Pętle iteracyjne

Źródło: Tine Ivanič, domena publiczna.

> W tym e-materiale powtarzamy również wiadomości ze szkoły podstawowej.

Pętle są jednymi z podstawowych konstrukcji stosowanych podczas pisania programów. Umożliwiają one powtarzanie określonych operacji dopóty, dopóki spełniony jest warunek zdefiniowany przez autora programu.

W tym e-materiale poznamy pętlę `for`.

Implementacje omawianego zagadnienia w poszczególnych językach programowania znajdziesz w e-materiałach:

- Pętle iteracyjne w języku C++,
- Pętle iteracyjne w języku Java,
- Pętle iteracyjne w języku Python.

Więcej zadań? Sięgnij do Pętle iteracyjne – zadania maturalne.

> **Twoje cele**
>
> - Poznasz budowę oraz działanie pętli `for`.
> - Przeanalizujesz, jak pętla `for` może zostać użyta przy implementacji algorytmu.
> - Wykorzystując zdobytą wiedzę, stworzysz własny algorytm.

Przeczytaj

Pętla jest konstrukcją programistyczną, która umożliwia cykliczne wykonywanie ciągu instrukcji określoną liczbę razy, do momentu zajścia pewnych warunków, dla każdego elementu kolekcji lub w nieskończoność. Pętla `for`, którą się zajmujemy, należy do grupy pętli iteratywnych. Istnieją jeszcze inne rodzaje pętli, m.in. warunkowe, jednak zostaną one omówione w innych rozdziałach.

Oznacza to, że pętla wykonuje instrukcje w niej zawarte ustaloną liczbę razy.

Przykład w pseudokodzie

Pętla `for` zapisana w pseudokodzie:

```
dla i = 1, 2, ..., 100 wykonuj
    wypisz i
```

W przedstawionym przykładzie pętla `for` wykona się `100` razy, a przy każdej iteracji wypisze wartość zmiennej `i`. Kolejne wartości będą wyglądać następująco:
`1 2 3 4 5 6 7 8 ... 100`

Budowa pętli `for`

Zazwyczaj deklaracja pętli `for` składa się z trzech elementów. Pierwszym z nich jest zmienna sterująca, używana w pętli `for` jako licznik wykonanych do tej pory iteracji. Nazwa tej zmiennej jest dowolna, natomiast powszechnie przyjęta zasada sugeruje użycie litery `i`. Dla kolejnych, zagnieżdżonych pętli używamy następnych liter alfabetu (`i`, `j`, `k`...).

Zadaniem drugiej części pętli `for` jest sprawdzenie, czy warunek wykonania pętli nadal jest spełniony. Jest to wyrażenie logiczne, które kontroluje, czy pętla wykonała się już określoną liczbę razy. Wyrażenie to jest sprawdzane przed każdą iteracją – jeżeli zwróci wartość logiczną `false`, działanie pętli zostaje zakończone, a program wykonuje dalsze instrukcje.

Ostatnim elementem pętli jest modyfikacja wartości zmiennej sterującej: możemy w ten sposób określić, o ile wartość zmiennej sterującej ma się zmienić po każdej iteracji. Zazwyczaj wartość ta wynosi jeden. Gdy ustalimy, że zmienna ma być zwiększana o jeden z każdą iteracją, wówczas mówimy o inkrementacji. W odwrotnym przypadku, gdy wartość zmiennej sterującej ulega zmniejszeniu o jeden, mamy do czynienia z dekrementacją.

> **Ważne!**

> Należy zawsze upewnić się, że warunek pozwalający programowi na wyjście z pętli jest możliwy do osiągnięcia. W przeciwnym wypadku pętla staje się pętlą nieskończoną i program – po natrafieniu na taką pętlę – może ulec zawieszeniu czy przerwaniu.
> **Ważne!**
>
> Zazwyczaj pętla `for` z każdą iteracją dodaje jakąś wartość do zmiennej sterującej, aż ta osiągnie wartość wymaganą do opuszczenia pętli. Ponieważ jednak zmienna sterująca może zmieniać wartość o dowolną liczbę, możliwa jest sytuacja, w której wartość zmiennej jest **zmniejszana** z każdą iteracją – nasza pętla skończy działanie, gdy wartość zmiennej sterującej spadnie poniżej pewnej określonej liczby.

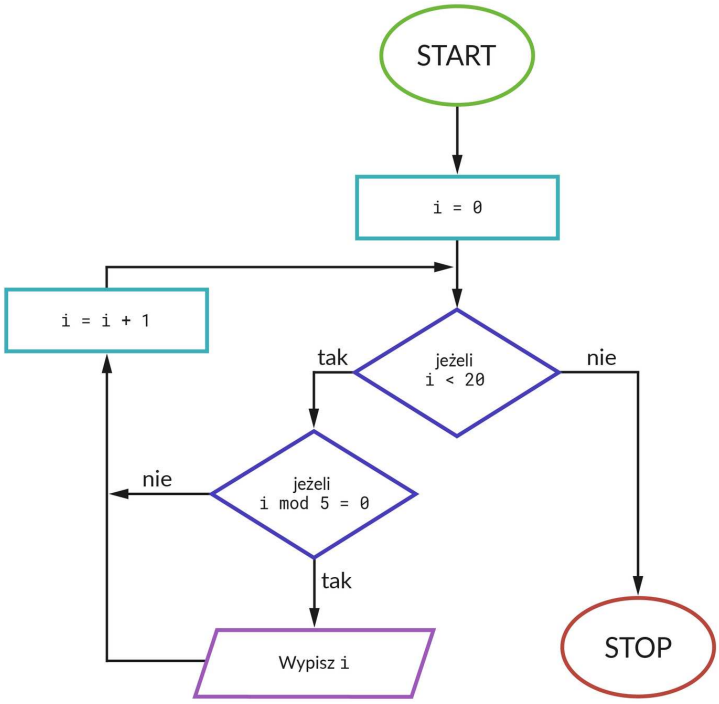
Pętla `for` w praktyce

Aby lepiej zrozumieć działanie pętli `for`, spróbujmy użyć jej w praktyce. Wyobraźmy sobie, że mamy zbiór `20` liczb całkowitych od `0` do `19` i musimy wypisać tylko te, których reszta z dzielenia przez `5` wynosi `0`. Zadanie to jest możliwe do rozwiązania w pamięci – zbiór rozwiązań będzie miał tylko cztery elementy. Kiedy jednak zwiększymy liczbę elementów w zbiorze do `2000`, ręczne szukanie wartości może okazać się bardzo czasochłonne. Dlatego też w celu rozwiązania przedstawionego problemu napiszemy algorytm w postaci pseudokodu:

```
dla i = 0, 1, ..., 19 wykonuj
    jeżeli i mod 5 = 0 to
        wypisz i
```

Program sprawdza kolejne liczby z zakresu od `0` do `19` i bada, czy reszta z dzielenia przez `5` (operator `mod`) każdej z liczb z tego zakresu jest równa `0`. Jeżeli tak jest, liczba ta zostaje wypisana.

Rozwiązanie to można również przedstawić w postaci schematu blokowego. Dzięki temu będzie wiadomo, co dzieje się w naszej pętli `for`.



Spróbuj samodzielnie przeanalizować, co dzieje się na każdym etapie przetwarzania tego schematu blokowego. Na następnej stronie znajdziesz szczegółową analizę bardzo podobnego schematu blokowego pętli `for`.

Słownik

iteracja

pojedyncze wykonanie poleceń zawartych w pętli
inkrementacja

zwiększenie wartości zmiennej o jeden
dekrementacja

zmniejszenie wartości zmiennej o jeden
zagnieżdżone pętle

wywołanie pętli wewnątrz innej pętli
nieskończona pętla

pętla, która nigdy nie zostanie zakończona, ponieważ warunek wyjścia z pętli nigdy nie zostaje spełniony

Animacja

Polecenie 1

Przeanalizuj animację, w której przedstawiono kilka przykładów użycia pętli licznikowej.

Film dostępny pod adresem /preview/resource/RKguS8Fa5TZKK

Film nawiązujący do treści pętli iteracyjnych.

Polecenie 2

Opracuj notatkę podsumowującą najważniejsze informacje przedstawione w animacji.

Polecenie 3

Zapisz propozycję wykorzystania pętli licznikowej przy rozwiązywaniu problemu z lekcji matematyki lub fizyki.

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Napisz w pseudokodzie algorytm, który wypisze sumę wszystkich liczb całkowitych z zakresu ⟨`0, 100`⟩ kończących się cyfrą 3.

Ćwiczenie 2

Wskaż, jak **zazwyczaj** nazywana jest zmienna sterująca w pętli `for`.

- `s`
- `i`
- `a`
- `t`

Ćwiczenie 3

Zaznacz wartości o jakie może zmieniać się zmienna sterująca typu `int`, z każdą iteracją.

- `-1`
- dowolne całkowite
- 2
- 1

Ćwiczenie 4

Nieskończona pętla jest: [ ].

- pętlą, której liczba wykonań jest bardzo duża
- pętlą, która nie została jeszcze do końca napisana
- pętlą, w przypadku której nigdy nie jest spełniony warunek zakończenia

Ćwiczenie 5

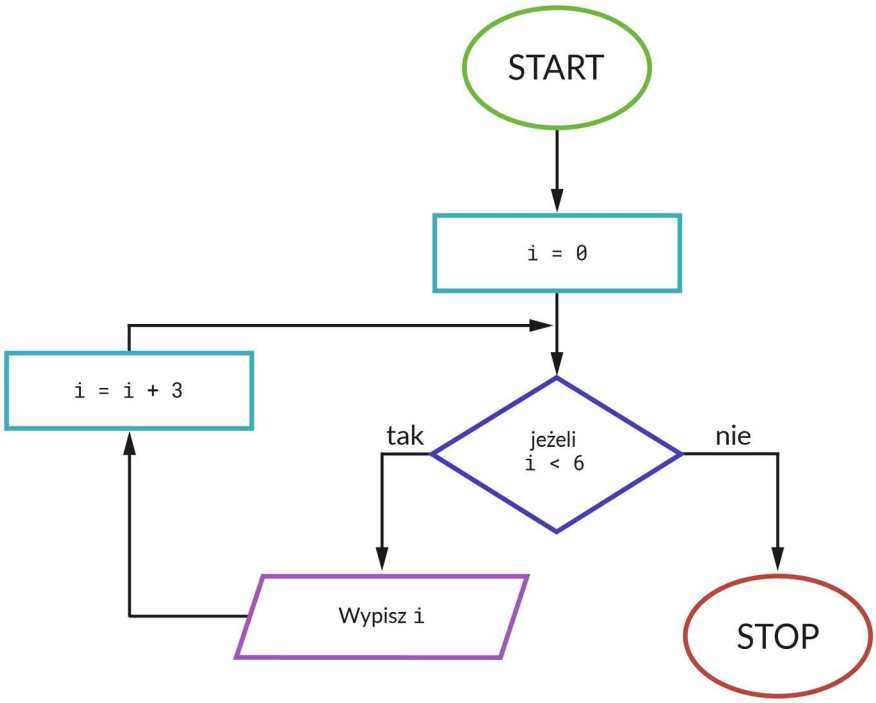
Wskaż, jakie liczby zostaną wypisane dzięki poniższemu pseudokodowi.

```
dla i = 10, 11, ..., 15 wykonuj:
    wypisz 3 * (i - 10) - 1
```

- ◯ `24, 27, 30, 33, 36`
- ◯ `-3, -1, 2, 5, 8, 14`
- ◯ `-1, 2, 5, 8, 11, 14`
- ◯ `0, 11, 12, 13, 14`
- ◯ `0, 1, 2, 3, 4`

Ćwiczenie 6

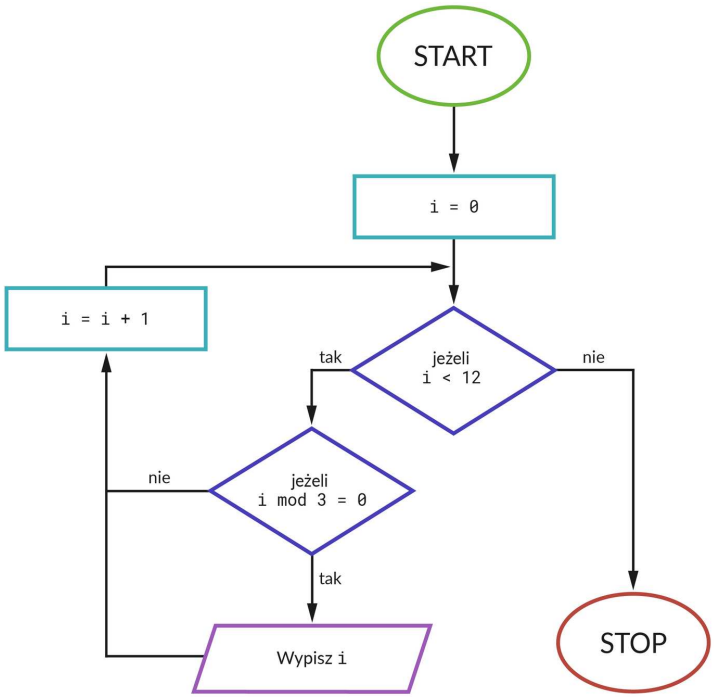
Wskaż, jakie liczby zostaną wypisane po zastosowaniu przedstawionego algorytmu.



- ◯ `3`
- ◯ `3, 6`
- ◯ `0, 3, 6`
- ◯ `0, 3`

Ćwiczenie 7

Wskaż, jakie liczby zostaną wypisane po zastosowaniu przedstawionego algorytmu.



- ◯ `0, 3, 6, 9, 12`
- ◯ `0, 3, 6, 9`
- ◯ `3, 6, 9`
- ◯ `3, 6, 9, 12`

Ćwiczenie 8

Napisz w pseudokodzie algorytm, który wypisze wszystkie liczby całkowite z zakresu $\langle 1,\ 500 \rangle$ posiadające dokładnie cztery dzielniki całkowite.

Dla nauczyciela

Autor: Anna Kwaśna
Przedmiot: Informatyka

Temat: Pętle iteracyjne

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum

Podstawa programowa:

Cele kształcenia – wymagania ogólne

> 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

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.
>
> > Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:
> >
> > > 2) stosuje zasady programowania strukturalnego i obiektowego w rozwiązywaniu problemów;

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):

- Poznasz budowę oraz działanie pętli `for`.
- Przeanalizujesz, jak pętla `for` może zostać użyta przy implementacji algorytmu.
- Wykorzystując zdobytą wiedzę, stworzysz własny algorytm.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja.

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: „Pętle iteracyjne". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Prowadzący wyświetla na tablicy interaktywnej zawartość sekcji „Wprowadzenie" i omawia cele do osiągnięcia w trakcie lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Uczniowie przystępują do cichego czytania tekstu zawartego w sekcji „Przeczytaj", jeśli nauczyciel - na podstawie raportu na platformie - uważa, że przygotowanie uczniów jest wystarczające, może pominąć tę czynność.
2. **Praca z multimedium.** Uczniowie zapoznają się indywidualnie z treścią sekcji „Animacja" i poleceniem nr 1, zapisują problemy i pytania z nią związane. Po czym następuje dyskusja, w trakcie której nauczyciel wyjaśnia niezrozumiałe kroki procedury.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują ćwiczenia nr 1-8 z sekcji „Sprawdź się". Nauczyciel sprawdza poprawność wykonanych zadań, omawiając je wraz z uczniami.

Faza podsumowująca:

1. Nauczyciel zadaje pytania podsumowujące, np.
   – co to jest iteracja?
   – co oznacza termin inkrementacja?
   – do czego służy zagnieżdżona pętla?

Praca domowa:

1. Uczniowie wykonują polecenie 2 z sekcji „Animacja”.

Wskazówki metodyczne:

- Uczniowie mogą wykorzystać treści w sekcjach: „Przeczytaj”, „Animacja”, „Sprawdź się” jako materiał do lekcji powtórkowej.