INF4

Arbeitsblatt: INF2

Name: _____ **Kurznamen:** _____

Grundlagen Java-Applikationen

Einführung

In diesem Praktikum werden Sie Java-Applikationen schreiben, die auf der Kommandozeile arbeiten. Im Laufe des Semesters werden wir zusätzlich Java-Applikationen mit grafischer Ausgabe einführen. Java-Applikationen bestehen aus einer oder mehreren Klassen. Eine davon muss eine statische (*static*) und öffentliche (*public*) *main*-Methode enthalten. Über diese Klasse wird die Applikation i.d.R. gestartet.

Eine Methode, die aus einer statischen Methode derselben Klasse heraus gestartet werden soll, muss ebenfalls statisch sein. Den Grund dafür werden wir im Zusammenhang mit Klassen und Objekten ausführlich besprechen.

Aufgabe 1: Pi Berechnung nach Leibniz

Es soll eine Java-Methode `public static double pi(int n)` die Formel

$$\sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots = \frac{\pi}{4}$$

zur Berechnung der Kreiszahl π geschrieben werden, die als Näherungswert für *pi (k)* die ersten k Terme der Reihenentwicklung aufsummiert - und am Schluss mit 4 multipliziert.

Aufgaben

a) Implementieren Sie `pi` als Java-Methode.

b) Implementieren Sie ausserdem eine *main*-Methode, in der n eingelesen wird. Anschliessend soll π berechnet und das Ergebnis ausgegeben werden.

c) Bestimmen Sie in `int correctDigits` die *Anzahl Stellen Genauigkeit*, die Sie erreicht haben, z.B. 3.148776 soll 3 ergeben. Auch dieser Wert soll ausgegeben werden.

Hinweise

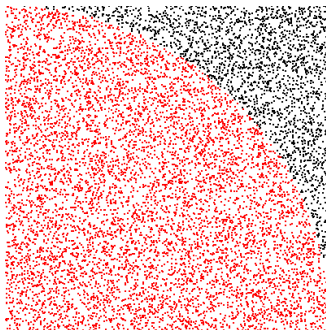
- Berechnen Sie in einer Schleife den jeweils neuen Summanden der Reihenentwicklung aus.
- Math.PI steht die Kreiszahl als Konstante mit hoher Genauigkeit zur Verfügung
- Verwenden Sie das Gerüst

Abgabe

Praktikum: INF4.1

Filename: Leibniz.java

Aufgabe 2: Pi mittels Monte-Carlo Verfahren



Die Zahl π lässt sich mittels einer Monte-Carlo Simulation bestimmen. Ein einfaches, aber nicht sehr genaues Verfahren funktioniert folgendermassen: Es werden Punkte innerhalb des Einheitsquadrates zufällig bestimmt. Ist der Abstand zum Ursprung kleiner als 1, dann zählt man ihn zur roten Menge. Die Anzahl der roten Punkte dividiert durch die Gesamtzahl der Versuche ergibt eine Näherung für $\pi/4$.

Schreiben Sie folgende Methoden:

a) eine Methode `pi`, die n Schritte der Simulation ausführt (mit n als Parameter)

b) eine *`main`*-Methode, soll die Anzahl Wiederholungen einlesen und das Resultat ausgeben.

Hinweise:

- *Math.random()* liefert eine Zufallszahl zwischen 0 und 1
- Wenn Sie 1 als Radius wählen, dann brauchen Sie für obige Methode keine Wurzelfunktion um zu bestimmen ob sie innerhalb des Einheitskreises sind.

Abgabe

Praktikum: INF4.2

Filename: Montecarlo.java