Institut für Angewandte und
Numerische Mathematik

PD Dr. Mathias J. Krause
M.Sc. Stefan Karch
M.Sc. Mariia Sukhova

27.10.2023

Einstieg in die Informatik und Algorithmische Mathematik

Aufgabenblatt 3

Bearbeitungszeitraum: 13.11.2023 – 24.11.2023

Aufgabe 1 *Summen*

Schreiben Sie mit Java ein Programm mit dem Namen `Summen`. Erstellen Sie in der Programmklasse eine `main`–Methode, und setzen Sie in dieser Methode folgende Schritte um:

- Lesen Sie eine ganzzahlige Variable von der Konsole ein, und speichern Sie diese ab. Berechnen Sie anschließend die Summe

  $$q_n = \sum_{k=1}^{n} k^2 = 1^2 + 2^2 + 3^2 + 4^2 + \cdots + n^2$$

  der ersten n Quadratzahlen mit einer **`for`**-Schleife. Geben Sie das Ergebnis auf der Konsole aus.

- Berechnen Sie anschließend mit einer **`while`**-Schleife die Summe

  $$u_n = \sum_{k=1}^{n} (2k-1) = 1 + 3 + 5 + \cdots + (2n-1)$$

  der ersten n ungeraden Zahlen, und geben Sie das Ergebnis auf der Konsole aus.

- Natürlich können die Summen q_n und u_n auch direkt berechnet werden: Es gilt

  $$q_n = \frac{n(n+1)(2n+1)}{6} \quad \text{und} \quad u_n = n^2.$$

  Geben Sie auch dieses Ergebnis auf der Konsole aus und vergleichen Sie es mit den Obigen.

Musterlösung: Für $n = 6$ gilt $q_n = 91$ und $u_n = 36$.

Aufgabe 2 *Boolean Algebra*

Schreiben Sie ein Java Program, das zwei ganzzahlige Variablen a und b und einen **`boolean`** c einliest. Das Programm soll **`true`** ausgeben, wenn

(a) `a` und `b` positiv sind,

(b) `a` gleich 1 und `b` nicht gleich 0,

(c) `a` kleiner gleich 2 oder größer als 5 und `c` wahr ist,

(d) `a` gleich `b` und `c` **`true`** oder `a` ungleich `b` und `c` **`false`**,

(e) `a` und `b` unterschiedliche Vorzeichen haben, es sei denn der Parameter `c` ist **`true`**, dann nur, wenn beide negativ sind.

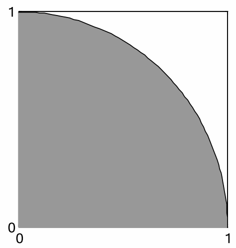
Sie können das Programm mit dem Input `a = 2`, `b = -1` und `c = true` testen. Die Ausgabe sieht dann folgendermaßen aus:

```
a: false
b: false
c: true
d: false
e: false
```

Hinweis: Der Ausdruck `System.out.println(a == a)` gibt **`true`** auf der Konsole aus.

Aufgabe 3 (Pflichtaufgabe) *Monte-Carlo-Methode*

Geben sei ein Quadrat mit der Kantenlänge $r = 1$. In das Quadrat sei ein Viertelkreis mit Radius $r = 1$ eingeschrieben, wie im folgenden Bild dargestellt. Stellen Sie sich nun vor, dass ein herab-



fallendes Staubkorn innerhalb des Rechtecks landet. Wie groß ist dann die Wahrscheinlichkeit p dafür, dass das Staubkorn innerhalb des Viertelkreises landet? Offenbar ist diese Wahrscheinlichkeit durch das Verhältnis der Viertelkreisfläche A zur Quadratfläche $A_\square$ gegeben, d.h. es gilt

$$p = \frac{A}{A_\square} = \frac{\frac{1}{4}\pi r^2}{r^2} = \frac{1}{4}\pi. \tag{1}$$

Angenommen, man lässt nun N Staubkörner in das Quadrat fallen und zählt die Anzahl M der Staubkörner, die im Viertelkreis landen. Werden unendlich viele Staubkörner N in das Quadrat fallen gelassen, so ergibt sich

$$\frac{M}{N} \to p. \tag{2}$$

Aus den Gleichungen (1) und (2) erhält man für unendlich viele Staubkörner N

$$\pi_N := \frac{4M}{N} \to \pi. \tag{3}$$

Schreiben Sie mit Java ein Programm, das die Gleichung (3) empirisch bestätigt. Gehen Sie dabei wie folgt vor:

- Erstellen Sie die `main`–Methode in einer öffentlichen Klasse mit dem Namen `MonteCarlo`. Erzeugen Sie zwei ganzzahlige Variablen für die Größen M und N. Die Variable M soll mit dem Wert 0 initialisiert werden. Zudem soll die Variable, in der N abgespeichert wird, zunächst mit 10 initialisiert werden.

- Erstellen Sie eine **`for`**–Schleife, die N-mal durchlaufen wird. Erzeugen Sie in dieser Schleife zwei im Intervall $[0, 1)$ gleichverteilte Zufallszahlen x und y, und speichern Sie diese in zwei Gleitkomma - Variablen ab. Die Zufallszahlen x und y sollen dabei die Koordinaten eines Punktes im Quadrat wiedergeben, auf dem ein herabfallendes Staubkorn landet. Bei jedem Schleifendurchlauf wird so die Landung eines neuen Staubkorns simuliert.

  Hinweis: Eine im Intervall $[0, 1)$ gleichverteilte Zufallszahl kann in Java mit der Methode `Math.random()` erzeugt werden.

- Überprüfen Sie für jedes gelandete Staubkorn, ob es im Innern des Viertelkreises gelandet ist. Dies ist genau dann der Fall, wenn die Koordinaten x und y des Landungspunktes die Ungleichung

  $$x^2 + y^2 \leq 1 \tag{4}$$

  erfüllen. Erhöhen Sie den Wert von M um 1, wenn das Staubkorn im Viertelkreis gelandet ist. Verwenden Sie hierzu eine **`if`**–Anweisung.

- Berechnen Sie nach der Schleife einen Näherungswert π_N für die Kreiszahl π gemäß Formel (3). Achten Sie dabei darauf, die Zahlkonstante 4 als `4.0` zu schreiben, da Java sonst eine ganzzahlige Division durchführt.

  Berechnen Sie außerdem den absoluten Fehler e_{abs} des Näherungswertes. Der absolute Fehler ist durch

  $$e_{abs} = |\pi - \pi_N| \tag{5}$$

  gegeben. Geben Sie den Näherungswert π_N und den absoluten Fehler e_{abs} auf der Konsole aus.

  Hinweis: Die Betragsfunktion $|\cdot|$ heißt unter Java `Math.abs()`. Ferner erhält man die Kreiszahl π in Java durch den Ausdruck `Math.PI` .

- Testen Sie ihr Programm mehrmals für $N = 100$, $N = 10.000$ und $N = 1.000.000$. Was kann man beobachten? Ist dieses Programm Ihrer Meinung nach geeignet, die Kreiszahl π anzunähern? Betrachten Sie dazu einmal die Dezimaldarstellung des Bruches $\frac{355}{113}$.

Fragen:

- Was ist der Unterschied zwischen einer **`while`**–Schleife und einer **`do`**–**`while`**–Schleife?

- Wie kann man eine Schleife in Java vorzeitig abbrechen?