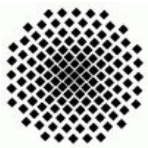

Universität Stuttgart
Fakultät 5, Institut FMI
Einführung in die Informatik III

Wintersemester 2007/2008
Claus, Riexinger
28.10.2007

Aufgabenblatt 1 **Abgabe: 04.11.2007, 24.00 Uhr**

---

1. π berechnen (leicht) (4 Punkte)

Wie Sie wissen, bezeichnet die Zahl π die Fläche des Einheitskreises, d.h. des Kreises mit Mittelpunkt $(0, 0)$ und Radius 1. Umgekehrt heißt das, dass man π berechnen kann, indem man die Fläche des Einheitskreises misst. Dazu kann man sich einer wahrscheinlichkeitstheoretischen Methode (einer sogenannten Monte-Carlo-Methode) bedienen. Diese geht wie folgt:

> Wähle n-mal ($n > 1000000$ ist hierbei empfohlen) einen Punkt (x, y) zufällig gleichverteilt aus dem Quadrat $[-1, 1] \times [-1, 1]$ aus. Falls (x, y) innerhalb des Einheitskreises liegt, nennen wir das einen Treffer. Wenn n gegen unendlich tendiert, wird das Verhältnis der Treffer zur Anzahl der ausgewählten Punkte sich dem Verhältnis der Fläche des Einheitskreises zu der des Quadrats $[-1, 1] \times [-1, 1]$, d.h. $\pi/4$ annähern.

Schreiben Sie ein Java-Programm, das auf diese Weise π näherungsweise berechnet. Führen Sie die Auswahl der zufälligen Punkte sehr lange aus und lassen Sie sich alle 1000 Punkte den neu errechneten Schätzwert für π ausgeben.

2. Klasse für komplexe Zahlen (leicht) (8 Punkte)

Entwerfen Sie eine Klasse für komplexe Zahlen. Ihre Klasse soll über folgende Bestandteile verfügen:

- Konstruktoren für Polarform und kartesische Form
- Methoden zur Ausgabe (auf der Konsole) in beiden Formen
- Methoden (es dürfen auch Operatoren überladen werden) zur Division, Multiplikation, Addition, Subtraktion, Wurzelziehung - die Berechnungen sollen hierbei auf der Konsole ausgegeben werden, mit sinnvollen Zwischenschritten und in der passenden Darstellung/Form

Implementieren Sie die obige Klasse in Java und entwerfen Sie ein Hauptprogramm (eine Hauptklasse) zum Test Ihrer Klasse.

3. Klasse Graph (Mittel) (8 Punkte)

Implementieren Sie eine Klasse zur Realisierung von mit ganzen Zahlen gewichteten, gerichteten Graphen (auch mit negativen Kantengewichten!). Intern soll der Graph in Form einer Adjazenzmatrix gehalten werden. Die Anzahl der Knoten wird der Klasse im Konstruktor übergeben. Implementieren Sie folgende Funktionalität in die Klasse.

- Die Möglichkeit einen Graph zu erzeugen. (Kanteneingabe)

- Erreichbarkeitsüberprüfung - Eine Methode, die zwei Knotennummern als Parameter erhält und ausgibt, ob vom 1. Knoten der 2. erreichbar ist (definieren Sie die Erreichbarkeit zwischen 2 Knoten und geben Sie an, wie Sie diese algorithmisch überprüfen können)
- Kürzeste Wege - Eine Methode, die zwei Knotennummern als Parameter erhält und den kürzesten Weg vom 1. zum 2. Knoten ermittelt, ist kein Weg verfügbar oder existiert ein negativer Zyklus soll dies ausgegeben werden. (bsp. Bellman-Ford-Algorithmus; suchen Sie diesen aus der Literatur heraus!)

Entwerfen Sie eine Klasse zum Test Ihrer Klasse für Graphen. Einfache Testfälle für alle Funktionalitäten sind erwünscht.

4. Klasse Graph - Ersatz (Schwer) (16 Punkte)

Diese Aufgabe ist für Studenten/Studentinnen mit Vorkentnissen in Java als Ersatz für die Aufgaben 2 und 3 gedacht!

Schreiben Sie die Klasse für Graphen wie in Aufgabe 3, jedoch halten Sie den Graphen in Form einer Adjazenzliste. Erweitern Sie die Klasse um eine Methode sich selbst, also einen Graphen, zu kopieren.