Wyznaczanie liczby π metodą Monte Carlo w języku Java

Źródło: Nadine Shaabana, domena publiczna.

Istnieje wiele rodzajów obliczeń matematycznych, które bardzo trudno jest wykonać, korzystając z metod analitycznych. Czasami okazuje się to wręcz niemożliwe. Za przykład niech posłużą obliczenia całek oznaczonych pewnych funkcji, wyznaczanie pól powierzchni figur o skomplikowanych kształtach lub działania na liczbach niewymiernych.

W takich sytuacjach sięga się po numeryczne techniki obliczeniowe. Jednej z nich – metodzie Monte Carlo – poświęcimy ten e-materiał.

Czy potrafisz wskazać inne sytuacje, w których sięga się po numeryczne techniki obliczeniowe?

Podstawowe informacje na temat wyznaczania liczby π metodą Monte Carlo znajdziesz w e-materiale: Wyznaczanie liczby π metodą Monte Carlo.

W tym e-materiale poznamy specyfikę wyznaczania liczby π metodą Monte Carlo w języku Java.

Jeśli chcesz dowiedzieć się, jak to zagadnienie wygląda w przypadku innych języków programowania, sięgnij do e-materiałów:

- Wyznaczanie liczby π metodą Monte Carlo w języku Python,
- Wyznaczanie liczby π metodą Monte Carlo w języku C++.

> **Twoje cele**
>
> - Scharakteryzujesz sposoby generowania liczb losowych w języku Java.
> - Przeanalizujesz, jak losuje się punkty w obrębie danej figury geometrycznej.
> - Zaimplementujesz algorytm obliczania wartości liczby π metodą Monte Carlo w języku Java.

Przeczytaj

Przypomnienie wiadomości

Wiemy, jakie są założenia metody Monte Carlo. Przypomnijmy, w jaki sposób można wyznaczyć liczbę π za jej pomocą.

Podczas przeprowadzania symulacji metodą Monte Carlo korzystamy z nierówności koła o środku w punkcie (a, b) i promieniu r, którą przedstawimy następująco:

$$(x-a)^2 + (y-b)^2 \leq r^2$$

Wybieramy punkt o środku w punkcie (0,0) i promieniu 1. Kwadrat opisany na tym kole może być przestrzenią ograniczoną warunkami $x \in \langle -1,\ 1 \rangle$ oraz $y \in \langle -1,\ 1 \rangle$.

By przypomnieć sobie, w jaki sposób obliczaliśmy omawianą metodą liczbę π, wróćmy do metafory rzucania długopisem. Rzucany nim w kartkę, na której narysowano kwadrat opisany na kole.

Jeśli będziesz rzucać długopisem dostatecznie długo, kropki tuszu pokryją cały rysunek. W konsekwencji stosunek liczby kropek wewnątrz koła do liczby wszystkich kropek będzie zbliżony do stosunku pola koła do pola kwadratu.

Znając wzór na pole kwadratu oraz pole koła, możemy wyznaczyć wartość liczby π. W e-materiale Wyznaczanie liczby π metodą Monte Carlo wyznaczyliśmy wzór na przybliżenie liczby π. Przypomnijmy go:

$$\pi \approx 4 \cdot \frac{l_{trafień}}{l_{rzutów}}$$

Wyznaczanie liczby π metodą Monte Carlo w języku Java

Aby zaimplementować opisany algorytm, zastanówmy się, w jaki sposób wyznaczyć losową liczbę z przedziału $\langle -1,\ 1 \rangle$. W języku Java dysponujemy wieloma generatorami liczb pseudolosowych. Użyjemy funkcji `random()` z biblioteki `Math`. Funkcja ta zwraca

losowe liczby zmiennoprzecinkowe z przedziału $\langle 0,\ 1)$ z jednakowym prawdopodobieństwem. Nie musimy przejmować się w tym wypadku brakiem możliwości wylosowania liczby 1, ponieważ prawdopodobieństwo tego zdarzenia jest bardzo małe. Algorytm nie straci na dokładności.

Aby wylosować liczbę z przedziału $\langle -1,\ 1)$, przeskalujmy przedział. Wylosowaną liczbę należy pomnożyć przez 2, a następnie odjąć od niej 1. Dzięki takim operacjom przedział zostanie rozciągnięty, a następnie przesunięty. Zapiszmy kod w języku Java:

```
// losowa liczba z przedziału od -1 do 1
double randomDouble = 2 * Math.random() - 1;
```

Napiszmy funkcję, która wyznaczy liczbę π metodą Monte Carlo:

```
static double wyznaczPi(int liczbaLosowan) {
    int liczbaTrafien = 0;

    for (int i = 0; i < liczbaLosowan; ++i) {
        // losowe liczby z przedziału <-1, 1)
        double x = 2 * Math.random() - 1;
        double y = 2 * Math.random() - 1;

        if (x * x + y * y <= 1) {
            ++liczbaTrafien;
        }
    }

    double pi = (double) 4 * liczbaTrafien / liczbaLosowan;

    return pi;
}
```

Aby przetestować napisany algorytm, utwórzmy główną klasę oraz funkcję `main()`:

```java
public class Main {
    static double wyznaczPi(int liczbaLosowan) {
        int liczbaTrafien = 0;

        for (int i = 0; i < liczbaLosowan; ++i) {
            // losowe liczby z przedziału <-1, 1)
            double x = 2 * Math.random() - 1;
            double y = 2 * Math.random() - 1;

            if (x * x + y * y <= 1) {
                ++liczbaTrafien;
            }
        }

        double pi = (double) 4 * liczbaTrafien / liczbaLosowan;

        return pi;
    }

    public static void main(String[] args) {
        System.out.println(wyznaczPi(10000));
    }
}
```

Dla liczby losowań 10 000 możemy spodziewać się wyników zbliżonych do liczby π, jednak należy pamiętać, że nie zawsze będą one dokładne. Im więcej losowych punktów, tym dokładniejszy wynik. Przetestuj działanie programu, podając różne liczby losowań. Przeanalizuj, jak zmienia się wynik. Przypomnijmy, że $\pi \approx 3.1415$.

Dokładność wyniku

Przetestujmy działanie programu dla różnych parametrów. Przeprowadźmy próby uruchomienia funkcji `wyznaczPi()` z parametrami 10, 100, 1000, 10 000, 100 000 oraz 1 000 000.

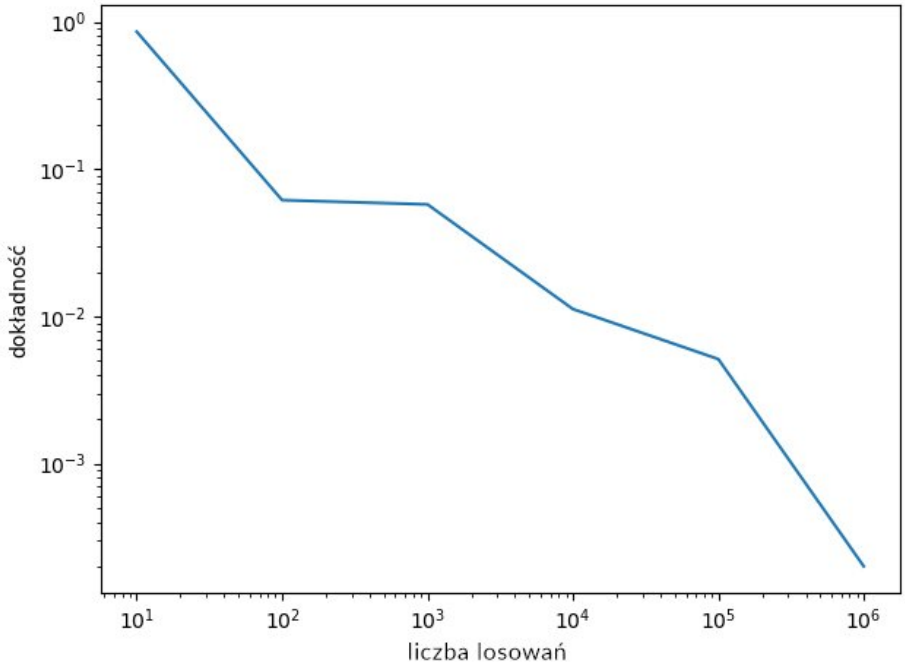
Aby utworzyć wykres dokładności, możemy obliczyć wartość bezwzględną różnicy pomiędzy wynikiem teoretycznym, a otrzymanym dzięki metodzie Monte Carlo. W tym celu wykorzystamy stałą `PI` z biblioteki `Math` oraz funkcję `abs()`, również z biblioteki `Math`:

```
public class Main {
    static double wyznaczPi(int liczbaLosowan) {
        int liczbaTrafien = 0;

        for (int i = 0; i < liczbaLosowan; ++i) {
            // losowe liczby z przedziału <-1, 1)
            double x = 2 * Math.random() - 1;
            double y = 2 * Math.random() - 1;

            if (x * x + y * y <= 1) {
                ++liczbaTrafien;
            }
        }

        double pi = (double) 4 * liczbaTrafien / liczbaLosowan;

        return pi;
    }

    public static void main(String[] args) {
        System.out.println(Math.abs(Math.PI - wyznaczPi(10)));
        System.out.println(Math.abs(Math.PI - wyznaczPi(100)));
        System.out.println(Math.abs(Math.PI - wyznaczPi(1000)));
        System.out.println(Math.abs(Math.PI - wyznaczPi(10000)));
        System.out.println(Math.abs(Math.PI - wyznaczPi(100000)))
        System.out.println(Math.abs(Math.PI - wyznaczPi(1000000))
    }
}
```

Oto przykładowe wyniki, jakie otrzymamy po uruchomieniu algorytmu:

```
0.8584073464102069
0.06159265358979045
0.05759265358979304
0.011207346410206931
0.005127346410207068
2.006535897929318E-4
```

Analizując otrzymane dane, można zauważyć, że dokładność zwiększa się wraz ze wzrostem liczby losowań. Wykres danych w skali logarytmicznej prezentuje się następująco:



Słownik

funkcja abs()

służy do obliczenia wartości bezwzględnej wyrażenia

generator liczb pseudolosowych

podprogram zwracający kolejne liczby z deterministycznego ciągu liczb, który ma podobne własności do ciągów losowych

Symulacja interaktywna

Polecenie 1

Przeanalizuj symulację interaktywną, która ukazuje proces wyznaczania liczby π metodą Monte Carlo.

Symulacja 1

Polecenie 2

Poszukaj informacji na temat tego, do czego może przydać się wyznaczone przybliżenie liczby Pi.

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który za pomocą metody Monte Carlo obliczy pole koła. Przetestuj swój program dla koła o promieniu $r = 2$. Wynik wypisz z dokładnością do dwóch miejsc po przecinku (nie zaokrąglaj).

Specyfikacja:

Dane:

- `r` – promień koła, którego pole należy obliczyć; liczba rzeczywista

Wynik:

Program wyświetla pole koła o zadanym promieniu.

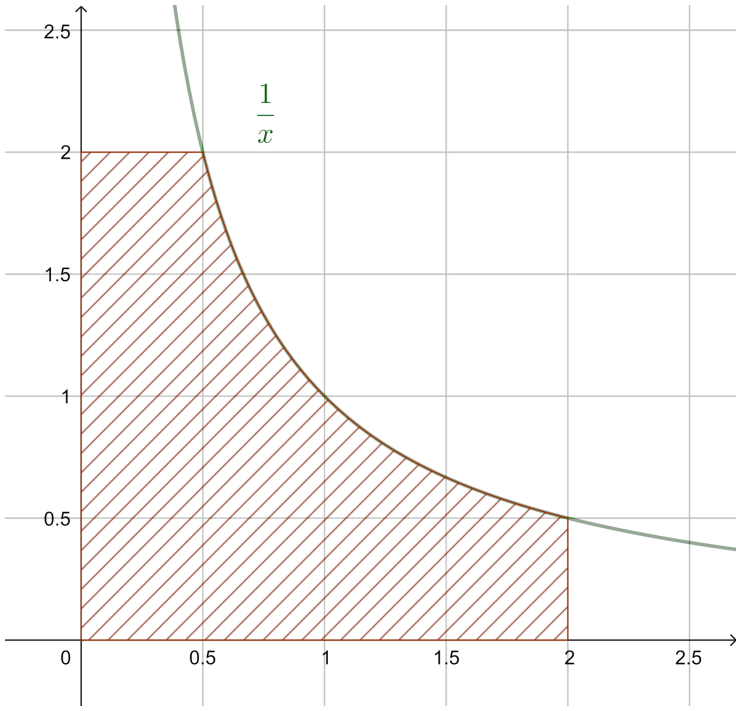
Twoje zadania

1. Program oblicza pole koła o danym promieniu metodą Monte Carlo.

Polecenie 1

Zapoznaj się z materiałem źródłowym i wykonaj ćwiczenie nr 2.

Materiał źródłowy do ćwiczenia nr 2



Ćwiczenie 2

Napisz program, który za pomocą metody Monte Carlo obliczy pole powierzchni figury zawartej w kwadracie o wierchołkach $(0,0)$, $(p,0)$, (p,p), $(0,p)$ i leżącej pod wykresem funkcji f. W tym celu należy losować punkty z podanego kwadratu, a następnie sprawdzać, czy leży on pod wykresem funkcji $y = f(x)$. Przestestuj swój algorytm dla $p = 2$ i $f(x) = \frac{1}{x}$. Wynik wypisz z dokładnością do dwóch miejsc po przecinku (nie zaokrąglaj).

Specyfikacja:

Dane:

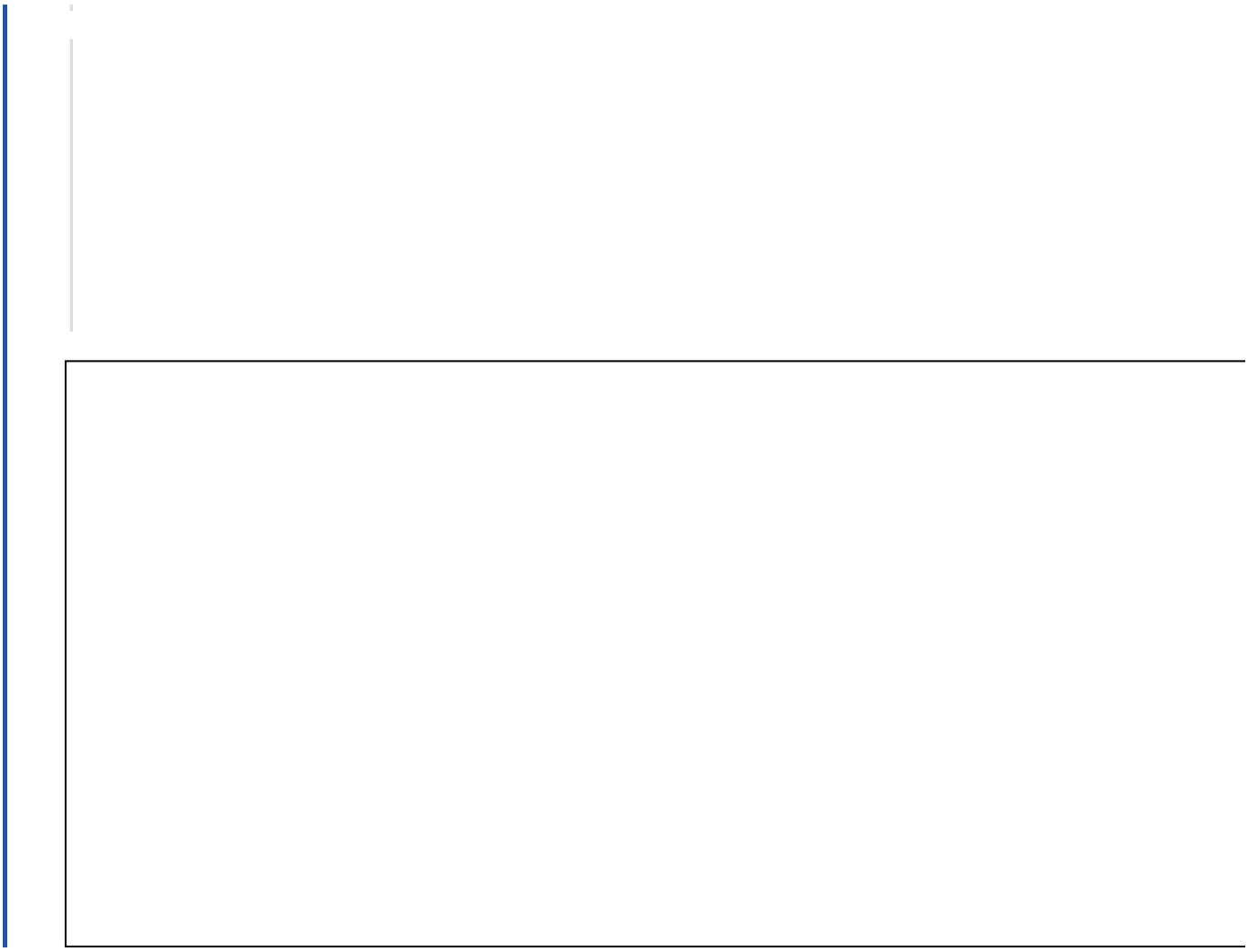
- p – bok kwadratu, który zawiera pole figury; liczba rzeczywista
- f – funkcja, której wykres ogranicza pole figury

Wynik:

Program oblicza i wyświetla pole danej figury.

Twoje zadania

1. Program oblicza pole powierzchni danej figury metodą Monte Carlo.



Dla nauczyciela

Autor: Maurycy Gast

Przedmiot: Informatyka

Temat: Wyznaczanie liczby π metodą Monte Carlo w języku Java

Grupa docelowa:

Liceum ogólnokształcące i technikum, 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

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

Zakres podstawowy. Uczeń:

2) do realizacji rozwiązań problemów prawidłowo dobiera środowiska informatyczne, aplikacje oraz zasoby, wykorzystuje również elementy robotyki;

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:

3) objaśnia, a także porównuje podstawowe metody i techniki algorytmiczne oraz struktury danych, wykorzystując przy tym przykłady problemów i algorytmów, w szczególności:

h) metodę Monte Carlo (obliczanie przybliżonej wartości liczby π, symulacja ruchów Browna),

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):

- Scharakteryzujesz sposoby generowania liczb losowych w języku Java.
- Przeanalizujesz, jak losuje się punkty w obrębie danej figury geometrycznej.
- Zaimplementujesz algorytm obliczania wartości liczby π metodą Monte Carlo w języku Java.

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;
- telefony z dostępem do internetu;
- 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: „Wyznaczanie liczby π metodą Monte Carlo w języku Java". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wprowadza uczniów szczegółowo w temat lekcji i jej cele. Może posłużyć się wyświetloną na tablicy zawartością sekcji „Wprowadzenie”.
2. Prowadzący prosi uczniów, aby zgłaszali swoje propozycje pytań do tematu. Jedna osoba może zapisywać je na tablicy. Gdy uczniowie wyczerpią swoje pomysły, a pozostały jakieś ważne kwestie do poruszenia, nauczyciel je dopowiada.

Faza realizacyjna:

1. **Praca z tekstem.** Uczniowie przystępują do cichego czytania tekstu e-materiału. Indywidualnie zapoznają się z treścią w sekcji „Przeczytaj”.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Symulacja interaktywna”. Uczniowie wspólnie analizują symulację interaktywną, która ukazuje proces wyznaczania liczby *pi* metodą Monte Carlo.
3. Liga zadaniowa – uczniowie pracując w parach, wykonują na czas ćwiczenie nr 1 z sekcji „Sprawdź się”: „Napisz program, który za pomocą metody Monte Carlo obliczy pole okręgu o promieniu 2. Wynik wypisz z dokładnością do dwóch miejsc po przecinku (nie zaokrąglaj)”. Następnie dzielą się swoimi wynikami przez porównywanie napisanego kodu z inną grupą, która również zakończyła zadanie.

Faza podsumowująca:

1. Na koniec zajęć z programowania w Javie nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak…”.

Praca domowa:

1. Uczniowie wykonują ćwiczenie nr 2 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:

- Przed lekcją wybrane lub chętne osoby mogą przygotować krótkie wystąpienia, podczas których zaprezentują ciekawostki związane z liczbą Pi.