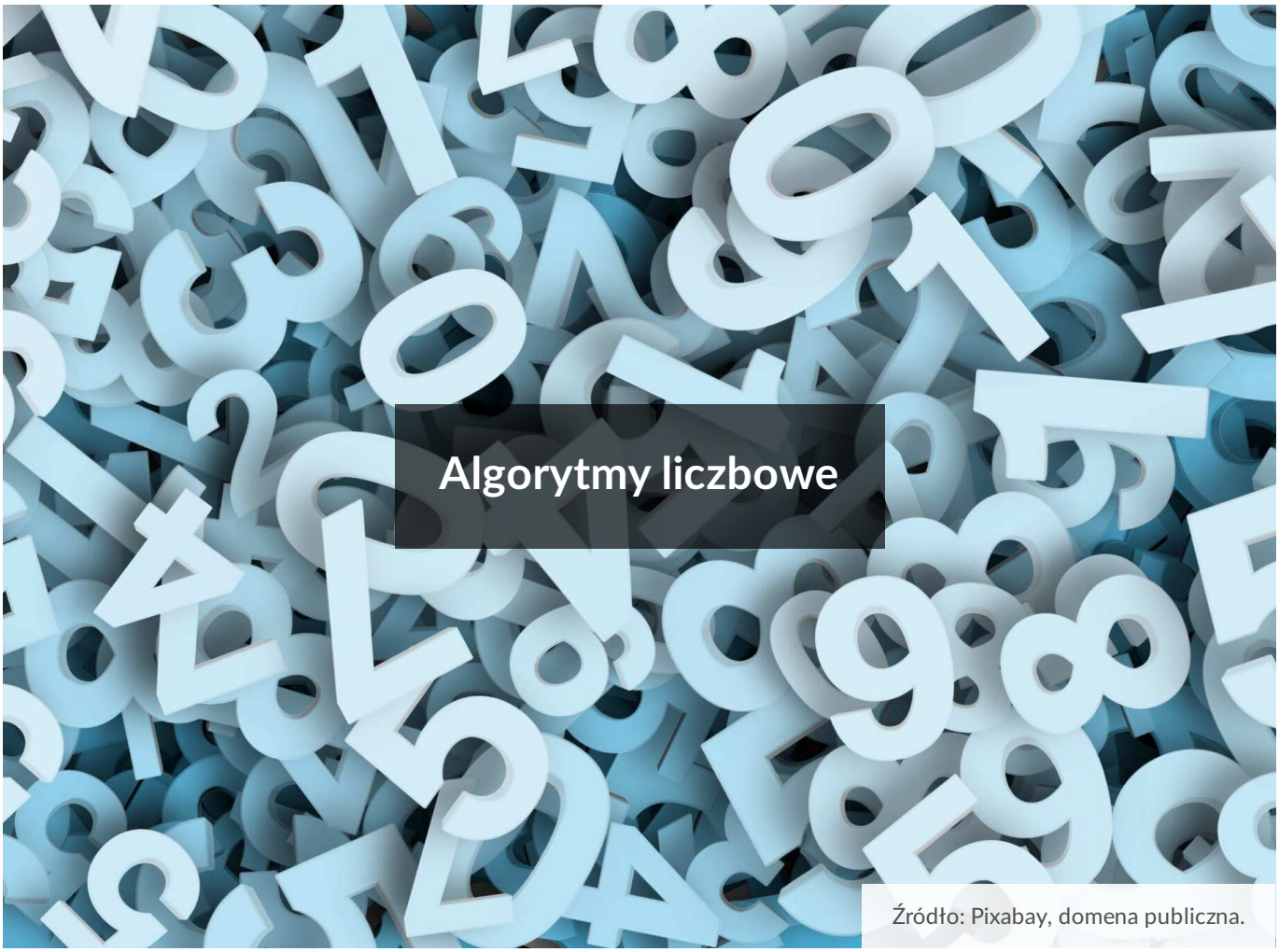
Algorytmy liczbowe



Źródło: Pixabay, domena publiczna.

Choć z pojęciem **algorytmu** spotykasz się najczęściej na lekcjach informatyki, nie znaczy to, że algorytmy nie mogą być wykorzystywane do rozwiązywania problemów z innych dziedzin.

Algorytmy stosowane do rozwiązywania problemów matematycznych za pomocą operacji na liczbach nazywamy algorytmami **liczbowymi** lub **numerycznymi.**

Jednym z problemów, który możemy rozwiązać, wykorzystując algorytmy liczbowe, jest sprawdzenie, czy dana liczba spełnia określone warunki.

Na lekcjach matematyki nauczyliśmy się, jak definiować liczby pierwsze czy liczby doskonałe. Wiemy zatem, że są to liczby, które spełniają określone warunki. To, czy dana liczba należy do konkretnego zbioru, weryfikujemy, korzystając z odpowiedniego algorytmu liczbowego.

Implementacja algorytmów liczbowych w poszczególnych językach programowania została przedstawiona w e-materiałach:

- Algorytmy liczbowe w języku C++,
- Algorytmy liczbowe w języku Java,
- Algorytmy liczbowe w języku Python.

Więcej zadań? Przejdź do Algorytmy liczbowe – zadania maturalne.

> **Twoje cele**
>
> - Wyjaśnisz, jakie właściwości musi posiadać dana liczba, aby mogła zostać nazwana liczbą doskonałą, zaprzyjaźnioną, bliźniaczą lub pierwszą.
> - Przeanalizujesz algorytmy, które pozwalają sprawdzić, czy dana liczba spełnia założone kryteria.
> - Rozwiążesz zadania sprawdzające wiedzę na temat algorytmów liczbowych.

Przeczytaj

Liczby pierwsze

Na początek przyjrzyjmy się grupie liczb nazywanych pierwszymi. Należą do nich np.:

- 2,
- 3,
- 5,
- 7,
- 11,
- 13,
- 17.

Na pierwszy rzut oka trudno zauważyć jakąkolwiek właściwość, która byłaby wspólna dla wszystkich wymienionych wyżej liczb. Łączy je jednak pewna zależność dotycząca ich dzielników.

Przypomnijmy więc – dana liczba jest liczbą pierwszą, jeśli spełnia dwa warunki:

- jest liczbą naturalną większą od 1,
- posiada tylko dwa dzielniki: 1 oraz samą siebie.

Problem 1

Spróbujmy napisać algorytm w postaci pseudokodu, który sprawdzi, czy dana liczba jest liczbą pierwszą.

Specyfikacja:

Dane:

- `liczba` – liczba do sprawdzenia; liczba naturalna

Wynik:

Program wypisuje komunikat `Liczba jest liczbą pierwszą` lub `Liczba nie jest liczbą pierwszą`.

```
liczba ← wprowadź liczbę naturalną
czyPierwsza ← prawda

jeżeli liczba <= 1:
    czyPierwsza ← fałsz
w przeciwnym razie:
    i ← 2
    dopóki i * i <= liczba:
        jeżeli liczba mod i = 0:
            czyPierwsza ← fałsz
            przerwij pętlę
        i ← i + 1

jeżeli czyPierwsza = prawda:
    wypisz "Liczba jest liczbą pierwszą"
w przeciwnym razie:
    wypisz "Liczba nie jest liczbą pierwszą"
```

Zaczynamy od wprowadzenia liczby, która ma zostać sprawdzona, i zapisania jej w zmiennej `liczba`. Następnie deklarujemy zmienną logiczną `czyPierwsza`, której wartość będzie wskazywać, czy dana liczba spełnia warunki liczby pierwszej, czy też nie.

> Kolejnym krokiem jest sprawdzenie w instrukcji warunkowej, czy wprowadzona liczba jest mniejsza lub równa 1. Jeśli ten warunek jest spełniony, dana liczba na pewno nie będzie liczbą pierwszą. Dlatego zmiennej logicznej `czyPierwsza` zostaje przypisana wartość `fałsz`.
>
> W przeciwnym przypadku następuje sprawdzenie dzielników danej liczby z przedziału $\left\langle 2, \sqrt{\text{liczba}} \right\rangle$. Jeżeli w zadanym przedziale zostanie znaleziony dzielnik, to liczba ta nie jest liczbą pierwszą. Sprawdzenie odbywa się w instrukcji warunkowej wewnątrz pętli. Wyrażenie `mod` oznacza operację modulo (w wielu językach programowania oznaczana symbolem %), czyli zwraca resztę z dzielenia liczby przez `i`.

Liczby doskonałe

Kolejną ciekawą grupą liczb są liczby doskonałe. Wyróżniająca je właściwość również dotyczy dzielników.

Aby daną liczbę naturalną można było nazwać doskonałą, suma jej dzielników właściwych musi być równa tej liczbie. Przykładami takich liczb są:

- 6,
- 28,
- 496,
- 8128.

Przyjrzyjmy się zatem dokładnie liczbie 6:

- dzielniki właściwe: 1, 2, 3.
- suma dzielników właściwych: 1 + 2 + 3 = 6.
- Czy jest to liczba doskonała? **TAK**

Podobnie sprawdzimy liczbę 28:

- dzielniki właściwe: 1, 2, 4, 7, 14.
- suma dzielników właściwych: 1 + 2 + 4 + 7 + 14 = 28.
- Czy jest to liczba doskonała? **TAK**

Problem 2

Napiszmy teraz algorytm w postaci pseudokodu, który sprawdzi, czy dana liczba jest liczbą doskonałą.

Specyfikacja:

Dane:

- `liczba` – liczba do sprawdzenia; liczba naturalna

Wynik:

Program wypisuje komunikat `Liczba jest liczbą doskonałą` lub `Liczba nie jest liczbą doskonałą`.

Zapisany za pomocą pseudokodu algorytm sprawdzający, czy dana liczba jest liczbą doskonałą, będzie wyglądał następująco:

```
liczba ← wprowadź liczbę naturalną
sumaDzielników ← 0

dla i = 1, 2, 3, ..., liczba - 1 wykonuj:
    jeżeli liczba mod i = 0:
        sumaDzielników ← sumaDzielników + i

jeżeli sumaDzielników = liczba wykonaj:
    wypisz "Liczba jest liczbą doskonałą"
w przeciwnym razie:
    wypisz "Liczba nie jest liczbą doskonałą"
```

Pierwszym krokiem jest wprowadzenie liczby, którą chcemy sprawdzić, i zapisanie jej wartości w zmiennej `liczba`. Następnie zostaje zadeklarowana zmienna `sumaDzielników`. Zmienna, jak wskazuje jej nazwa, będzie służyła do sumowania dzielników właściwych zadanej liczby.

Następnie zostaje zadeklarowana pętla, która iteruje po liczbach z przedziału $\langle 1, \text{liczba} - 1 \rangle$, sprawdzając, czy dana wartość `i` jest dzielnikiem sprawdzanej liczby. Jeżeli tak, to następuje

> modyfikacja wartości zmiennej `sumaDzielników`. Wartość `sumaDzielników` zostaje zwiększona o wykryty dzielnik.
>
> Ostatnim krokiem jest sprawdzenie, czy `sumaDzielników` jest równa sprawdzanej liczbie. Jeśli tak, oznacza to, że liczba ta jest liczbą doskonałą.

Liczby bliźniacze

Inną interesującą grupą są liczby bliźniacze. Są to pary liczb pierwszych, których różnica jest równa 2. Przykładami takich par będą:

- 3 i 5,
- 5 i 7,
- 11 i 13,
- 17 i 19,
- 857 i 859.

Problem 3

Napiszmy algorytm w postaci pseudokodu, który sprawdzi, czy dane dwie liczby `liczba1` oraz `liczba2` są liczbami bliźniaczymi.

Specyfikacja:

Dane:

- `liczba1` – liczba do sprawdzenia; liczba naturalna
- `liczba2` – liczba do sprawdzenia; liczba naturalna

Wynik:

Program wypisuje komunikat `Liczby są parą liczb bliźniaczych` lub `Liczby nie są parą liczb bliźniaczych`.

Oto zapisany za pomocą pseudokodu algorytm sprawdzający, czy dwie zadane liczby są liczbami bliźniaczymi:

```
funkcja czyPierwsza(liczba):
    jeżeli liczba <= 1:
        zwróć fałsz
    w przeciwnym razie:
        i ← 2
        dopóki i * i <= liczba:
            jeżeli liczba mod i = 0:
                zwróć fałsz
            i ← i + 1
    zwróć prawda

funkcja wartośćBezwzględna(liczba):
    jeżeli liczba < 0:
        zwróć -liczba
    w przeciwnym razie:
        zwróć liczba

```

```
liczba1 ← wprowadź liczbę
liczba2 ← wprowadź liczbę

jeżeli wartośćBezwzględna(liczba1 - liczba2) = 2:
    jeżeli czyPierwsza(liczba1) = prawda:
        jeżeli czyPierwsza(liczba2) = prawda:
            wypisz "Liczby są parą liczb bliźniaczych"
        w przeciwnym razie:
            wypisz "Liczby nie są parą liczb bliźniaczych"
    w przeciwnym razie:
        wypisz "Liczby nie są parą liczb bliźniaczych"
w przeciwnym razie:
    wypisz "Liczby nie są parą liczb bliźniaczych"
```

Funkcja `czyPierwsza()` sprawdza, czy dana liczba jest liczbą pierwszą. Jeśli tak, zwraca wartość `prawda`. Wykorzystuje ona algorytm przedstawiony w sekcji powyżej.

Funkcja `wartośćBezwzględna()` zwraca wartość bezwzględną z danej liczby.

Na początku sprawdzamy, czy różnica między liczbami wynosi 2. Jeżeli tak, to sprawdzamy, czy liczby są liczbami pierwszymi. W takim wypadku wiemy, że podane liczby są liczbami bliźniaczymi.

Liczby zaprzyjaźnione

Aby dwie liczby naturalne można było uznać za parę liczb zaprzyjaźnionych, suma dzielników właściwych pierwszej z nich musi być równa wartości drugiej liczby, natomiast suma dzielników właściwych drugiej liczby musi być równa wartości pierwszej liczby. Przykładem pary spełniającej ten warunek są liczby 220 i 284. Sprawdźmy:

- dzielniki właściwe liczby 220:
    - 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110 – ich suma wynosi 284;
- dzielniki właściwe liczby 284:
    - 1, 2, 4, 71, 142 – ich suma wynosi 220.

Słownik

dzielnik

liczba, która dzieli bez reszty daną liczbę całkowitą
dzielniki właściwe

dzielniki danej liczby mniejsze od niej samej

Schemat interaktywny

Polecenie 1

Korzystając z języka programowania lub schematu interaktywnego, zastosuj algorytm sprawdzający, czy dwie podane liczby są parą liczb zaprzyjaźnionych.

Specyfikacja:

Dane:

- `liczba1` – liczba do sprawdzenia; liczba naturalna
- `liczba2` – liczba do sprawdzenia; liczba naturalna

Wynik:

Program wypisuje komunikat `Liczby są parą liczb zaprzyjaźnionych` lub `Liczby nie są parą liczb zaprzyjaźnionych`.

```
1
```

```
1
```

Polecenie 2

Przygotuj notatkę ze swoimi spostrzeżeniami dotyczącymi stworzonego algorytmu.

Polecenie 3

Podaj przykłady innych algorytmów liczbowych.

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Ćwiczenie 2

Ćwiczenie 3

Ćwiczenie 4

Ćwiczenie 5

Ćwiczenie 6

Zapoznaj się z kodem źródłowym i wykonaj polecenie.

```
liczba ← wprowadź liczbę naturalną
s ← 0

dla i = 1, 2, 3, ..., liczba - 1 wykonuj:
    jeżeli liczba mod i = 0:
        s ← s + i

jeżeli s = liczba:
    wypisz "TAK"
w przeciwnym razie:
    wypisz "NIE"
```

Ćwiczenie 7

Zapoznaj się z kodem źródłowym i wykonaj polecenie.

```
liczba ← wprowadź liczbę naturalną
k ← prawda

jeżeli liczba < 2:
    k ← fałsz
w przeciwnym razie:
    i ← 2
    dopóki i * i <= liczba:
        jeżeli liczba mod i = 0:
            k ← fałsz
            wyjdź z pętli
        i ← i + 1

jeżeli k = prawda:
    wypisz "TAK"
w przeciwnym razie:
    wypisz "NIE"
```

Ćwiczenie 8

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Algorytmy liczbowe

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

1) planuje kolejne kroki rozwiązywania problemu, z uwzględnieniem podstawowych etapów myślenia komputacyjnego (określenie problemu, definicja modeli i pojęć, znalezienie rozwiązania, zaprogramowanie i testowanie rozwiązania).

2) stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy:

a) na liczbach: badania pierwszości liczby, zamiany reprezentacji liczb między pozycyjnymi systemami liczbowymi, działań na ułamkach z wykorzystaniem NWD i NWW,

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) w zależności od problemu rozwiązuje go, stosując metodę wstępującą lub zstępującą;

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) projektuje i tworzy rozbudowane programy w procesie rozwiązywania problemów, wykorzystuje w programach dobrane do algorytmów struktury danych, w tym struktury dynamiczne i korzysta z dostępnych bibliotek dla tych struktur;

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Wyjaśnisz, jakie właściwości musi posiadać dana liczba, aby mogła zostać nazwana liczbą doskonałą, zaprzyjaźnioną, bliźniaczą lub pierwszą.
- Przeanalizujesz algorytmy, które pozwalają sprawdzić, czy dana liczba spełnia założone kryteria.
- Rozwiążesz zadania sprawdzające wiedzę na temat algorytmów liczbowych.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Algorytmy liczbowe". Uczniowie mają zapoznać się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla temat i cele zajęć. Prosi uczniów, by na podstawie wiadomości zdobytych przed lekcją zaproponowali kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Uczniowie tworzą pytania dotyczące tematu zajęć, na które odpowiedzą w trakcie lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel ocenia, na podstawie informacji na platformie, stan przygotowania uczniów do zajęć. Jeżeli jest ono niewystarczające prosi o ciche zapoznanie się z treścią w sekcji „Przeczytaj".
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Schemat interaktywny". W przedstawionej aplikacji uczniowie przygotowują algorytm sprawdzający, czy para liczb 220 i 284 jest parą liczb zaprzyjaźnionych.
3. **Ćwiczenie umiejętności.** Uczniowie realizują indywidualnie ćwiczenia nr 1-8, po ich wykonaniu porównują otrzymane wyniki z inną osobą.

Faza podsumowująca:

1. Nauczyciel ponownie wyświetla na tablicy temat i cele lekcji zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji następuje omówienie ewentualnych problemów z rozwiązaniem ćwiczeń z sekcji „Sprawdź się".

Praca domowa:

1. Zapisz w wybranym języku programowania program przygotowany w sekcji „Schemat interaktywny".

Wskazówki metodyczne:

- Treści w sekcji „Przeczytaj" można wykorzystać jako podsumowanie i utrwalenie wiedzy uczniów.