1 Pi berechnen (3 Punkte)

Ein (nicht besonders effizientes) Verfahren zur Berechnung von π ist, die Nullstelle der Funktion $f(x) = \sin x$ zwischen 2 und 4 durch Intervallschachtelung zu berechnen. Man hat dabei in jeder Iteration am Anfang ein Intervall $I = [u, o]$; die Intervallmitte ist $h = \frac{o+u}{2}$. Nun verwendet man folgende Iterationsvorschrift:

$$\begin{array}{lcl} f(u) \cdot f(h) > 0 & \Rightarrow & u := h \\ f(u) \cdot f(h) < 0 & \Rightarrow & o := h \end{array}$$

und rechnet mit dem neuen Intervall $[u, o]$ weiter. (Was ist mit $f(u) \cdot f(h) = 0$?)

Berechne π auf acht Nachkommastellen genau.

Hinweis: Du kannst den Sinus eines Ausdrucks `x` mit dem Ausdruck `Math.sin(x)` berechnen. Siehe Klasse `Math`, Package `java.lang`.

2 Fibonacci-Zahlen (3 Punkte)

Die *Fibonacci-Zahlen* F_n, ursprünglich ein sehr primitives Modell der Bevölkerungsentwicklung bei Kaninchen, sind wie folgt rekursiv definiert:

$$\begin{array}{lcl} F_1 & = & 1 \\ F_2 & = & 1 \\ F_n & = & F_{n-2} + F_{n-1} \quad \text{falls } n > 2 \end{array}$$

(a) Schreibe eine Methode, die F_n rekursiv berechnet.

(b) Schreibe eine Methode, die F_n iterativ, d.h. mit einer Schleife statt mit einer Rekursion, berechnet. Du kannst mit drei Variablen für die Zwischenergebnisse auskommen.

(c) Verwende die beiden Methoden, um F_{45} zu berechnen. Welche der beiden Implementierungen braucht mehr Rechenzeit? Warum?

3 Kalender, objektorientiert (3 Punkte)

Implementiere die in der Vorlesung skizzierte Klasse `Kalender`, deren Objekte den Kalender eines bestimmten Jahres darstellen. Die Klasse soll Methoden zur Verfügung stellen, um zu prüfen, ob das Jahr ein Schaltjahr ist, und um für einen gegebenen Tag und Monat den Wochentag (als Integerwert) zu berechnen. Du wirst auch einen eigenen Konstruktor brauchen. Achte darauf, dass die Parameter und Rückgabetypen der Methoden sinnvoll sind.

Die zweite Methode soll zumindest für das Jahr 2004 funktionieren und ansonsten einen definierten Fehlerwert zurückgeben. Ersetze die beiden Hauptprogramme von letzter Woche durch zwei neue, die ein `Kalender`-Objekt erzeugen und dann ihre Ausgaben durch Aufruf von Methoden dieses Objekts berechnen.

4 Durchschnitte berechnen (3 Punkte)

Schreibe ein Programm, das auf der Kommandozeile eine durch Kommata separierte Liste von Integerliteralen annimmt und den (`double`-) Durchschnitt der Zahlen berechnet.

Beispiel (Eingaben sind *kursiv* gedruckt):
`koller@cicero:~$` *java Durchschnitt 1,-27,0*
`Durchschnitt ist -13.0`

Hinweis: Du kannst ein Objekt der Klasse `StringTokenizer` (Package `java.lang`) verwenden, um die Eingabe in ihre Bestandteile aufzubrechen, wenn Du daran denkst, dass die Ausdrücke `args[i]` den Typ `String` haben. Um das Ergebnis mit der Meldung "Durchschnitt ist ..." zusammenzusetzen, kannst Du ein Objekt der Klasse `StringBuffer` (Package `java.lang`) verwenden.

5 Rechtecke (5 Punkte)

(a) Schreibe eine Klasse `RechteckEcken`, die ein Rechteck durch die Koordinaten seiner Eckpunkte repräsentiert. Diese Klasse soll mindestens zwei Methoden unterstützen:

- das Rechteck als Zeichengrafik auf dem Bildschirm ausgeben. Für ein Rechteck von $(2, 2)$ nach $(5, 3)$ könnte das folgendermaßen aussehen (ich schreibe zur Verdeutlichung `.`, wo Ihr ein Leerzeichen ausgeben sollt):

```
.
.****
.****
```

- den Flächeninhalt des Rechtecks berechnen.

Ihr werdet einen eigenen Konstruktor sowie einige Felder brauchen.

(b) Schreibe ein Hauptprogramm, das zwei Objekte der Klasse `RechteckEcken` anlegt, sie auf den Bildschirm ausgibt und ihre Flächeninhalte anzeigt.

(c) Schreibe eine zweite Klasse `RechteckBreite`, die ein Rechteck durch die Koordinaten seiner linken oberen Ecke sowie seiner Höhe und Breite repräsentiert. Diese Klasse soll die gleichen beiden Methoden unterstützen wie `RechteckEcken`. Der Konstruktor dieser Klasse soll genauso wie der von `RechteckEcken` funktionieren, also das Rechteck über seine Eck-Koordinaten spezifizieren.

(d) Schreibe ein zweites Hauptprogramm, das die Aufgabe in (b) mit Objekten der Klasse `RechteckBreite` erfüllt.

Abgabe bis 17. 5. 2004, 9 Uhr

(java-uebungen@coli.uni-sb.de)