Funkcje w języku Java

- Wprowadzenie
- Przeczytaj
- Gra edukacyjna
- Sprawdź się
- Dla nauczyciela

Źródło: cagper, domena publiczna.

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

Kiedy piszemy coraz bardziej rozbudowane programy, ich kod staje się trudny do zrozumienia. Natłok instrukcji umieszczonych jedna za drugą sprawia, że całość jest mało czytelna.

Jednym ze sposobów uporządkowania kodu i uczynienia go bardziej przejrzystym jest podzielenie programu na bloki funkcjonalne, które odpowiadają za realizację określonych zadań (przykładowo, za wykonywanie sekwencji obliczeń albo prezentację ich wyników).

Takie odrębne części kodu nazywa się różnie, w zależności od tego, z jakim językiem programowania mamy do czynienia (można zetknąć się na przykład z terminami „podprogram” albo „procedura”). W językach C/C++ oraz Java bloki logiczne noszą nazwę funkcji – i nimi właśnie zajmiemy się w tym e-materiale.

Więcej informacji o funkcjach znajdziesz w e-materiale Funkcje. Chcesz wiedzieć, jak wygląda omawiane zagadnienie w innych językach programowania? Zapoznaj się z e-materiałami:

- Funkcje w języku C++,
- Funkcje w języku Python.

Twoje cele

- Dowiesz się, jak utworzyć funkcje w języku Java.
- Przeanalizujesz kilka przypadków wykorzystania funkcji.
- Napiszesz własne programy, w których zastosujesz funkcje.

Przeczytaj

Czym jest funkcja?

Wspomnieliśmy już, że funkcja jest wydzieloną częścią kodu, odpowiedzialną za wykonanie określonych zadań. Jeżeli piszemy rozbudowany program, stanie się on bardziej przejrzysty po rozbiciu całości na funkcje.

Co istotne, funkcje ułatwiają użycie tej samej sekwencji poleceń w różnych częściach programu. Zamiast powielać za każdym razem identyczne wycinki kodu, wywołujemy zawierającą je funkcję.

Gdybyśmy wreszcie chcieli edytować powtarzające się fragmenty kodu, dzięki zastosowaniu funkcji robimy to tylko raz. Poprawiamy wówczas samą funkcję, nie musząc zmieniać programu w wielu miejscach.

Pierwsza funkcja w języku Java

Oto deklaracja przykładowej funkcji w Javie:

```
public static int dodajLiczby(int argumentFunkcji, int drugiArgum
    int wynikDoZwrocenia = argumentFunkcji + drugiArgumentFunkcji
    return wynikDoZwrocenia;

    System.out.println("Ta linijka nie zostanie wykonana");
}
```

Słowa kluczowe `public` oraz `static` omówimy w innym e-materiale. Na razie musisz pamiętać, aby zawsze używać ich podczas deklarowania funkcji.

Przejdziemy do określenia typu danych zwracanego przez funkcję. Przede wszystkim trzeba podkreślić, że każda zmienna zdefiniowana wewnątrz funkcji jest zmienną lokalną. Oznacza to, że nie można odwołać się do niej spoza funkcji, a sama zmienna jest usuwana, gdy tylko funkcja zakończy działanie.

Co jednak zrobić w sytuacji, gdy funkcja ma wykonać jakieś obliczenia? Zapewne chcielibyśmy poznać ich wyniki, więc zależy nam na tym, aby nie zniknęły one z pamięci.

Otóż można sprawić, że funkcja przed zakończeniem działania zwróci jakąś wartość – na przykład przekaże programowi głównemu rezultat wykonanych operacji

matematycznych. Musimy jednak poinformować kompilator, jakich danych powinien się spodziewać.

Wróćmy zatem do definicji funkcji. Tuż po słowach `public` i `static` musimy podać typ zwracanych wartości. Rezultatem działania przykładowej funkcji będzie liczba całkowita (`int`).

Jeżeli nie chcemy, aby funkcja zwracała jakąkolwiek wartość, jako typ danych wpisujemy `void` (czyli „pusty"). Robimy tak choćby w przypadku definiowania funkcji, której zadaniem jest wyświetlanie komunikatów; nie zwraca ona nic programowi głównemu.

> **Ważne!**
>
> Funkcja, dla której określimy typ zwracanych danych, musi takie właśnie dane zwrócić. W przeciwnym przypadku pojawią się błędy kompilacji.

Tuż za typem zwracanych wartości podajemy nazwę funkcji. W naszym przypadku brzmi ona: `dodajLiczby`. Będziemy się nią posługiwali podczas wywoływania funkcji.

Gdy funkcja nie będzie przyjmować żadnych parametrów, za nazwą stawiamy parę okrągłych nawiasów: `()` – i deklaracja jest gotowa. Kiedy jednak chcemy przekazać funkcji jakieś parametry, musimy je opisać.

Argument jest wartością przekazaną do parametru funkcji. Parametry deklarujemy wewnątrz nawiasów okrągłych, wskazując ich typy i nazwy. Nie ma ograniczeń co do liczby przekazanych parametrów; musimy tylko oddzielić je przecinkami.
W zaprezentowanym przykładzie mamy dwa parametry typu całkowitego (`int`) o nazwach `argumentFunkcji` oraz `drugiArgumentFunkcji`.

> **Ważne!**
>
> Kiedy jako argument przekazujemy zmienną, funkcja tworzy lokalną kopię jej wartości. Oznacza to, że jeżeli wewnątrz funkcji zmienna zostanie zmodyfikowana, nie będzie miało to wpływu na jej oryginalną wartość.

Po deklaracji zapisujemy ciało funkcji, czyli instrukcje, które zostaną wykonane po jej wywołaniu. Polecenia te umieszczamy w parze nawiasów klamrowych. W przykładowej funkcji definiujemy zmienną lokalną `wynikDoZwrocenia` i przypisujemy jej sumę podanych argumentów.

Dalej pojawia słowo kluczowe `return`. Jest to polecenie zatrzymania działania funkcji i zwrócenia programowi głównemu wskazanej wartości (w tym przypadku jest nią `wynikDoZwrocenia`). Podkreślmy to wyraźnie: polecenie `return` oznacza natychmiastowe opuszczenie funkcji. Ostatnia komenda w przykładzie – `System.out.println...` – jest więc przewrotna, bo nigdy nie zostanie wykonana, choć z punktu widzenia składni języka Java najzupełniej poprawna.

> **Ciekawostka**
>
> Jeżeli chcemy przerwać działanie funkcji, która nie zwraca żadnej wartości (czyli przed jej nazwą pojawiło się słowo `void`), możemy zrobić to w dowolnym momencie, wydając polecenie `return` (i pamiętając aby nie podawać żadnej wartości do zwrócenia).

Wywołanie funkcji

Gdy chcemy użyć funkcji w którymkolwiek miejscu programu, po prostu podajemy jej nazwę (oraz argumenty, o ile są one niezbędne):

```
int wynikDzialaniaFunkcji = dodajLiczby(2, 4);
```

Zdefiniowana wcześniej funkcja `dodajLiczby` zwraca wartość typu `int`. Przypisaliśmy ją zmiennej `wynikDzialaniaFunkcji`. Jeżeli funkcja nie oczekuje argumentów, za jej nazwą umieszczamy pustą parę okrągłych nawiasów.

> **Ważne!**
>
> Gdy funkcja wymaga podania argumentów, ale ich nie otrzyma (bądź przekażemy jej złą liczbę argumentów), mogą pojawić się błędy kompilacji.

Kolejną część lekcji poświęcimy przykładom wykorzystania funkcji.

Słownik

> **return**
>
> polecenie przerywające wykonywanie funkcji i zwracające wskazaną za nim wartość programowi nadrzędnemu
> **parametr funkcji**
>
> element składni języka, który jest definiowany przy deklaracji metody lub funkcji; deklarujemy go w nagłówku podprogramu; umożliwia komunikację pomiędzy podprogramem (funkcją/metodą), a programem go wywołującym
> **argument funkcji**
>
> element składni w określonym języku programowania, który w wyniku wywołania podprogramu zostaje utożsamiony (skojarzony) z określonym parametrem podprogramu
> **prototyp funkcji**
>
> zwany również deklaracją; funkcja która nie posiada ciała, a jedynie informuje kompilator o postaci parametrów oraz zwracanej wartości

Gra edukacyjna

Polecenie 1

Zapoznaj się z prezentacją.

1

Aby oswoić się z funkcjami, napiszemy dwa nieskomplikowane programy w języku Java. W obydwu aplikacjach wykorzystamy funkcje.

2

Najpierw napiszemy funkcję, której parametrami są dwie liczby całkowite, a wynikiem większa z tych liczb.

3

Zaczynamy od deklaracji:

```
1 public class Main {
2     public static void main(String [] args) {
3
4     }
5
6     public static int wiekszaLiczba(int a, int b) {
7         // Zwróć większą liczbe
8     }
9 }
```

Wspominaliśmy już, że słowa kluczowe `public` oraz `static` muszą pojawić się na początku

deklaracji. Ponieważ funkcja ma wskazać większą liczbę, typem zwracanych danych będzie `int`. Nazwa funkcji to `wiekszaLiczba`. Wymaga ona podania dwóch parametrów typu całkowitego: a oraz b.

4

Teraz wystarczy napisać instrukcję warunkową, dzięki której wybierzemy większą liczbę:

```
1  public class Main {
2      public static void
   main(String [] args) {
3
4      }
5
6      public static int
   wiekszaLiczba(int a, int
   b) {
7          if (a > b) {
8              return a;
9          }
10
11         return b;
12     }
13 }
```

Zmienna o większej wartości jest zwracana programowi głównemu dzięki poleceniu `return`. Ponieważ jeżeli zmienna a okaże się większa, zostanie zwrócona a funkcja zakończy działanie nie musimy używać instrukcji `else`.

5

Możemy już użyć funkcji i zobaczyć efekt działania programu:

```
1 public class Main {
2     public static void
  main(String [] args) {
```

```
3         int wynikDzialaniaFunkcji = wiekszaLiczba(5, 10);
4
5         System.out.println(wynikDzialaniaFunkcji);
6     }
7
8     public static int wiekszaLiczba(int a, int b) {
9         if (a > b) {
10            return a;
11        }
12
13        return b;
14    }
15 }
```

6

Teraz napiszemy funkcję, która podany jej parametr pomnoży przez 10, a wynik wyświetli na ekranie (nie zwracając żadnej wartości).

7

Zacznijmy od deklaracji:

```
1 public class Main {
2     public static void main(String [] args) {
3         int wynikDzialaniaFunkcji = wiekszaLiczba(5, 10);
4
5         System.out.println(wynikDzialaniaFunkcji);
6     }
```

```
7
8       public static void zwiekszWypisz(int a) {
9           // Pomnóż liczbę przez 10 i wypisz
10      }
11  }
```

8

Funkcja nie będzie zwracać jakiejkolwiek wartości, a zatem używamy słowa kluczowego `void`. Następnie nadajemy funkcji nazwę oraz definiujemy jeden parametr: liczbę całkowitą `a`.

9

W ciele funkcji wpiszemy dwie instrukcje: mnożenia parametru przez 10 oraz wyświetlenia rezultatu:

```
1  public class Main {
2      public static void main(String [] args) {
3          int wynikDzialaniaFunkcji = wiekszaLiczba(5, 10);
4
5          System.out.println(wynikDzialaniaFunkcji);
6      }
7
8      public static void zwiekszWypisz(int a) {
9          a = a * 10;
10         System.out.println(a);
11     }
12 }
```

10

Wywołujemy funkcję `zwiekszWypisz` i sprawdzamy jej działanie:

```
1  public class Main {
2      public static void main(String [] args) {
3          int wynikDzialaniaFunkcji = wiekszaLiczba(5, 10);
4
5          System.out.println(wynikDzialaniaFunkcji);
6          zwiekszWypisz(30);
7      }
8
9      public static void zwiekszWypisz(int a) {
10         a = a * 10;
11         System.out.println(a);
12     }
13 }
```

Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Problem 1

Specyfikacja problemu:

Napisz funkcję, której parametrami są dwie liczby całkowite, a wynikiem iloczyn tych liczb.

Lista kroków:

```
public class Main {
    public static void main(String [] args) {
        // zdefiniuj zmienne i wywołaj funkcję
    }

    // tutaj zdefiniuj funkcję
}
```

```
```

Polecenie 2

Sprawdź swoją wiedzę, biorąc udział w poniższej grze.

Test

Sprawdź swoją wiedzę o funkcjach w języku Java, biorąc udział w grze

Poziom trudności:

łatwy

Limit czasu:

7 min

Twój ostatni wynik:

-

Uruchom

Polecenie 3

Zaproponuj trzy pytania, które mogłyby pojawić się w grze edukacyjnej.

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

W programowaniu funkcje często są wykorzystywane do przeliczania jednostek. Aby przekonwertować wartość w stopniach Kelwina na stopnie Celsjusza, od temperatury w Kelwinach odejmujemy 273,15 - otrzymany wynik to temperatura w stopniach Celsjusza. Ponadto, woda przy ciśnieniu jednej atmosfery zamarza w temperaturze 0 stopni Celsjusza.

Korzystając z podanych informacji, napisz dwie funkcje. Pierwsza z nich będzie konwertowała wartość temperatury podaną w stopniach Kelwina na stopnie Celsjusza, zaś druga będzie określała, czy w podanej temperaturze woda zamarza. Do obliczeń użyj typów `double`.

Twoje zadania

1. Program wypisuje pojedyncze słowo "tak" jeżeli woda zamarza w podanej temperaturze wyrażonej w stopniach Kelwina, lub "nie" jeżeli woda nie zamarza w tej temperaturze.

Ćwiczenie 2

Napisz program, który obliczy średnią arytmetyczną tablicy składającej się z 10 liczb. Funkcja powinna zwracać typ `double`.

Twoje zadania

1. Program wypisuje średnią arytmetyczną liczb podanych w tablicy.

Ćwiczenie 3

Na każdy przebyty kilometr samolot pasażerski zużywa około 12 litrów paliwa. Napisz program, który dla podanej ilości kilometrów (jako liczby całkowitej) oraz stanu zbiornika wyrażonego w litrach poda, czy samolot będzie w stanie przebyć ten dystans.

Twoje zadania

1. Funkcja `paliwo` przyjmuje dwie wartości typu `double` - ilość kilometrów, oraz stan zbiornika w litrach. Funkcja powinna zwracać `true` jeżeli paliwa wystarczy na przebycie tego dystansu, zaś w przeciwnym wypadku zwraca `false`.

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Funkcje w języku Java

Grupa docelowa:

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

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 podstawowy. Uczeń:
> >
> > > 1) projektuje i programuje rozwiązania problemów z różnych dziedzin, stosuje przy tym: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych; w szczególności programuje algorytmy z punktu I.2);

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):

- Dowiesz się, jak utworzyć funkcje w języku Java.
- Przeanalizujesz kilka przypadków wykorzystania funkcji.
- Napiszesz własne programy, w których zastosujesz funkcje.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

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

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;
- oprogramowanie dla języka Java SE 8 (lub nowszej wersji), w tym Eclipse 4.4 (lub nowszej wersji).

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Funkcje w języku Java". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj" w kontekście programowania.

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.** Nauczyciel zadaje uczniom pytanie dotyczące ich aktualnego stanu wiedzy w obszarze poruszanego tematu, opartego o programowanie.

Faza realizacyjna:

1. **Praca z tekstem.** Jeżeli przygotowanie uczniów do lekcji jest niewystarczające, nauczyciel prosi o indywidualne zapoznanie się z treścią zawartą w sekcji „Przeczytaj". Każdy uczestnik zajęć podczas cichego czytania wynotowuje najważniejsze kwestie poruszane w tekście.

2. **Praca z multimedium.** Nauczyciel czyta polecenie nr 1 „Zapoznaj się z prezentacją, a następnie napisz funkcję, która zwróci wynik mnożenia dwóch argumentów." w sekcji „Gra edukacyjna". Prosi uczniów, aby w parach przeanalizowali rozwiązanie problemu. Uczniowie odtwarzają kolejne kroki na swoich komputerach.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują ćwiczenia nr 1-3 z sekcji „Sprawdź się". Nauczyciel sprawdza poprawność wykonanych zadań, omawiając je wraz z uczniami.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności, omawia ewentualne problemy podczas rozwiązania ćwiczeń z programowania w języku Java.

Praca domowa:

1. Uczniowie opracowują FAQ (minimum 3 pytania i odpowiedzi) do tematu lekcji („Funkcje w języku Java").

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Java SE 8 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Eclipse 4.4 (lub nowszej wersji).

Wskazówki metodyczne:

- Uczniowie mogą wykorzystać multimedia z sekcji: „Przeczytaj", „Gra edukacyjna", „Sprawdź się" do przygotowania się do lekcji powtórkowej.