Obliczanie przybliżonej wielkości pola obszarów zamkniętych w języku Java

Źródło: Markus Spiske, domena publiczna.

Wiem już jak obliczyć przybliżoną wielkość pola obszarów zamkniętych. W tym e-materiale zaimplementujemy w języku Java metody prostokątów i trapezów obliczające pole obszaru ograniczanego wykresem funkcji.

Implementacje w pozostałych językach programowania znajdziesz w e-materiałach:

- Obliczanie przybliżonej wielkości pola obszarów zamkniętych w języku C++,
- Obliczanie przybliżonej wielkości pola obszarów zamkniętych w języku Python.

Więcej zadań? Przejdź do e-materiału Obliczanie przybliżonej wielkości pola obszarów zamkniętych – zadania maturalne.

> **Twoje cele**
>
> - Użyjesz w praktyce wiedzy na temat wyznaczania przybliżonej wielkości pól obszarów zamkniętych metod prostokątów i trapezów.
> - Przeanalizujesz zaimplementowane w języku Java algorytmy, które pozwalają na obliczenie pola obszaru ograniczonego wykresem funkcji oraz osią x.
> - Rozwiążesz kilka zadań dotyczących zagadnienia obliczania przybliżonej wielkości pola obszarów zamkniętych.

Przeczytaj

W tej sekcji zapoznamy się ze sposobem obliczania przybliżonej wielkości pola obszarów zamkniętych metodą trapezów w języku Java.

Metoda trapezów

Specyfikacja problemu:

Dane:

- `xPoczatek` – liczba rzeczywista; początek przedziału
- `xKoniec` – liczba rzeczywista; koniec przedziału
- `liczbaTrapezow` – liczba naturalna dodatnia; liczba podprzedziałów
- `f(x)` – funkcja rzeczywista zmiennej rzeczywistej

Wynik:

- `wynik` – liczba rzeczywista; przybliżona wartość pola pod wykresem funkcji na zadanym przedziale

Zacznijmy omówienie zagadnienia dotyczącego metod obliczania przybliżonego pola pod krzywą od zdefiniowania funkcji $f(x) = -x^2 + 1$. Będzie zwracała liczbę zmiennoprzecinkową i przyjmowała jeden argument, liczbę zmiennoprzecinkową `x`. Za jej pomocą obliczymy wartość funkcji w punkcie `x`.

```
1  public class Zadanie {
2
3      static double f(double x) {
4          return -x * x + 1;
5      }
6
7      public static void main(String[] args) {
8
9      }
10
11 }
```

Zdefiniujemy funkcję `f(x)` oraz zadeklarujemy zmienne `liczbaTrapezow`, `xPoczatek`, `xKoniec`.

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

    }

}
```

Wyznaczamy wysokość każdego trapezu. Tworzymy również zmienną pomocniczą `wynik`. Do niej przekażemy otrzymaną wartość.

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

        double h = (xKoniec - xPoczatek) / liczbaTrapezow;
        double wynik = 0;

    }

}
```

Zapiszmy pętlę, która zsumuje pola trapezów i otrzymaną wartość przypisze do zmiennej `wynik`. Program zakończy działanie pętli po obliczeniu pól wszystkich trapezów. By wyznaczyć pole trapezu, potrzebujemy długości jego wysokości, którą obliczyliśmy

i przypisaliśmy do zmiennej . Brakuje jeszcze długości podstaw – wyznaczymy je, obliczając wartość funkcji w punktach `xi` oraz `xi1`.

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

        double h = (xKoniec - xPoczatek) / liczbaTrapezow;
        double wynik = 0;

        for(int i = 0; i < liczbaTrapezow; i++) {
            double xi = xPoczatek + (i / liczbaTrapezow) * (xKoni
            double xi1 = xPoczatek + ((i + 1) / liczbaTrapezow) *

    }

}
```

Stwórzmy zmienne pomocnicze `a`, do której zapiszemy wartość funkcji `f(xi)`, oraz `b`, do której zapiszemy wartość funkcji w drugim punkcie, czyli `f(xi1)`.

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

        double h = (xKoniec - xPoczatek) / liczbaTrapezow;
```

```
        double wynik = 0;

        for(int i = 0; i < liczbaTrapezow; i++) {
            double xi = xPoczatek + (i / liczbaTrapezow) * (xKoni
            double xi1 = xPoczatek + ((i + 1) / liczbaTrapezow) *

            double a = f(xi);
            double b = f(xi1);

    }

}
```

Do zmiennej `wynik` dodajemy wartość obliczonego pola trapezu.

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

        double h = (xKoniec - xPoczatek) / liczbaTrapezow;
        double wynik = 0;

        for(int i = 0; i < liczbaTrapezow; i++) {
            double xi = xPoczatek + (i / liczbaTrapezow) * (xKoni
            double xi1 = xPoczatek + ((i + 1) / liczbaTrapezow) *

            double a = f(xi);
            double b = f(xi1);

            wynik = wynik + ((a + b) * h) / 2;

    }

}
```

Wypisujemy wynik.

Cały kod programu wygląda następująco:

```
public class Zadanie {

    static double f(double x) {
        return -x * x + 1;
    }

    public static void main(String[] args) {
        double liczbaTrapezow = 3;
        double xPoczatek = -1;
        double xKoniec = 0.5;

        double dx = (xKoniec - xPoczatek) / liczbaTrapezow;
        double wynik = 0;

        for(int i = 0; i < liczbaTrapezow; i++) {
            double xi = xPoczatek + (i / liczbaTrapezow) * (xKoni
            double xi1 = xPoczatek + ((i + 1) / liczbaTrapezow) *

            double a = f(xi);
            double b = f(xi1);
            double h = dx;

            wynik = wynik + ((a + b) * h) / 2;

        }

        System.out.println(wynik);

    }

}
```

Wynik działania programu:

```
1.0625
```

Problem 1

Przetestuj działanie programu dla następujących danych:

```
xPoczatek = -1
xKoniec = 0.5
liczbaTrapezow = 1000
f(x) = -x * x + 1
```

Specyfikacja problemu:

Dane:

- `xPoczatek` – liczba rzeczywista; początek przedziału
- `xKoniec` – liczba rzeczywista; koniec przedziału
- `liczbaTrapezow` – liczba naturalna dodatnia; liczba podziałów
- `f(x)` – funkcja rzeczywista zmiennej rzeczywistej

Wynik:

Program wypisuje przybliżoną wartość pola ograniczonego wykresem funkcji `f(x)` oraz prostymi `xPoczatek` i `xKoniec` obliczoną za pomocą metody trapezów w zadanym przedziale z dokładnością do dwóch miejsc po przecinku.

Słownik

metoda prostokątów

metoda szacowania wartości pola powierzchni pod wykresem funkcji za pomocą sumy pól prostokątów, będących przybliżeniem obszaru ograniczonego wykresem funkcji i osią OX

metoda trapezów

metoda szacowania wartości pola powierzchni pod wykresem funkcji za pomocą sumy pól trapezów, będących przybliżeniem obszaru ograniczonego wykresem funkcji i osią OX

Prezentacja multimedialna

Polecenie 1

Napisz program, który wyznaczy przybliżoną wartość pola ograniczonego wykresem funkcji $f\,(x)$ i osią OX w zadanym przedziale za pomocą metody trapezów.

Przetestuj działanie programu dla funkcji $f(x) = \mathrm{x}^2$ i przedziału $\langle -4, 4 \rangle$. Rozwiązanie powinno bazować na wykorzystaniu 200 trapezów.

Specyfikacja problemu:

Dane:

- `xPoczatek` – liczba rzeczywista; początek przedziału
- `xKoniec` – liczba rzeczywista; koniec przedziału
- `liczbaTrapezow` – liczba naturalna dodatnia; liczba podprzedziałów
- `f(x)` – funkcja rzeczywista zmiennej rzeczywistej

Wynik:

- `wynik` – liczba rzeczywista; przybliżona wartość pola pod wykresem funkcji na zadanym przedziale

Przykładowe rozwiązanie:

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

1

Implementację algorytmu obliczania przybliżonej wielkości pola obszarów zamkniętych ograniczonych osią OX oraz wykresem funkcji metodą trapezów zaczniemy od stworzenia funkcji, której pole pod wykresem będziemy szacować.

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Funkcją w naszym programie będzie $f(x) = x^2$.

```
1 public static double
  f(double x) {
2     return x * x;
3 }
```

Chcemy, żeby program umożliwiał wykorzystanie dowolnej (również bardzo dużej) liczby trapezów w obliczaniu przybliżonej wielkości obszaru ograniczonego funkcją oraz osią OX. Potrzebny nam jest odpowiedni typ danych, który będzie w stanie przechować duże liczby rzeczywiste. Wybierzemy typ `double`.

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Kolejnym krokiem będzie inicjacja zmiennych.

```
1 double xPoczatek = -4,
  xKoniec = 4, wysokość,
  wynik = 0;
2 double liczbaTrapezow =
  200;
```

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Zmienna `xPoczatek` reprezentuje dolną granicę przedziału, w naszym przypadku -4, a zmienna `xKoniec` reprezentuje górną granicę przedziału. Ponieważ wysokość trapezu będzie taka sama dla każdej figury, obliczamy ją raz i zapiszemy w zmiennej `wysokosc`. Zmienna `wynik` będzie przechowywać sumę pól trapezów, a zmienna `liczbaTrapezow` będzie przechowywać liczbę trapezów.

5

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Mając już wszystkie dane, możemy obliczyć wysokość każdego z trapezów.

```
1 double xPoczatek = -4,
  xKoniec = 4, wysokość,
  wynik = 0;
2 double liczbaTrapezow =
  200;
3 wysokosc = (xKoniec -
  xPoczatek) /
  liczbaTrapezow;
```

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Wynik działania `xKoniec - xPoczatek` daje długość przedziału, który następnie

dzielimy przez liczbę trapezów. Dzięki temu otrzymujemy wysokość każdego z trapezów. Obliczamy pole każdego z trapezów, dlatego potrzebujemy pętli, która pozwoli po nich przeiterować. Skoro wiemy, ile iteracji będzie potrzebnych (wiemy, ile jest trapezów), użyjemy pętli `for`.

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

```
1 double xPoczatek = -4,
  xKoniec = 4, wysokość,
  wynik = 0;
2 double liczbaTrapezow =
  200;
3
4 wysokosc = (xKoniec -
  xPoczatek) /
  liczbaTrapezow;
5
6 for (int i = 0; i <
  liczbaTrapezow; i++) {
7
8 }
```

Obliczanie pola każdego z trapezów rozbijemy na trzy części. Pierwszą z nich będzie znalezienie długości lewej podstawy.

8

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Długość lewej podstawy znajdujemy w sposób analogiczny do tego, jak znajdujemy długość lewego boku w metodzie prostokątów. Długość lewej podstawy określamy, sprawdzając, jaka

jest wartość funkcji dla punktu x, w którym zaczyna się dana podstawa.

```
1 double xPoczatek = -4,
  xKoniec = 4, wysokosc,
  wynik = 0;
2 double liczbaTrapezow =
  200;
3
4 wysokosc = (xKoniec -
  xPoczatek) /
  liczbaTrapezow;
5
6 for (int i = 0; i <
  liczbaTrapezow; i++) {
7     double lewyBok =
  f(xPoczatek + i *
  wysokosc);
8 }
```

Żeby znaleźć wartość x, w której rozpoczyna się lewa podstawa badanego trapezu, mnożymy zmienną sterującą przez wartość zmiennej `wysokosc`, a następnie dodajemy lewą granicę przedziału, aby przesunąć wartość we właściwe miejsce. Całą wartość przekazujemy do `f(x)`.

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

9

Znalezienie prawej podstawy przebiega podobnie. Prawa podstawa trapezu jest również lewą podstawą kolejnego trapezu. Możemy więc do linijki z poprzedniego kroku dodać po prostu jedynkę do zmiennej sterującej.

```
1 double xPoczatek = -4,
  xKoniec = 4, wysokosc,
  wynik = 0;
2 double liczbaTrapezow =
  200;
```

```
3
4 wysokosc = (xKoniec -
  xPoczatek) /
  liczbaTrapezow;
5
6 for (int i = 0; i <
  liczbaTrapezow; i++) {
7     double lewyBok =
  f(xPoczatek + i *
  wysokosc);
8     double prawyBok =
  f(xPoczatek + (i + 1) *
  wysokosc);
9 }
```

10

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P8TAt4OMo

Ostatnim krokiem jest obliczenie pola trapezu. Wykorzystamy wzór na nie: $\frac{(a+b)\cdot h}{2}$. Uzyskany wynik dodajemy do zmiennej `wynik`. Na koniec wypiszemy szacowaną wartość pola pod wykresem.

Oto kompletny kod programu:

```
1 public class Main {
2     public static void
  main (String [] args) {
3         double xPoczatek =
  -4, xKoniec = 4, wysokosc,
  wynik = 0;
4         double
  liczbaTrapezow = 200;
5
6         wysokosc =
  (xKoniec - xPoczatek) /
  liczbaTrapezow;
7
```

```
        for (int i = 0; i < liczbaTrapezow; i++) {
            double lewyBok = f(xPoczatek + i * wysokosc);
            double prawyBok = f(xPoczatek + (i + 1) * wysokosc);

            wynik += ((lewyBok + prawyBok) * wysokosc) / 2;
        }


System.out.println("Szacowane pole pod wykresem: " + wynik);
    }

    public static double f(double x) {
        return x * x;
    }
}
```

Polecenie 2

Napisz program, który wyznaczy przybliżoną wartość pola ograniczonego wykresem funkcji $f\,(x)$ i osią OX w zadanym przedziale za pomocą metody prostokątów.

Przetestuj działanie programu dla funkcji $f(x) = \mathrm{x}^2$ i przedziału $\langle -4, 4 \rangle$. Rozwiązanie powinno bazować na wykorzystaniu 200 prostokątów.

Specyfikacja problemu:

Dane:

- `xPoczatek` – liczba rzeczywista; początek przedziału
- `xKoniec` – liczba rzeczywista; koniec przedziału
- `liczbaProstokatow` – liczba naturalna dodatnia; liczba podprzedziałów
- `f(x)` – funkcja rzeczywista zmiennej rzeczywistej

Wynik:

- `wynik` – liczba rzeczywista; przybliżona wartość pola pod wykresem funkcji na zadanym przedziale

Przykładowe rozwiązanie:

1

Implementację algorytmu obliczania przybliżonej wielkości pola obszarów zamkniętych metodą prostokątów zaczniemy od stworzenia funkcji, której pole pod wykresem będziemy szacować.

2

Funkcją w naszym programie będzie $f(x) = x^2$.

```
public static double
f(double x) {
    return x * x;
}
```

Chcemy, żeby program umożliwiał wstawianie dużej liczby prostokątów do badanego obszaru w celu obliczenia z pewną dokładnością pola między wykresem funkcji a osią x. Potrzebny nam jest typ danych, który będzie w stanie przechować duże liczby i ułamki. Wybierzemy typ `double`.

Kolejnym krokiem będzie inicjacja zmiennych.

3

```
double xPoczatek = 1,
xKoniec = 3,
podstawaProstokata, wynik
= 0;
double liczbaProstokatow =
200;
```

4

Zmienna `xPoczatek` reprezentuje dolną granicę przedziału, w naszym przypadku 1, a zmienna `xKoniec` reprezentuje górną granicę przedziału. Ponieważ podstawa prostokąta będzie taka sama dla każdej figury, obliczamy ją raz i zapiszemy w zmiennej `podstawaProstokata`. Zmienna `wynik` będzie przechowywać sumę pól prostokątów, a zmienna `liczbaProstokatow` będzie przechowywać liczbę prostokątów.

5

Mając już wszystkie dane, możemy obliczyć podstawę każdego z prostokątów.

```
1 double xPoczatek = 1,
  xKoniec = 3,
  podstawaProstokata, wynik
  = 0;
2 double liczbaProstokatow =
  200;
3
4 podstawaProstokata =
  (xKoniec - xPoczatek) /
  liczbaProstokatow;
```

6

Wynik działania `xKoniec - xPoczatek` daje długość obszaru przedziału, który następnie dzielimy na liczbę prostokątów. Dzięki temu otrzymujemy podstawę każdego z prostokątów. Obliczamy pole każdego z prostokątów, dlatego potrzebujemy pętli, która pozwoli po nich przejść. Skoro wiemy, ile iteracji będzie potrzebnych (wiemy, ile jest prostokątów), użyjemy pętli `for`.

7

```
1 double xPoczatek = 1,
  xKoniec = 3,
  podstawaProstokata, wynik
  = 0;
2 double liczbaProstokatow =
  200;
3
4 podstawaProstokata =
  (xKoniec - xPoczatek) /
  liczbaProstokatow;
5
```

```
6 for (int i = 0; i <
  liczbaProstokatow; i++) {
7
8 }
```

8

Wewnątrz pętli należy dodać do zmiennej `wynik` obliczone pole aktualnego `i`-tego prostokąta. Zauważ, że długością wysokości prostokąta jest wartość funkcji w początku przedziału będącego podstawą prostokąta. Jest to wariant metody prostokątów, który nazywamy wariantem lewych prostokątów. Możemy zredukować liczbę zbędnych operacji mnożenia (obliczania pól prostokątów), poprzez wyłączenie wspólnego czynnika przed nawias. Dzięki temu wewnątrz pętli będą jedynie sumowane wysokości prostokątów, a po jej zakończeniu pomnożymy sumę przez podstawę prostokąta.

```
1 public class Main {
2     public static void
  main (String [] args) {
3         double xPoczatek =
  1, xKoniec = 3,
  podstawaProstokata, wynik
  = 0;
4         double
  liczbaProstokatow = 200;
5
6         podstawaProstokata
  = (xKoniec - xPoczatek) /
  liczbaProstokatow;
7
8         for (int i = 0; i
  < liczbaProstokatow; i++)
  {
9             wynik = wynik
  + f(xPoczatek +
  podstawaProstokata * i);
```

```
        }

    wynik = wynik * podstawaProstokata;
```

Ostatnim krokiem jest wypisanie szacowanej wartości pola pod wykresem.

9

Oto kompletny kod programu:

```
public class Main {
    public static void main (String [] args) {
        double xPoczatek = 1, xKoniec = 3, podstawaProstokata, wynik = 0;
        double liczbaProstokatow = 200;

        podstawaProstokata = (xKoniec - xPoczatek) / liczbaProstokatow;

        for (int i = 0; i < liczbaProstokatow; i++) {
            wynik = wynik + f(xPoczatek + podstawaProstokata * i);
        }

    wynik = wynik * podstawaProstokata;


System.out.println("Szacowane pole pod wykresem: " + wynik);
    }

```

```
17      public static double
f(double x) {
18          return x * x;
19      }
20  }
```

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Dana jest funkcja $f\left(x\right) = x^2 \cdot \left(x + 10\right)$. Napisz program, który oszacuje wartość pola pod wykresem funkcji, używając metody prostokątów. Swój program przetestuj dla przedziału $\langle 5,\ 11 \rangle$ i 100 prostokątów.

Specyfikacja problemu:

Dane:

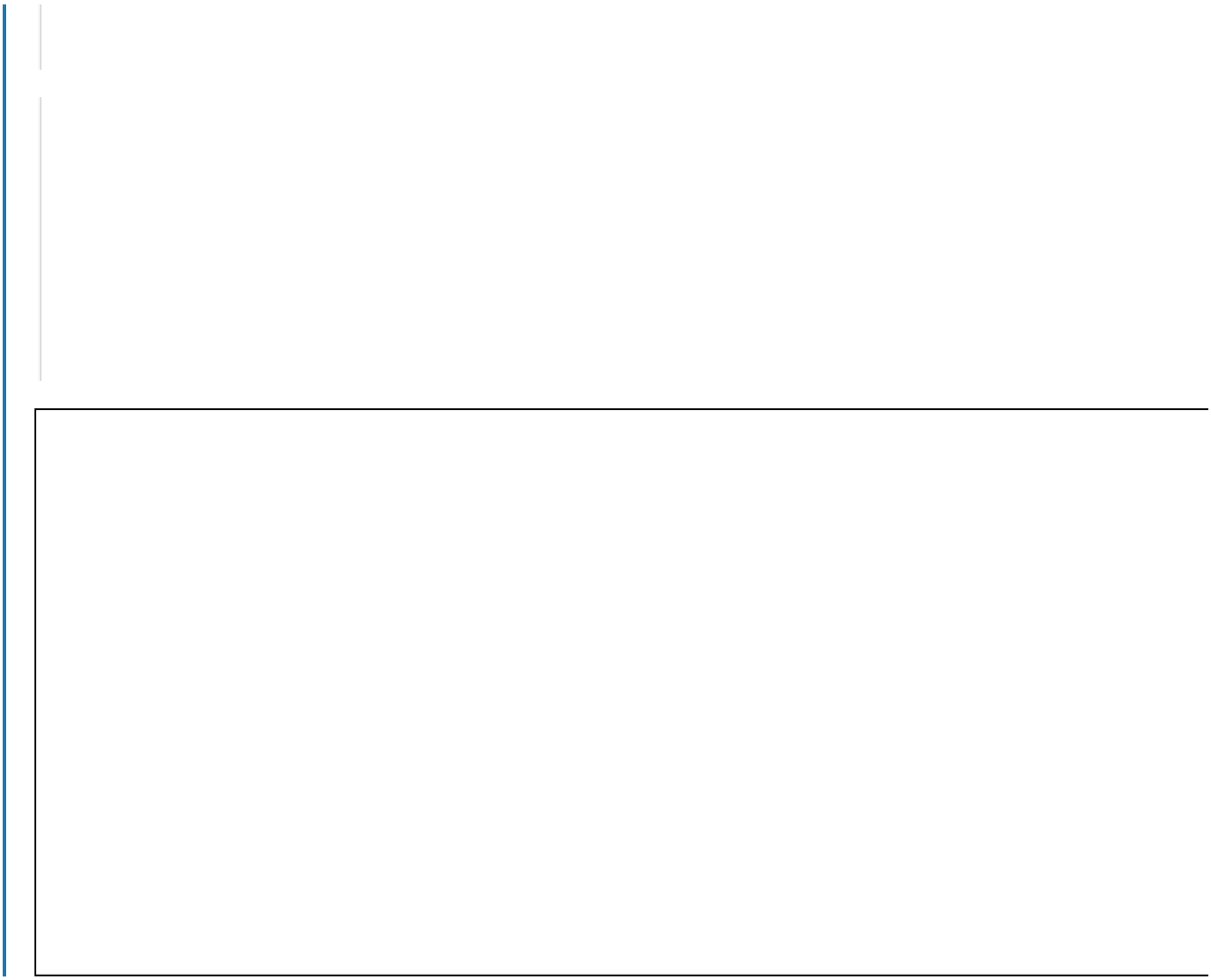
- `xPoczatek` – liczba zmiennoprzecinkowa; początek przedziału
- `xKoniec` – liczba zmiennoprzecinkowa; koniec przedziału
- `liczbaProstokatow` – liczba całkowita; liczba prostokątów używanych do obliczenia przybliżonej wielkości pola obszaru zamkniętego

Wynik:

- `wynik` – liczba rzeczywista; przybliżona wartość pola pod wykresem funkcji na zadanym przedziale

Twoje zadania

1. Program szacuje wartość pola pod wykresem funkcji.



Ćwiczenie 2

Dana jest funkcja $f\left(x\right) = x^2 \cdot \left(x + 10\right)$. Napisz program, który oszacuje wartość pola pod wykresem funkcji, używając metody trapezowej. Swój program przetestuj dla przedziału $\langle 5,\ 11 \rangle$ i 100 trapezów.

Specyfikacja problemu:

Dane:

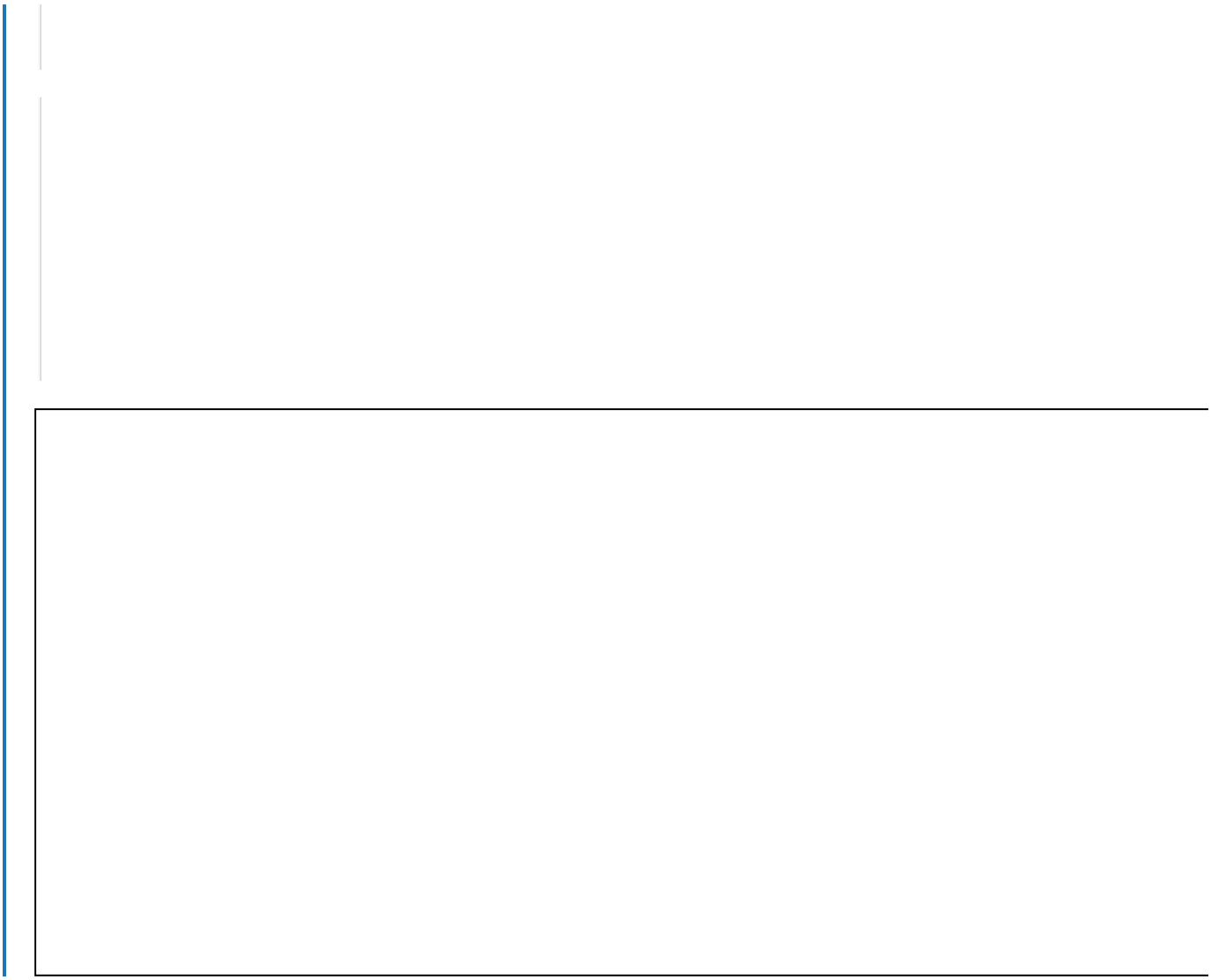
- `xPoczatek` – liczba zmiennoprzecinkowa; początek przedziału
- `xKoniec` – liczba zmiennoprzecinkowa; koniec przedziału
- `liczbaTrapezow` – liczba całkowita; liczba trapezów używanych do obliczenia przybliżonej wielkości pola obszaru zamkniętego

Wynik:

- `wynik` – liczba rzeczywista; przybliżona wartość pola pod wykresem funkcji na zadanym przedziale

Twoje zadania

1. Program szacuje wartość pola pod wykresem funkcji.



Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Obliczanie przybliżonej wielkości pola obszarów zamkniętych w języku Java

Grupa docelowa:

Szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres 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 rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) do realizacji rozwiązania problemu dobiera odpowiednią metodę lub technikę algorytmiczną i struktury danych;

4) ilustruje i wyjaśnia rolę pojęć, obiektów i operacji matematycznych w projektowaniu rozwiązań problemów informatycznych i z innych dziedzin, posługuje się pojęciem logarytmu;

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

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

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

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

2) wykorzystuje znane sobie algorytmy przy rozwiązywaniu i programowaniu rozwiązań następujących problemów:

f) obliczanie przybliżonej wielkości pola obszarów zamkniętych;

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):

- Użyjesz w praktyce wiedzy na temat wyznaczania przybliżonej wielkości pól obszarów zamkniętych metod prostokątów i trapezów.
- Przeanalizujesz zaimplementowane w języku Java algorytmy, które pozwalają na obliczenie pola obszaru ograniczonego wykresem funkcji oraz osią x.
- Rozwiążesz kilka zadań dotyczących zagadnienia obliczania przybliżonej wielkości pola obszarów zamkniętych.

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: „Obliczanie przybliżonej wielkości pola obszarów zamkniętych w języku Java". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Chętna lub wybrana osoba przedstawia najważniejsze informacje związane z całkowaniem numerycznym. W razie potrzeby nauczyciel uzupełnia wypowiedź.
2. Nauczyciel wprowadza uczniów szczegółowo w temat lekcji i jej cele. Może posłużyć się wyświetloną na tablicy zawartością sekcji „Wprowadzenie".

Faza realizacyjna:

1. Uczniowie w parach dyskutują na temat rozwiązania problemu 1 z sekcji „Przeczytaj". Chętne lub wybrane osoby przedstawiają projekt swojego rozwiązania. Nauczyciel wskazuje ewentualne problemy. Następnie uczniowie indywidualnie piszą program rozwiązujący ten problem. Chętne lub wybrane osoby prezentują swoje implementacje. Klasa dyskutuje na ich temat. Uczniowie oglądają film z rozwiązaniem.
2. **Praca z multimedium.** Uczniowie w grupach analizują polecenie 1 z sekcji „Prezentacja multimedialna". Następnie zapoznają się z prezentacją i implementują program wykorzystujący metodę prostokątów do liczenia pola powierzchni pod wykresem funkcji.
3. **Ćwiczenie umiejętności.** Nauczyciel wyświetla zawartość sekcji „Sprawdź się" i poleca uczniom wykonanie w parach ćwiczenia 1. Po wykonaniu zadania, uczniowie porównują swoje rozwiązanie z inną grupą.
4. Uczniowie indywidualnie rozwiązują ćwiczenie 2 z sekcji „Sprawdź się". Następnie na forum klasy chętne osoby prezentują swoje rozwiązanie.

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".
2. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności.

Praca domowa:

1. Uczniowie wykonują ćwiczenie 3 z sekcji „Sprawdź się".

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 multimedium w sekcji „Prezentacja multimedialna” do przygotowania się do lekcji powtórkowej.