Programmierstarthilfe SS 2009

Fakultät für Ingenieurwissenschaften und Informatik

4. Blatt

Für die Woche vom 18.5. bis zum 22.5.2009 (KW 21)

Organisatorisches

Die Webseiten zur Veranstaltung sind unter `http://www.uni-ulm.de/in/mi/lehre/2009ss/programmierstarthilfe.html` zu finden.

Diese Woche führen wir Methoden ein und behandeln auch die Gültigkeit von Variablen in Programmteilen.

1 Aufgaben

1.1 Wo gilt was?

Schau dir den folgenden Programmcode an und überlege dir (hier wird kein PC benötigt!) wann welche Variable gilt. Welche Ausgaben werden an den vorgesehenen Stellen getätigt. Gibt es Dinge im Programm, die so nicht funktionieren? Warum?

```
1  public class Test {
2      public static String methode(double b, int a) {
3          String c = "Methoden-String";
4          a = 12;
5          System.out.println(a+"\t"+b+"\t"+c);
6          return c+a;
7      }
8      public static void main(String ... args) {
9          {
10             int a = 0; String b = ""; double c = 1.0;
11             System.out.println(a+"\t"+b+"\t"+c);
12         }
13         double a = 2.4;
14         for(int b=0; b<a; b++) {
15             System.out.println(a+"\t"+b+"\t"+c);
16         }
17         System.out.println(a+"\t"+b+"\t"+c);
18         int b = 41;
19         String c = methode(a,b);
20         System.out.println(a+"\t"+b+"\t"+c);
21     }
22 }
```

1.2 Methode für Summierung

a) Schreibe eine Methode, die zwei natürliche Zahlen a und b übergeben bekommt und alle Zahlen von a bis b aufsummiert und die Summe zurückgibt.

b) Erweitere das Programm so, dass immer von der kleineren zur größeren Zahl gezählt wird, unabhängig davon, in welcher Reihenfolge sie als Parameter der Methode übergeben werden.

c) Die Fakultät einer Zahl lässt sich, ähnlich wie unsere Summenfunktion, mit einer For-Schleife Programmieren. Schreibe eine Funktion `fak(int n)`, welche die Fakultät von n berechnet.

1.3 Rechnen mit Methoden

a) Schreibe jeweils eine Methode für Addition, Subtraktion und Multiplikation. Die Methoden sollen als Parameter zwei Zahlen entgegen nehmen, und das Ergebnis der Rechnung zurückgeben.

b) Teste deine Methoden mit einfachen eigenen Rechenbeispielen.

c) Berechne das Ergebnis des Rechenterms (5+3)(6-2) mit Hilfe deiner Methoden. Überlege dir zunächst, wie du schrittweise vorgehen musst.

1.4 Pascalsches Dreieck

Das Pascalsche Dreieck ist das Zahlendreieck, das so beginnt:

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5  10  10   5   1
```

Schreibe ein Programm, dass dieses Dreieck bis zur n-ten Zeile ausgibt. Schreibe dir dazu zuerst eine Methode, welche den Binomialkoeffizienten berechnet. Denn das Dreieck kann man auch mit den Koeffizienten schreiben:

$$\begin{array}{ccccc} & & \binom{0}{0} & & \\ & \binom{1}{0} & & \binom{1}{1} & \\ \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} \end{array}$$

Die Formel für den Binomialkoeffizienten ist

$$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$

Tipp: Du kannst die Methode für die Fakultät aus der Aufgabe *Methode für Summierung* übernehmen.

2 Bonusaufgaben

Durch die Aufgaben oben hast du ein Verständnis für das neue Konzept dieses Blatts bekommen. Durch Bonusaufgaben kannst du nun deine Kenntnisse vertiefen.

2.1 Symmetrie in Arrays

Schreibe eine Methode, die ein Array übergeben bekommt. Die Methode soll prüfen ob das Array symmetrisch gefüllt ist, also der Inhalt des ersten Wertes gleich dem letzten ist, der des zweiten gleich dem vorletzten usw.

a) Schreibe eine Variante für Integer-Werte.

b) Erstelle nun eine Methode mit einem Palindromtest, also mit Char-Werten

c) Können auch Arrays mit Strings verglichen werden? Wenn ja, wie sieht die Methode dazu aus?

2.2 Matrizen-Addition

Implementiere eine Methode `matrixAdd(int m1[][], int m2[][])` derart, dass sie die beiden Eingabematrizen `m1` und `m2` addiert und die Ergebnismatrix zurückgibt.

Eine Matrix wird mit einer anderen Matrix addiert, indem die Matrixelemente einzeln addiert werden. Das heißt: Seien A, B Matrizen mit I Zeilen und J Spalten und a_{ij}, b_{ij} das jeweilige Element in Zeile i und Spalte j. Dann ist $C = A + B$ gegeben durch $c_{ij} = a_{ij} + b_{ij}$ für alle i, j. Eine Matrixaddition ist nur möglich, wenn beide Operanden die selbe Anzahl von Zeilen und Spalten besitzen.

2.3 Stringmatching

Informiere dich über die Methoden `indexOf()` und `lastIndexOf()` der Klasse String über die Dokumentation auf der Seite von Sun:
`http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html`

a) Schreibe eine Methode welche das erste Vorkommen eines Musters in einem Text findet und die Fundstelle zurückgibt. Wird das Muster nicht gefunden, wird -1 zurückgegeben.

   Beispiel: Beim Text `Dies ist eine Furzkissenfabrik!` und dem Muster `Furz` gibt die Methode 14 zurück.

b) Schreibe eine main()-Methode, um die geschriebene Methode zu testen.

c) Modifiziere deine Methode so, dass nicht das erste, sondern das letzte Vorkommen gefunden wird.

*d) Versuche, die Aufgaben zu lösen, ohne `indexOf()` bzw. `lastIndexOf()` zu benutzen, indem du diese Methoden selbst implementierst. Dabei darf aber die Methode `charAt()` benutzt werden.

2.4 *Zählen von Buchstaben in Texten

Schreibe eine Methode, die einen Text und einen Buchstaben als Parameter entgegen nimmt und zurückgibt, wie oft der Buchstabe im Text vorhanden ist. Wenn beispielweise der Text „das ist ein test“ übergeben wird, und alle Vorkommen von 't' gezählt werden, so soll die Methode die Zahl 3 zurückgeben.

a) Wie sieht die sogenannte Methodensignatur aus? Also welche Datentypen werden für die Paramter benötigt und was wird zurückgegeben?

b) Schreibe die Methode in Java und teste sie anhand verschiedener Testfälle. Beachte zur Vereinfachung ausschließlich das kleine Alphabet.

c) Schreibe nun eine zweite Methode, die bei einem übergebenem Text eine Häufigkeitstabelle der Vorkommen der Buchstaben a - z ausgibt. Für die einzelnen Häufigkeiten der Buchstaben kannst du jeweils auf die vorherige Methode zurückgreifen.

2.5 *Quersummen

Eine Quersumme einer Zahl ist die Summe der einzelnen Ziffern.

a) Es soll eine Methode erstellt werden, die die Quersumme einer übergebenen Zahl berechnet und zurückgibt. Es ist allerdings etwas trickreich, die einzelnen Ziffern zu bestimmten. Deswegen hier ein kleiner Tipp: Nehme eine beliebige Zahl und teile sie auf einem Blatt Papier modulo und diviso 10, also eine ganzzahlige Division mit Rest. Welche zwei Zahlen hast du hiermit errechnet? Wie kannst du das nun für einen Algorithmus nutzen?

b) Von einer Quersumme kann man wiederum eine Quersumme bilden. Eine besondere Quersumme ist die einstellige Quersumme. Hier wird solange von einer Quersumme wieder Quersumme gebildet, bis diese einstellig ist (Beispiel: 93: 9+3 = 12: 1+2 =3). Schreibe nun eine Methode, die eine Zahl übergeben bekommt und daraus die einstellige Quersumme berechnest. Hierfür kannst du (mehrmals) auf deine vorherige Methode zurückgreifen, bis du das gewünschte Ergebnis gefunden hast.

2.6 **Matrizen-Multiplikation

Schreibe ein Programm, welches zwei Matrizen miteinander multipliziert[1]. Die zu multiplizierenden Matrizen sollen dabei als Eingabe vom Benutzer gelesen werden. Gehe bei der gesamten Aufgabe davon aus, dass nur Ganzzahlen als Matrizenelemente behandelt werden.

a) Schreibe eine Methode, welche eine Matrix von der Standardeingabe einliest und an die aufrufende Funktion als zweidimensionales Array zurückgibt. Prüfe beim Einlesen der Matrix auch, ob die Eingabe wirklich eine gültige Matrix ist und gib andernfalls eine Fehlermeldung aus.

[1] Matrizen-Multiplikation wird hier anschaulich erklärt: `http://de.wikipedia.org/wiki/Falksches_Schema`

b) Implementiere eine Methode, welche eine Matrix im schönen Format wieder auf dem Bildschirm ausgibt. Teste mit Hilfe dieser Methode, ob deine Eingabemethode korrekt funktioniert.

c) Schreibe eine Methode, die zwei Matrizen als Eingabe bekommt, diese miteinander multipliziert und das Ergebnis zurückgibt. Die Methode sollte auch prüfen ob die beiden übergebenen Matrizen überhaupt miteinander multipliziert werden können.

d) Setze alle bisher erstellten Methoden zu einem kompletten Programm zusammen, welches zuerst zwei Matrizen von der Standardeingabe einliest, diese dann multipliziert und das Ergebnis dann wieder auf dem Bildschirm ausgibt.

2.7 **Nimm-Spiel

Schreibe ein kleines Spiel: Zunächst soll die Anzahl der verwendeten Stäbchen eingegeben werden, dann nehmen der Spieler und der Computer abwechselnd 1 bis 3 Stäbchen weg (wobei der Computer immer die optimale Strategie verfolgt, schreibe dazu eine Methode `rechnerNimmt()`). Verloren hat derjenige, der das letzte Stäbchen wegnimmt. Der Spieler kann anschließend entscheiden, ob er nochmal spielen will.

Überlege dir zunächst den Programmablauf und skizziere ihn durch Kommentare in deinem Quellcode.

3 Für das nächste Blatt

Nächste Woche werden wir Rekursion behandeln. Lies dazu das Kapitel *Rekursion* im Skript.