania imion, nazwisk, danych teleadresowych i liczb, na których nie wykonuje się obliczeń (np. numerów telefonów).
Długi tekst	Podobnie jak Krótki tekst pozwala zapisać teksty, może jednak zawierać do 64 tysięcy znaków.
Liczba	Przechowuje dane liczbowe, na których można wykonywać obliczenia, z wyjątkiem obliczeń walutowych (patrz: typ Waluta).
Data/Godzina	Przechowuje informacje na temat czasu: daty i godziny.
Waluta	Przechowuje dane liczbowe z dokładnością do 4 miejsc dziesiętnych i zapobiega zaokrągleniom podczas obliczeń.
Autonumerowanie	Przechowuje unikatową wartość generowaną przez program dla każdego nowego rekordu.
Tak/Nie	Przechowuje wartość logiczną typu prawda (1) lub fałsz (0).
Załącznik	Pozwala dołączać pliki, takie jak obrazy, dokumenty tekstowe, arkusze kalkulacyjne lub wykresy. Każde pole typu Załącznik może zawierać dowolną liczbę załączników na rekord.
Hiperłącze	Przechowuje adres (ścieżkę) do zasobów znajdujących się w Internecie, sieci lokalnej lub na dysku komputera.
Kreator odnośników	Nie jest faktycznym typem danych, pozwala jednak uruchomić kreator, za pomocą którego można wybrać wartości z innej tabeli lub listy.

Wybrane znaki stosowane do definiowania masek wprowadzania

Znak	Opis
0	Użytkownik musi wprowadzić cyfrę.
9	Użytkownik może wprowadzić cyfrę.
L	Użytkownik musi wprowadzić literę.
A	Użytkownik musi wprowadzić cyfrę lub literę.
?	Użytkownik może wprowadzić literę.
#	Użytkownik może wprowadzić cyfrę, spację, znak plus lub minus.
>	Wszystkie znaki po tym symbolu wyświetlane są jako duże litery.
<	Wszystkie znaki po tym symbolu wyświetlane są jako małe litery.
&	Użytkownik musi wprowadzić dowolny znak lub spację.

3. Jak zaprojektować i utworzyć bazę danych?

Kiedy tworzysz nową bazę danych, postępuj według następujących kroków.

1. Zastanów się, w jakim celu tworzysz bazę danych, jakie informacje chcesz w niej przechowywać, a także jakie informacje będą potrzebne w przyszłości.
2. Zaplanuj tabele opisujące poszczególne obiekty w twojej bazie. Pamiętaj, że jedna tabela powinna opisywać tylko jeden rodzaj obiektu – np. adresy, uczniów, sklepy, modele samochodów.
3. W każdej tabeli jedna kolumna powinna opisywać pojedynczą cechę obiektu, np. kod pocztowy, ale nie kod pocztowy i miasto. Imię i nazwisko również wprowadzaj w odrębnych kolumnach.
4. Zaplanuj, jakiego typu dane będą przechowywane w poszczególnych atrybutach (kolumnach) każdej z tabel. Przykładowe typy to tekst, data, liczba.
5. Upewnij się, że te same dane nie znajdują się w dwóch różnych tabelach lub polach tabeli.
6. Utwórz nową, pustą bazę w programie, z którego korzystasz, np. Microsoft Access.
7. Utwórz tabele według stworzonego planu, nadaj nazwy poszczególnym kolumnom (atrybutom) oraz określ, jakiego typu dane będą w nich przechowywane.
8. Dodaj testowe dane w każdej z tabel.
9. W razie potrzeby wykorzystaj maski wprowadzania danych dla wybranych pól, aby ograniczyć możliwość popełnienia błędu.
10. Utwórz relacje między tabelami, korzystając np. z kreatora odnośników. Przetestuj działanie odnośników: w widoku arkusza danych dodaj, usuń i zmodyfikuj dane w każdej z dostępnych tabel.
11. W razie potrzeby wprowadź modyfikacje dla typów danych przechowywanych w poszczególnych polach tabel i ponownie je przetestuj.
12. Utwórz kwerendy ułatwiające korzystanie z bazy danych.
13. Zacznij używać swojej bazy.

Wybrane systemy zarządzania bazami danych:

- Microsoft Access (<https://products.office.com/pl-pl/access>)
- LibreOffice Base (<https://www.libreoffice.org/>)
- Kexi (<http://kexi-project.org/>)
- Symphytum (<https://github.com/giowck/symphytum>)

Wybrane systemy zarządzania bazami danych pracujące w architekturze klient–serwer:

- MySQL
- Microsoft SQL Server
- Oracle
- PostgreSQL

4. Sposoby zapisu algorytmów

Algorytm to opis postępowania prowadzącego do rozwiązania problemu. Istotą algorytmu jest jego uniwersalność, czyli to, że służy do rozwiązywania wielu zadań tego samego rodzaju. Na przykład szkolny algorytm mnożenia pisemnego możesz zastosować dla dowolnych dwóch liczb, np. $21 \cdot 57$, $123 \cdot 87$. Algorytm musi być precyzyjny, choć może mieć różną formę.

Opis słowny

Algorytmy mogą być opisane słownie. Przykładem może być instrukcja obsługi kuchennego minutnika: „Wykonać pełen obrót wskaźnika aż do pozycji 0. Następnie ustawić wskaźnik na wymaganą liczbę minut”.

Lista kroków

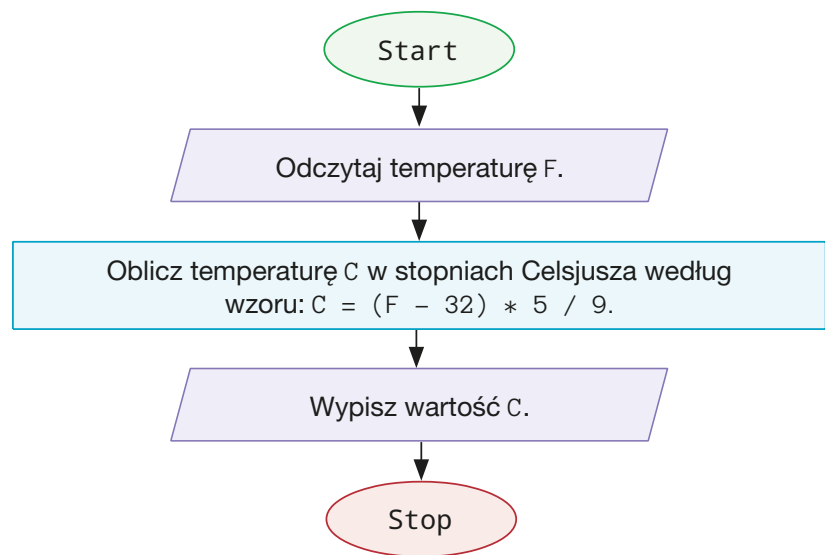
Lista kroków algorytmu to opis postępowania podzielony na poszczególne kroki, które mogą się odwoływać do siebie nawzajem.

Oto lista kroków algorytmu zaplatania warkocza klasycznego:

- 1 Rozdziel włosy na trzy pasma.
- 2 Dopóki nie spleciesz warkocza po końcówki włosów, wykonuj krok 3 i krok 4.
- 3 Przełóż lewe pasmo nad środkowym. Teraz to pasmo staje się środkowym.
- 4 Przełóż prawe pasmo nad środkowym. Teraz to pasmo staje się środkowym.

Schemat blokowy (sieć działań)

Schemat blokowy algorytmu jest zbudowany z umownych kształtów oraz strzałek pokazujących kierunek wykonywania operacji. Oto przykład schematu blokowego algorytmu przeliczania temperatury ze skali Fahrenheita na skalę Celsjusza:



Pseudokod (pseudojęzyk)

Pseudokod to zapis przypominający kod języka programowania, ale pozbawiony szczegółów związanych z implementacją w języku programowania. Przedstawienie algorytmu w postaci pseudokodu ułatwia później zapisanie tego algorytmu w języku programowania. Nie istnieje jedna, powszechnie przyjęta konwencja zapisu pseudokodu. Podstawową zasadą jest jednak wybór intuicyjnych oznaczeń i ich konsekwentne wykorzystanie. Oto pseudokod algorytmu wyznaczania części całkowitej $\sqrt{n}$ dla dodatniej liczby całkowitej n :

```
wczytaj n
p ← 1
dopóki (p + 1) · (p + 1) ≤ n wykonuj p ← p + 1
wypisz p
```

W powyższym zapisie strzałka ($\leftarrow$) oznacza instrukcję przypisania wartości do zmiennej. Symbole $+$, $\cdot$ i $\leq$ mają takie samo znaczenie jak w matematyce.

Kod źródłowy

Kod źródłowy to sformalizowany zapis algorytmu zgodny z regułami wybranego języka programowania. Kody źródłowe zazwyczaj zapisuje się w plikach tekstowych. Oto przykłady kodów źródłowych algorytmu obliczania całkowitego pierwiastka kwadratowego z liczby n , zapisane w językach Python i C++:

Python	C++
<pre>1. n = int(input()) 2. p = 1 3. while (p + 1)*(p + 1) <= n: 4. p = p + 1 5. print(p)</pre>	<pre>1. int main() 2. { 3. int n; cin >> n; 4. int p = 1; 5. while ((p + 1)*(p + 1) <= n) 6. { 7. p = p + 1; 8. } 9. cout << p << endl; 10. return 0; 11. }</pre>

Języki programowania, podobnie jak języki naturalne, mają słownictwo podstawowe, czyli **słowa kluczowe**, oraz **składnię**, czyli reguły tworzenia wypowiedzi.

Język programowania służy do porozumiewania się człowieka i komputera. W tej komunikacji pośredniczą specjalne **translatory**, czyli programy tłumaczące, gdyż docelowo kod źródłowy musi być przetłumaczony na **kod maszynowy**, zapisany binarnie, właściwy dla architektury procesora (CPU) i systemu operacyjnego.

5. Logika matematyczna i operatory logiczne

Zdania logiczne

Zdaniem logicznym jest każde zdanie, któremu możemy przyporządkować jednoznacznie wartość logiczną, czyli jedną z dwóch możliwych ocen: prawdę (ang. *true*) albo fałsz (ang. *false*).

Przykłady zdań logicznych:

- ▶ Liczba 8 jest liczbą pierwszą. (zdanie fałszywe)
- ▶ $2 + 2 < 5$ (zdanie prawdziwe)
- ▶ $2 + 8 = 20$ (zdanie fałszywe)
- ▶ Liczba parzysta jest podzielna przez 2. (zdanie prawdziwe)

Tradycyjnie przyjęło się, że prawdzie odpowiada wartość 1, a fałszowi – wartość 0.

Formy zdaniowe i zdania złożone

Formą zdaniową jest każde zdanie, które zawiera zmienne i które staje się zdaniem logicznym z chwilą zastąpienia zmiennych konkretnymi wartościami. Formie zdaniowej nie można więc jednoznacznie przypisać wartości logicznej, gdyż ta wartość zależy od wyboru wartości dla zmiennych.

Przykłady form zdaniowych:

- ▶ Liczba x jest podzielna przez 3. (np. dla $x = 6$ zdanie jest prawdziwe, a dla $x = 7$ – fałszywe)
- ▶ $y < 4$ (np. dla $y = 0$ zdanie jest prawdziwe, a dla $y = 4$ – fałszywe)
- ▶ Liczba z jest liczbą złożoną. (np. dla $z = 5$ zdanie jest fałszywe, a dla $z = 12$ – prawdziwe)

Zdania logiczne łączymy w zdania złożone, a formy zdaniowe – w formy zdaniowe złożone. Najczęściej używane konstrukcje to zaprzeczenie (negacja), alternatywa oraz koniunkcja.

Zaprzeczenie (negacja)

Zaprzeczenie to zdanie logiczne, które ma postać: *nieprawda, że p* , gdzie p jest zdaniem logicznym. Negacją może być też forma zdaniowa o postaci *nieprawda, że $p(x)$* , gdzie $p(x)$ jest formą zdaniową. Zaprzeczenie zdania p zapisuje się często jako $\sim p$.

p	$\sim p$
$x < 4$	$x \geq 4$
$2 + 2 = 5$	$2 + 2 \neq 5$

Alternatywa

Alternatywą jest połączenie zdań p i q spójnikiem „lub”, które ma postać: $p \text{ lub } q$. Może to być także połączenie form zdaniowych $p(x)$ i $q(x)$ w postaci: $p(x) \text{ lub } q(x)$.

p	q	$p \text{ lub } q$
$x < 4$	$x > 9$	$x < 4 \text{ lub } x > 9$
y jest liczbą parzystą.	z jest liczbą nieparzystą.	y jest liczbą parzystą lub z jest liczbą nieparzystą.

Zdanie złożone zwane alternatywą jest – jako całość – fałszywe jedynie wtedy, gdy tworzące je zdania proste są jednocześnie fałszywe. W pozostałych przypadkach alternatywa jest prawdziwa.

Koniunkcja

Koniunkcja to połączenie zdań p i q spójnikiem „i”, które ma postać: $p \text{ i } q$. Może to być także połączenie form zdaniowych $p(x)$ i $q(x)$ w postaci: $p(x) \text{ i } q(x)$.

p	q	$p \text{ i } q$
$x > 0$	$x < 4$	$x > 0 \text{ i } x < 4$
y jest liczbą pierwszą.	z jest liczbą złożoną.	y jest liczbą pierwszą i z jest liczbą złożoną.

Koniunkcja jest zdaniem prawdziwym jedynie wtedy, gdy tworzące ją zdania proste są jednocześnie prawdziwe. W pozostałych przypadkach koniunkcja jest fałszywa.

Tabela wartości logicznych

Wszystkie możliwe przypadki wartości logicznych zdań składowych i odpowiadające im wartości logiczne zdań złożonych można zestawzić w tabeli:

p	q	$\sim p$	$p \text{ lub } q$	$p \text{ i } q$
1	0	0	1	0
0	1	1	1	0
1	1		1	1
0	0		0	0

Operatory porównań (relacji) w językach Python i C++

Porównywanie dwóch wartości w językach programowania wykonuje się za pomocą operatorów porównań (relacji). Wynikiem ich działania są wartości True albo False w języku Python lub true albo false w języku C++.

Zapis w matematyce	Zapis w językach Python i C++
=	==
$\neq$	!=
>	>
<	<
$\geq$	>=
$\leq$	<=

Operatory logiczne w językach Python i C++

Spójniki logiczne („i”, „lub”, „nieprawda, że”) nazywa się operatorami logicznymi. Umożliwiają one stosowanie wyrażeń logicznych w tworzonym kodzie. Można je łączyć z operatorami porównań przedstawionymi na poprzedniej stronie.

Nazwa operatora logicznego	Zapis w języku Python	Przykład w języku Python
AND (koniunkcja)	and	Dla zmiennej x o wartości 3 i zmiennej y o wartości 2 wyrażenie <code>x < 5 and y > 1</code> ma wartość True.
OR (alternatywa)	or	Dla zmiennej x o wartości 5 i zmiennej y o wartości 5 wyrażenie <code>x == 5 or y == 5</code> ma wartość True.
NOT (negacja)	not	Dla zmiennej x o wartości 3 wyrażenie <code>not x == 4</code> ma wartość True.

W języku C++ zamiast operatorów `&&`, `||` oraz `!` można używać odpowiednio operatorów: `and`, `or` i `not`.

Nazwa operatora logicznego	Zapis w języku C++	Przykład w języku C++
AND (koniunkcja)	&&	Dla zmiennej x o wartości 3 i zmiennej y o wartości 2 wyrażenie <code>x < 5 && y > 1</code> ma wartość true.
OR (alternatywa)		Dla zmiennej x o wartości 5 i zmiennej y o wartości 5 wyrażenie <code>x == 5 y == 5</code> ma wartość true.
NOT (negacja)	!	Dla zmiennej x o wartości 3 wyrażenie <code>!(x == 4)</code> ma wartość true.

6. Typy danych w języku Python

W języku Python typ zmiennych określany jest na podstawie wartości, które są przypisane zmiennym.

Typy liczbowe

Wśród typów liczbowych wyróżnia się typy całkowite i zmiennoprzecinkowe (zmiennopozycyjne), pozwalające zapisać również ułamki (liczby wymierne). W programowaniu wartości zapisane jako 2 i 2.0 to liczby różnych typów. Pierwsza z nich jest typu całkowitego, a druga jest typu zmiennoprzecinkowego, ponieważ zawiera zapis części ułamkowej.

Nazwa typu	Przeznaczenie zmiennej	Przykład przypisania wartości
int	Liczby całkowite	a = 4
float	Ułamki dziesiętne i ich przybliżenia	a = -0.5
float	Liczby z częścią ułamkową	a = 2.45123 a = 4.00000023e-12

Uwaga: Zapis e-12 oznacza notację wykładniczą (tj. czynnik 10⁻¹²).

Typ logiczny

Do deklarowania zmiennych logicznych używa się typu bool, który przyjmuje wartości True i False. Są one słowami kluczowymi języka Python. Wartość False reprezentowana jest w pamięci komputera przez liczbę 0, a wartość True – przez liczbę 1.

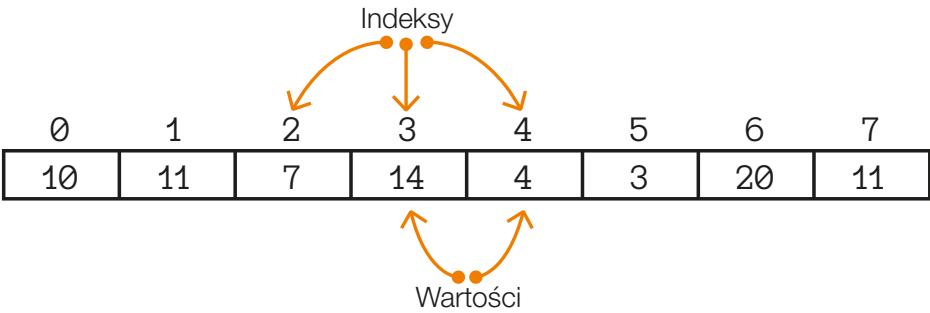
Typ znakowy i typ napisowy

Napisy, czyli łańcuchy znaków, Python zapisuje w pamięci komputera z użyciem typu str. Podczas przypisania wartości do zmiennej stosuje się cudzysłowy (napis = "ABC") albo pary apostrofów (napis = 'ABC').

Napisy można łączyć za pomocą operatora złączania (konkatenacji) +: np. kod napis = "ABC" + "DE" skutkuje zapamiętaniem napisu ABCDE.

Listy

Lista to struktura danych grupująca elementy dowolnych typów w sposób uporządkowany. Na przykład dzięki utworzeniu listy L = [1, 2, 3, 4] można zastąpić cztery zmienne typu całkowitego. Listy upraszczają proces kodowania algorytmów, gdyż można odwołać się do wielu wartości za pomocą jednej nazwy z liczbowymi indeksami. Dostęp do elementów zapisanych na liście uzyskuje się za pomocą mechanizmu indeksowania, tj. podania pozycji elementu na liście, licząc od 0, np. L[0], L[1], L[2] i L[3].



7. Podstawowe typy instrukcji w języku Python

Instrukcja warunkowa

Składnia prostej instrukcji warunkowej:

```
if warunek:
    instrukcja1
    instrukcja2
    instrukcja3
...
```

Warunek podawany po słowie `if` powinien być wyrażeniem typu logicznego (patrz: dodatek 5).

Składnia instrukcji warunkowej w postaci alternatywy:

```
if warunek:
    instrukcja1
    instrukcja2
...
else:
    instrukcja1
    instrukcja2
...
```

W przypadku zagnieżdżania instrukcji warunkowych trzeba zadbać o to, aby kolejne wcięcia były wielokrotnością pojedynczego wcięcia.

```
if warunek1:
    instrukcja
    if warunek2:
        instrukcja
    else:
        instrukcja
else:
    instrukcja
```

Oto przykładowe wykorzystanie instrukcji warunkowej:

```
if artykuł == "buty":
    if kolor == "zielony":
        rabat = 0.5
    else:
        rabat = 0.2
else:
    rabat = 0.1
```

Instrukcje powtarzania. Pętla `for`

Składnia pętli `for` wykonującej iterację według elementów listy ma postać:

```
for k in nazwa_listy:
    instrukcja1
    instrukcja2
...
```

Przykład: wypisanie elementów listy `[1, 3, 5]` z użyciem pętli `for`.

```
liczby = [1, 3, 5]
for k in liczby:
    print(k)
```

Składnia pętli `for` wykonującej iterację według liczb z zakresu:

```
for k in range(zakres):
    instrukcja1
    instrukcja2
...
```

Przykład: wypisanie liczb od 1 do 10 z użyciem pętli `for`.

```
for k in range(1, 11):
    print(k)
```

Instrukcje powtarzania. Pętla `while`

Składnia pętli `while` ma postać:

```
while warunek_logiczny:
    instrukcja1
    instrukcja2
...
```

Instrukcję `while` stosujemy wówczas, gdy nie znamy dokładnej liczby powtórzeń (cykli). Program wykonuje tę instrukcję, dopóki warunek jest spełniony. Jest więc możliwe, że nie wykona jej ani razu.

Oto przykładowe wykorzystanie pętli `while`:

```
imie = "Joanna"
proba = ""
i = 0

print("Spróbuj odgadnąć moje imię")
while proba != imie:
    i = i + 1
    print("Próba", i, ": ", end="")
    proba = input()

print("Brawo! Udało ci się odgadnąć moje imię za", i, "razem.")
```

Funkcje

Funkcje to podprogramy zawarte w programie głównym, które odpowiadają za realizację określonej części całego programu.

Budowa funkcji:

```
def nazwa_funkcji(argument1, argument2, ...):  
    instrukcja1  
    instrukcja2  
    ...  
    return wynik
```

Definicja funkcji składa się z nagłówka funkcji i bloku instrukcji. Nagłówek zawiera nazwę funkcji oraz listę nazw argumentów funkcji. Wartość zwracana przez funkcję jest wskazana po słowie `return`. Na przykład instrukcja `return a` oznacza, że funkcja zwróci wartość zmiennej `a`.

Gdy w definicji funkcji instrukcja `return` się nie pojawia, funkcja wykonuje zadania, które nie wymagają zwrócenia wartości wynikowej.

Oto przykład funkcji `PoleTrapezu`, która oblicza pole trapezu dla podanych długości podstaw i wysokości:

```
def PoleTrapezu(a, b, h):  
    return (a+b)*h/2  
  
print("*** Program Pole trapezu *** \n")  
a = float(input("Podaj długość jednej podstawy trapezu: "))  
b = float(input("Podaj długość drugiej podstawy trapezu: "))  
h = float(input("Podaj wysokość trapezu: "))  
  
print("Pole trapezu wynosi", PoleTrapezu(a, b, h))
```

8. Operatory i funkcje matematyczne w języku Python

Operatory arytmetyczne

Operator	Nazwa operatora	Wyrażenie	Wynik
+	Dodawanie	2 + 3 2 + 3.5	5 5.5
-	Odejmowanie	3 - 2 3.5 - 2	1 1.5
*	Mnożenie	2 * 3 2.5 * 3.5	6 8.75
**	Potęgowanie	2 ** 3	8
/	Dzielenie	3 / 2 3.0 / 2	1.5 1.5
//	Dzielenie całkowite	3 // 2 -3 // 2	1 -2
%	Reszta z dzielenia (modulo)	3 % 2	1

Wybrane wbudowane funkcje matematyczne

Działanie	Wyrażenie	Wynik	Uwagi
Potęgowanie	<code>pow(2.0, -2)</code>	0.25	Drugi argument funkcji to wykładnik potęgi.
Wartość bezwzględna	<code>abs(2)</code> <code>abs(-2)</code>	2 2	
Konwersja na liczbę typu całkowitego	<code>int(12.5)</code> <code>int("12")</code>	12 12	
Zaokrąglenie	<code>round(2.5)</code> <code>round(2.6)</code> <code>round(3.5)</code> <code>round(3.6)</code>	2 3 4 4	Liczby z częścią ułamkową 0,5 program zaokrągla inaczej dla liczb o części całkowitej parzystej i nieparzystej.

Wybrane funkcje matematyczne z biblioteki `math`

Działanie	Wyrażenie	Wynik	Uwagi
Pierwiastek kwadratowy	<code>sqrt(2.0)</code>	1.41421	Zwracana jest wartość przybliżenia.
Logarytm dziesiętny	<code>log10(2.0)</code>	0.30103	Zwracana jest wartość przybliżenia.
Zaokrąglenie w górę (sufit)	<code>ceil(2.5)</code> <code>ceil(-2.5)</code>	3 -2	
Zaokrąglenie w dół (podłoga)	<code>floor(2.5)</code> <code>floor(-2.5)</code>	2 -3	

9. Słowa kluczowe w języku Python

Słowa kluczowe nie mogą być nazwami zmiennych, funkcji ani innych obiektów w kodzie źródłowym.

Słowo kluczowe	Opis
and	Operator logiczny
def	Używane w definicji funkcji
else	Używane w instrukcji warunkowej do wyodrębnienia kodu wykonywanego, gdy warunek po if nie jest spełniony
for	Używane w instrukcji iteracyjnej
from	Używane przy imporcie fragmentu biblioteki (modułu)
if	Używane w instrukcji warunkowej
import	Używane przy imporcie biblioteki (modułu)
in	Używane do odwołania się do elementów listy
is	Używane do sprawdzania, czy dwa obiekty są tymi samymi obiektami, np. czy zwracana wartość jest równa True
not	Operator logiczny
or	Operator logiczny
return	Używane w instrukcji, która pozwala funkcji zwrócić wartość do miejsca jej wywołania
True	Stała oznaczająca wartość logiczną prawdą
while	Używane w instrukcji iteracyjnej
False	Stała oznaczająca wartość logiczną fałsz

Inne słowa kluczowe języka Python

as	assert	break	class
continue	del	elif	except
field	finally	global	lambda
None	nonlocal	pass	raise
try	with		

10. Typy danych w języku C++

W języku C++ każda zmienna, w zależności od tego, czy jest liczbą całkowitą, liczbą z częścią ułamkową, znakiem, napisem czy wartością logiczną, musi mieć jawnie przypisany odpowiedni typ danych. Określa on sposób zapisu zmiennej w pamięci komputera oraz zbiór operacji, które można wykonać na tej zmiennej.

Typy liczbowe

Wśród typów liczbowych wyróżnia się typy całkowite oraz zmiennoprzecinkowe (zmiennopozycyjne), czyli takie, które pozwalają zapisać liczby rzeczywiste. W programowaniu wartości zapisane jako 2 i 2.0 to liczby różnych typów. Pierwsza z nich jest typu całkowitego, a druga jest typu zmiennoprzecinkowego, ponieważ zawiera zapis części ułamkowej.

Nazwa typu	Przeznaczenie zmiennej	Deklaracja zmiennej w języku C++
int	Liczby całkowite z zakresu od -2 147 483 648 do 2 147 483 647	int a = 4;
long long	Duże liczby całkowite nie większe niż $9,22 \cdot 10^{18}$	long long a = 300000000000;
float	Ułamki dziesiętne i ich przybliżenia	float a = -0.5;
float	Liczby z częścią ułamkową, w których liczba cyfr znaczących (pewnych) jest nie większa niż 6	float a = 2.45123;
double	Bardzo małe liczby, które trzeba zapamiętać z większą dokładnością (nie więcej niż 15 cyfr znaczących)	double a = 4.00000023e-12;
double	Bardzo duże liczby (z częścią ułamkową) lub liczby, które trzeba zapamiętać z większą dokładnością (nie więcej niż 15 cyfr znaczących)	double a = 4.00000023e12;

Uwaga: Zapisy typu e12 oraz e-12 oznaczają notację wykładniczą (tj. odpowiednio czynniki 10^{12} oraz 10^{-12}).

Typ znakowy i typ napisowy

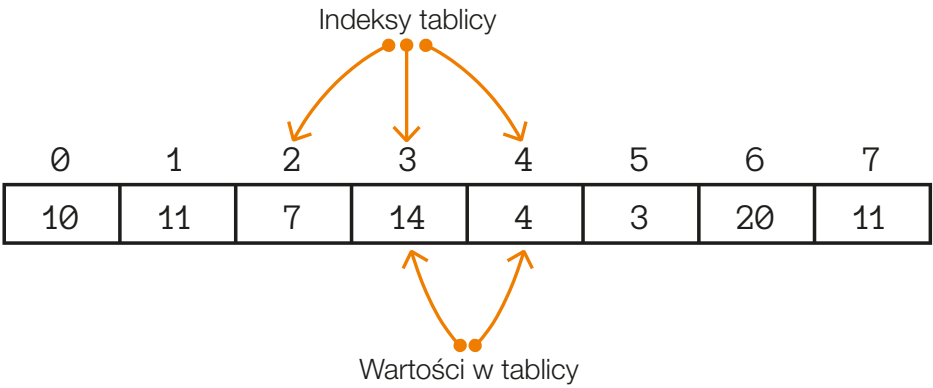
Pojedyncze znaki można zapisywać w zmiennych typu char. Na przykład deklaracja char litera='a'; to deklaracja zmiennej, w której następnie zapisano literę a. Napisy, czyli łańcuchy znaków, zapisuje się z użyciem zmiennych typu string. Typ ten w sensie ścisłym nie jest typem prostym C++, ale ma charakter obiektowy, tzn. po deklaracji zmiennej uzyskujemy dostęp do wbudowanych funkcji (tzw. metod), które pozwalają łatwo wykonać operacje na napisach. Dla zmiennej zadeklarowanej jako string napis="ABC"; liczbę znaków napisu wypiszemy na ekranie, jeśli użyjemy w kodzie zapisu cout<<napis.size().

Typ logiczny

Do deklarowania zmiennych logicznych używa się typu `bool`, który przyjmuje wartości `true` i `false`. Są to słowa kluczowe języka C++. Wartość `false` reprezentowana jest w pamięci komputera przez liczbę 0, a wartość `true` – przez liczbę różną od 0.

Tablice

Tablica to struktura danych grupująca elementy określonego typu w sposób uporządkowany. Na przykład deklaracja `int tablica [4];` zastępuje deklarację czterech zmiennych typu `int`.



Tablice upraszczają proces zapisywania algorytmów w języku programowania, gdyż można odwołać się do wielu wartości za pomocą jednej nazwy z liczbowymi indeksami. Dostęp do poszczególnych elementów zapisanych w tablicy uzyskuje się za pomocą mechanizmu indeksowania, tj. podania pozycji elementu w tablicy, licząc od 0, np. `tablica[0]`, `tablica[1]`, `tablica[2]` i `tablica[3]`.

Zapis	Wyjaśnienie
<code>int tablica[4];</code>	Deklaracja tablicy czterech liczb typu <code>int</code>
<code>const int n = 4;</code> <code>int tablica[n];</code>	Deklaracja tablicy, której liczbę elementów określa stała <code>n</code>
<code>int tablica[3] = {1,2,3};</code>	Deklaracja tablicy trzech liczb typu <code>int</code> połączona z przypisaniem im wartości
<code>int tablica[] = {1,2,3};</code>	Deklaracja tablicy liczb typu <code>int</code> połączona z przypisaniem im wartości – rozmiar tablicy wyznacza się na podstawie liczby wartości
<code>int tablica[4] = {0};</code>	Deklaracja tablicy czterech liczb typu <code>int</code> połączona z przypisaniem im wszystkim wartości 0
<code>int tablica[4] = {1, 2};</code>	Deklaracja tablicy czterech liczb typu <code>int</code> połączona z przypisaniem im wartości: 1, 2, 0 i 0

11. Podstawowe typy instrukcji w języku C++

Instrukcja warunkowa

Składnia prostej instrukcji warunkowej:

Pojedyncza instrukcja	Wiele instrukcji
<code>if (warunek) instrukcja;</code>	<code>if (warunek)</code> <code>{</code> <code>instrukcja1;</code> <code>instrukcja2;</code> <code>...</code> <code>}</code>

Warunek podawany po słowie `if` powinien być wyrażeniem typu logicznego (patrz: dodatek 5).

Składnia instrukcji warunkowej w postaci alternatywy:

Pojedyncza instrukcja	Wiele instrukcji
<code>if (warunek)</code> <code>instrukcja;</code> <code>else</code> <code>instrukcja;</code>	<code>if (warunek)</code> <code>{</code> <code>instrukcja1;</code> <code>instrukcja2;</code> <code>...</code> <code>}</code> <code>else</code> <code>{</code> <code>instrukcja1;</code> <code>instrukcja2;</code> <code>...</code> <code>}</code>

W przypadku zagnieżdżania instrukcji warunkowych warto zadbać o odpowiednie wcięcia.

Instrukcje powtarzania. Pętla for

Składnia pętli `for` powtarzającej pojedynczą instrukcję:

`for (licznik; warunek powtarzania; krok) instrukcja;`

Składnia pętli `for` powtarzającej wiele instrukcji:

```
for (licznik; warunek powtarzania; krok)
{
    instrukcja1;
    instrukcja2;
    ...
}
```


Instrukcje powtarzania. Pętla while

Składnia pętli while powtarzającej pojedynczą instrukcję:

while (warunek logiczny) instrukcja;

Składnia pętli while powtarzającej wiele instrukcji:

```
while (warunek logiczny)
{
    instrukcja1;
    instrukcja2;
    ...
}
```

Instrukcja iteracyjna. Pętla do while

Składnia pętli do while powtarzającej pojedynczą instrukcję:

do instrukcja while (warunek logiczny);

Składnia pętli do while powtarzającej wiele instrukcji:

```
do
{
    instrukcja1;
    instrukcja2;
    ...
} while (warunek logiczny);
```

Instrukcje sterujące. Funkcje

Funkcje to podprogramy zawarte w programie głównym, które odpowiadają za realizację określonej części całego programu. Funkcje definiuje się przed definicją funkcji main.

Definicja funkcji składa się z nagłówka funkcji i bloku instrukcji. Nagłówek zawiera typ wartości zwracanej przez funkcję, nazwę funkcji oraz listę argumentów wraz z typami i nazwami argumentów. Budowa funkcji:

```
typ nazwa_funkcji(typ argument1, typ argument2, ...)
{
    instrukcja1;
    instrukcja2;
    ...
    return wynik;
}
```

Wartość zwracana przez funkcję jest wskazana po słowie return.

Na przykład instrukcja return a; oznacza, że funkcja zwróci wartość zmiennej a.

Gdy typem funkcji w nagłówku jest void, funkcja wykonuje zadania, które nie wymagają zwrócenia wartości wynikowej.

12. Operatory i funkcje matematyczne w języku C++

Operatory arytmetyczne

Operator	Nazwa operatora	Wyrażenie	Wynik
+	Dodawanie	2 + 3 2 + 3.5	5 5.5
-	Odejmowanie	3 - 2 3.5 - 2	1 1.5
*	Mnożenie	2 * 3 2.5 * 3.5	6 8.75
/	Dzielenie	3 / 2 -3 / 2 3.0 / 2	1 -1 1.5
%	Reszta z dzielenia (modulo)	3 % 2	1

W języku C++ nie ma operatora potęgowania. Kwadrat lub sześćcian liczby najłatwiej zapisać jako iloczyn, odpowiednio: 2 * 2 lub 2 * 2 * 2.

Do zapisywania potęg można też użyć funkcji pow z biblioteki cmath.

Wybrane funkcje matematyczne z biblioteki cmath

Działanie	Wyrażenie	Wynik	Uwagi
Potęgowanie	pow(2.0; -2)	0.25	Drugi argument funkcji to wykładnik potęgi. Może być liczbą niecałkowitą.
Pierwiastek kwadratowy	sqr(2.0)	1.41421	Zwracana jest wartość przybliżenia.
Pierwiastek sześcienny	cbrt(2.0)	1.25992	Zwracana jest wartość przybliżenia.
Zaokrąglenie w górę (sufit)	ceil(2.5) ceil(-2.5)	3 -2	
Zaokrąglenie w dół (podłoga)	floor(2.5) floor(-2.5)	2 -3	
Zaokrąglenie (tradycyjne)	round(2.5) round(-2.5)	3 -3	
Obcięcie części ułamkowej	trunc(2.5) trunc(-2.5)	2 -2	
Wartość bezwzględna	abs(2) abs(-2)	2 2	Można też stosować funkcję fabs z biblioteki math.h
Logarytm dziesiętny	log10(2.0)	0.30103	Zwracana jest wartość przybliżenia.

13. Słowa kluczowe w języku C++

Słowa kluczowe nie mogą być nazwami zmiennych, funkcji ani innych obiektów w kodzie źródłowym.

Słowo kluczowe	Opis
<code>bool</code>	Nazwa typu logicznego
<code>const</code>	Używane przy definiowaniu stałej
<code>double</code>	Nazwa typu dla liczb z częścią ułamkową, o podwójnej precyzji
<code>else</code>	Używane w instrukcji warunkowej do wyodrębnienia kodu wykonywanego, gdy warunek po <code>if</code> nie jest spełniony
<code>false</code>	Stała oznaczająca wartość logiczną fałsz
<code>float</code>	Nazwa typu dla liczb z częścią ułamkową
<code>for</code>	Używane w instrukcji iteracyjnej
<code>if</code>	Używane w instrukcji warunkowej
<code>int</code>	Nazwa typu dla liczb całkowitych
<code>long</code>	Nazwa typu dla liczb całkowitych, definicja z <code>long long int</code> pozwala zapisywać w pamięci komputera liczby z większego zakresu liczb
<code>namespace</code>	Używane przy określaniu przestrzeni nazw
<code>return</code>	Używane w instrukcji, która pozwala funkcji zwrócić wartość do miejsca jej wywołania
<code>true</code>	Stała oznaczająca wartość logiczną prawdą
<code>using</code>	Używane przy określaniu przestrzeni nazw
<code>void</code>	Używane w nagłówku funkcji, która nie zwraca wartości
<code>while</code>	Używane w instrukcji iteracyjnej

Inne słowa kluczowe języka C++

<code>asm</code>	<code>auto</code>	<code>break</code>	<code>case</code>
<code>catch</code>	<code>char</code>	<code>class</code>	<code>const_cast</code>
<code>continue</code>	<code>default</code>	<code>delete</code>	<code>do</code>
<code>dynamic_cast</code>	<code>enum</code>	<code>explicit</code>	<code>export</code>
<code>extern</code>	<code>friend</code>	<code>goto</code>	<code>inline</code>
<code>mutable</code>	<code>new</code>	<code>operator</code>	<code>private</code>
<code>protected</code>	<code>public</code>	<code>register</code>	<code>reinterpret_cast</code>
<code>short</code>	<code>signed</code>	<code>sizeof</code>	<code>static</code>
<code>static_cast</code>	<code>struct</code>	<code>switch</code>	<code>template</code>
<code>this</code>	<code>throw</code>	<code>try</code>	<code>typedef</code>
<code>typeid</code>	<code>typename</code>	<code>union</code>	<code>unsigned</code>
<code>virtual</code>	<code>volatile</code>	<code>wchar_t</code>	

14. Systemy liczbowe

Systemem liczbowym nazywamy zbiór reguł jednolitego zapisu liczb. Do zapisu liczb w danym systemie liczbowym używa się cyfr.

System dziesiętny

Podstawą systemu jest liczba 10. Do zapisu wykorzystujemy cyfry: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Liczbę w systemie dziesiętnym możemy przedstawić jako sumę liczb, z których każda jest iloczynem potęgi liczby 10 oraz liczby od 0 do 9.

System dwójkowy (binarny)

Podstawą systemu jest liczba 2. Do zapisu wykorzystujemy zbiór cyfr: {0, 1}. Liczbę w systemie binarnym możemy przedstawić jako sumę liczb, z których każda jest iloczynem potęgi liczby 2 oraz liczby 0 lub 1.

System szesnastkowy (heksadecymalny)

Podstawą systemu jest liczba 16. Do zapisu wykorzystujemy zbiór cyfr: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Liczbę w systemie szesnastkowym możemy przedstawić jako sumę liczb, z których każda jest iloczynem potęgi liczby 16 oraz liczby od 0 do F (gdzie: A = 10, B = 11, C = 12, D = 13, E = 14, F = 15).

Przeliczanie liczb pomiędzy systemami

Zamiana liczby zapisanej w systemie dziesiętnym na liczbę zapisaną w innym systemie liczbowym polega na jej wielokrotnym dzieleniu całkowitym przez podstawę danego systemu i zapisywaniu reszty z dzielenia. Można to zrobić w tabeli – liczbę po zamianie odczytujemy od dołu do góry. Ilustruje to poniższy przykład.

Zamiana zapisu liczby z systemu dziesiętnego na dwójkowy						Zamiana zapisu liczby z systemu dziesiętnego na szesnastkowy					
150	:	2	=	75	0	19374	:	16	=	1210	E
75	:	2	=	37	1	1210	:	16	=	75	A
37	:	2	=	18	1	75	:	16	=	4	B
18	:	2	=	9	0	4	:	16	=	0	4
9	:	2	=	4	1						
4	:	2	=	2	0						
2	:	2	=	1	0						
1	:	2	=	0	1						
150 ₁₀ = 10010110 ₂						19374 ₁₀ = 4BAE ₁₆					

Indeks

A

- algorytm 91, 151
- argument 96, 156
- atrybut 68
 - znacznika 10

B

- bajt 111, 172
- baza danych 66
- biblioteka `cmath` 188
- bit 105, 166
 - najbardziej znaczący 108, 169
 - najmniej znaczący 108, 169
- blok instrukcji 99, 160
- błąd składniowy 97, 157

C

- cechy poprawnego algorytmu 92, 152
- clickbait 44
- CMS (system zarządzania treścią) 30
- część BODY 11
 - HEAD 11

D

- dodawanie relacji między tabelami 82
- dostępność cyfrowa 28
- drzewo decyzyjne 108, 169

F

- framework 28
- funkcja 121, 183
 - `float` (Python) 131
 - `float` (C++) 194
 - `input` 96
 - `int` 99
 - `print` 96
 - `range` 114
 - `reversed` 114
 - `sqrt` (Python) 125
 - `sqrt` (C++) 188

H

- hashtag 44

I

- IDE 93, 153
- identyfikacja wizualna 55
- indeks 113, 174

- indeksowanie 22
- instrukcja 95, 155
 - `cin` 156
 - `cout` 156
 - powtarzania (instrukcja iteracyjna, pętla) 101, 162
 - `return` (Python) 121
 - `return` (C++) 183
 - warunkowa `if` (Python) 99
 - — `if` (C++) 159
 - wejścia 96, 156
 - wyjścia 96, 156
- interpreter 93, 153

J

- język C++ 93, 153
 - CSS 23
 - HTML 8
 - programowania 93, 153
 - Python 93, 153

K

- klucz główny (klucz podstawowy) 69
 - obcy 69
- kod wynikowy (kod maszynowy) 93, 153
 - źródłowy 93, 153
- kompilator 93, 153
- konspekt 48
- kreator odnośników 82
- krotka 142
- kwerenda 85

L

- leniwe wartościowanie 146, 208
- liczby pierwsze 117, 179
 - złożone 117, 179
- lid 44
- lista 113
 - kroków 91, 151
- logo 55

M

- manipulator `endl` 156
- maska wprowadzania 81
- metoda `append` 114
- model RGB 22
- modulo 112, 173
- moduł 125
 - `math` 125

O

- operacja pierwiastkowania 125, 188
 - prostej zamiany 140, 202
- operator przypisania 96
- organiczne wyniki wyszukiwania 44

P

- parametr funkcji 121, 183
- parametry formalny i aktualny funkcji 122, 184
- pętla `for` (Python) 114
 - `for` (C++) 175
 - `while` (Python) 101
 - `while` (C++) 162
- pętle zagnieżdżone 142, 204
- podcast 56
- podstawa systemu liczbowego 103, 164
- podstawowa struktura strony internetowej 11
- podstawowe twierdzenie arytmetyki 117, 179
 - własności liczb binarnych 104, 165
- pole rekordu 69
- pozycjonowanie 22
- projektowanie bazy danych 72
- publikowanie 41

R

- rekord 69
- relacja 70
 - typu jeden do jednego 71
 - — — — wielu 70
- rozkład na czynniki pierwsze 120, 182

S

- scalanie komórek tabeli 20
- screencast 56
- segment 36
- serwis internetowy (witryna) 8
- siatka 28
- składnia 97, 157
 - języka 8
- słowa kluczowe 99, 160
- sortowanie bąbelkowe 140, 202
 - (porządkowanie) danych 133, 195
 - przez wstawianie 143, 205
- specyfikacja zadania 95, 155

- strona internetowa 8
 - responsywna 27
- sygnet, logotyp 55
- system dwójkowy (binarny) 104, 165
 - dziesiętny (decymalny) 103, 164
 - ósemkowy 105, 166
 - pozycyjny 103, 164
 - szesnastkowy 105, 166
 - zarządzania bazą danych (SZBD) 75
- szum tła 60

Ś

- ścieżka bezwzględna 15
 - względna 15

T

- tabela bazy danych 68
 - programu Access 75
- tablica 174
- tekst informacyjny 48
- translator 93, 153
- tryb RGB 23
- tworzenie kwerendy 85
 - linku 39
 - programu 95, 155
 - strony internetowej 12
 - — — w programie Mobirise 35
 - tabeli w programie Access 75
 - zaproszenia w serwisie Canva 52

W

- webcast 56
- webinarium (webinar) 56
- widok projektu 79
- wizualne tworzenie treści 30
- wyszukiwanie binarne (połówkowe) 138, 200
 - sekwencyjne 138, 200

Z

- zapisywanie zmian w tabeli 80
- zasada 5W 49
- zmienna 96, 156
- znacznik otwierający 10
 - (tag) 8
 - zamykający 10

Zdjęcie na okładce: Shutterstock/Photobank.kiev.ua.

Zdjęcia:

Agencja Gazeta: Arkadiusz Scichocki s. 110, 171, Łukasz Giza s. 120, 182; **BE&W:** Alamy Stock Photo – Addictive Stock Creatives s. 59, urbanbuzz s. 118 (tokeny), s. 180 (tokeny); **Envato:** s. 16 (laptop, tablet, smartfon); **Getty Images:** DigitalVision Vectors – miakievy s. 17, E+ – Cecilie_Arcurs s. 64 (programista), courtneyk s. 34, filadendron s. 64 (stomatolog), Goodboy Picture Company s. 148 (staż w kancelarii), s. 210 (staż w kancelarii), izusek s. 64 (YouTuber), JK1991 s. 88 (praca w zespole), martin-dm s. 58, nadla s. 73, technotr s. 128, 190, iStock/Getty Images Plus – Galeanu Mihai s. 7, g-stockstudio s. 88 (kontroler finansowy), nadla s. 149, stefanamer s. 88 (dietetyczka), Moment/WeiQuan Lin s. 65, Science Photo Library/Richard Jones s. 78, Westend61 s. 88 (firma kurierska); **Shutterstock:** A_stockphoto s. 148 (inżynier pożarnictwa), s. 210 (inżynier pożarnictwa), alphaspirit s. 135, 197, April stock s. 50, Axel Bueckert s. 129 (człowiek), s. 191 (człowiek), Designertonmoy1 s. 141, 203, DmitryRazinkov s. 106 (fale dźwiękowe), s. 167 (fale dźwiękowe), Dmitry Kalinovsky s. 148 (geodeta), s. 210 (geodeta), feeling lucky s. 46 (dziewczyna przy komputerze), givaga s. 16 (piramidy, Sfinks), s. 25 (piramidy), grey_and s. 25 (laptop), Here s. 46–47 (niebieskie tło), s. 144, 206, Hulahop s. 25 (smartfon), itechno s. 118 (Euklides), s. 180 (Euklides), Khalil Uddin s. 25 (grafika z falami w tle), Klyaksun s. 25 (drewniany blat), mangpor2004 s. 24 (kobieta z tabletem), MUNGKHOOD STUDIO s. 95, NDAB Creativity s. 47 (młodzież), s. 64 (spotkanie zespołu), NicoElNino s. 84, PabloLagarto s. 47 (tło), Picsfive s. 46 (kartka), Rawpixel.com s. 45 (laptop), SAG stock s. 94 (klawiatura), Sayan Puangkham s. 24–25 (niebieskie tło), sdecoret s. 211, Sergey Chirkov s. 46 (lampa fotograficzna), SeventyFour s. 148 (pracownik w firmie produkującej meble), s. 210 (pracownik w firmie produkującej meble), SkillUp s. 94 (tło), s. 154 (tło), Stanisław Tokarski s. 140, 202, Stefano Garau s. 46 (aparat na statywie), tanatat s. 47 (lupa), Tatiana Shepeleva s. 129 (robot), s. 191 (robot), Titima Ongkantong s. 106 (fale dźwiękowe), s. 167 (fale dźwiękowe), Vmaster s. 45 (rakietą), Yuriy Lukin s. 106 (nagrywanie koncertu na telefon), s. 167 (nagrywanie koncertu na telefon), Zakharchuk s. 89, zffoto s. 118–119 (tło), s. 180–181 (tło); **ČSSZ** (Česká správa sociálního zabezpečení) s. 70.

Wydawnictwo dołożyło wszelkich starań, aby odnaleźć posiadaczy praw autorskich do wszystkich utworów zamieszczonych w podręczniku. Pozostałe osoby prosimy o kontakt z Wydawnictwem.

Wydawnictwo Nowa Era nie ponosi odpowiedzialności za treści i zasoby zamieszczone na cytowanych w tej publikacji stronach internetowych. Adresy stron internetowych zweryfikowano 15.01.2025 r.

Za co polubisz podręcznik *NOWA Informatyka na czasie 2?*

Powód 1



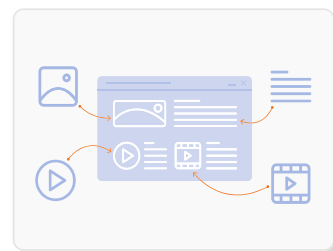
Jest tu dokładnie tyle treści, ile trzeba, nie ma nadmiaru. Ten podręcznik jest w pełni dostosowany, co oznacza, że wycieliśmy z niego treści, które od roku 2024 nie są już wymagane.



Powód 2



Dużo zrzutów ekranu i czytelne infografiki. Łatwo zorientujesz się, gdzie dokładnie znajdują się omawiane narzędzia i nauczysz się z nich korzystać.



Powód 3



Przykłady z życia codziennego – bo pokazują, jak możesz wykorzystać wiedzę w praktyce – na przykład zapisać rozgrzewkę sportową w formie algorytmu.

