Programmierstarthilfe SS 2009

Fakultät für Ingenieurwissenschaften und Informatik

1. Blatt

Für die Woche vom 27.4. bis zum 1.5.2009 (KW 18)

Organisatorisches

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

Diese Woche schließen wir die elementaren Datentypen ab. Wir rechnen mit Wahrheitswerten und Strings und machen Typecasts. Dieses Blatt bezieht sich im Skript auf die Kapitel *Datentypen* und *Kontrollstrukturen*.

Bisher haben wir immer feste Werte in unseren Programmen angegeben. Damit wir nun auch Eingaben machen können verwenden wir fertige Routinen, die das für uns erledigen. Dieses kleine Programm liest einen String und einen Integer ein.

```
1 import java.util.*;
2 public class Hello {
3   public static void main(String[] args) {
4     Scanner scanner = new Scanner(System.in);
5     String myString = scanner.nextLine();
6     int myInt = scanner.nextInt();
7   }
8 }
```

1 Aufgaben

1.1 Byte und Short

Java bietet verschiedene Datentypen für numerische Werte an, die kleinsten Datentypen für ganzzahlige Werte sind Byte und Short. Erstelle eine Integer-Variable `a` und weise ihr den Wert 127 zu. Deklariere nun eine Variable `b` von Typ `byte` und eine Variable `c` vom Typ `short`. Weise nun den Variablen `b` und `c` den Wert von `a` zu (beachte hierbei die unterschiedlichen Datentypen). Lasse dir die Variablen nun ausgeben.
Ändere nun dein ursprüngliches Programmm so ab, dass die Integer-Variable den Wert 128 besitzt. Was passiert bei der Ausgabe?

1.2 Genauigkeit von Fließkommazahlen

Berechne folgenden Rechenterm in Java:

$$\frac{b + (a \cdot b + e - c \cdot d) \cdot d + (a - c)}{e + e}$$

Dabei sollen die Variablen mit folgenden Werten belegt werden:

$$a = 300.1 \;\; b = 400.1 \;\; c = 200.1 \;\; d = 600.1 \;\; e = 10.0$$

a) Benutze für alle Variablen den Datentyp `float`.

b) Benutze anschließend den Datentyp `double` für alle Variablen.

Vergleiche die Ergebnisse und erkläre, wie es zu diesem Unterschied kommt!

1.3 Darstellung von reellen Zahlen auf Computern

Erstelle eine Variable von Typ `double` und weise ihr den Wert 0 zu. Addiere nun in drei einzelnen Schritten 0.1 auf diese Zahl und lasse dir anschließend das Ergebnis ausgeben. Wie erklärst du dir die Ausgabe? Was passiert, wenn du direkt 0.3 addierst?

1.4 Plantagenbewohner

Im Herzen einer Java-Plantage leben die vier Stämme der Asis, Belas, Cedis und Drudis. Forschungen ergaben, dass es vier Eigenschaften gibt, die eine Unterscheidung der Stämme erlauben: ein Bewohner der Plantage kann (muss aber nicht) manuseln, einen Knelt haben, löpsen und nopeln.

Man weiss, dass nur die Asis einen Knelt haben und manuseln. Hat jemand keinen Knelt und nopelt, dann ist er gewiss ein Bela. Ein Bewohner mit Knelt, der nicht manuselt, ist ein Cedi, wenn er immer nopelt. Wer keinen Knelt hat und löpst, nie nopelt und stets manuselt, ist mit Bestimmtheit ein Cedi; würde er nicht manuseln, wäre er ein Drudi. Es ist geradezu typisch für Drudis, dass sie weder manuseln noch nopeln, aber einen ordentlichen Knelt haben. Ganz enthaltsame Bewohner, die keinen Knelt haben, nicht löpsen und nicht nopeln, sind Drudis, wenn sie manuseln, und Cedis wenn sie nicht manuseln.

Schreibe ein Programm, dass die vier Eigenschaften eines Plantagenbewohners erfragt und eine Diagnose liefert, zu welchem Stamm dieser gehört.

Das folgende Stück Quelltext verwendet den Scanner von oben und liest eine Eingabe ein und liefert bei `J` oder `j` als Ergebnis `true` und sonst `false`.

```
1 String janein = scanner.nextLine();
2 test = janein.equals("j") || janein.equals("J");
3 // test ist true bei j oder J
```

Tipp: Im Skript steht in *1.2 Wahrheitswerte* wie man mehrere Booleans miteinander verknüpfen kann.

1.5 Dividieren

Wir möchten `0.1`, `0.5` und `2.8` ausgeben. Warum reicht dafür Folgendes nicht aus?

```
1 System.out.println(1/10);
2 System.out.println(5/10);
3 System.out.println(28/10);
```

Wie geht es richtig?

Hinweis: Java kennt zwei Divisionen: Teilt man `ints` durcheinander, so kommt auch wieder eine ganze Zahl heraus, der Rest wird ignoriert. Teilt man Kommazahlen durcheinander, so ist das Ergebnis ebenfalls eine Kommazahl.

1.6 Strings konkatenieren

Strings kann man mit einem Plus-Operator aneinander hängen, auch konkatenieren genannt (erster-String + zweiterString + ...). Auch andere Datentypen lassen sich damit in Strings wandeln. Wahrscheinlich hast du genau das schon mal innerhalb von `System.out.println()` benutzt, denn hier passiert nichts anderes als dass alle durch + getrennten Teile in einen String konvertiert werden, hintereinander gehängt und anschließend ausgegeben werden.

Schreibe ein Programm in dem du Variablen aller Typen, die du bisher kennst, anlegst und speichere deren Konkatenation in einem String, den du anschließend ausgibst. (Hinweis: eventuell muss die Konkatenation mindestens einen String enthalten, dies kann aber auch ein Leerer (`""`) sein.)

2 Bonusaufgaben

Wenn du noch mehr üben willst, kannst du ein paar von diesen Aufgaben lösen.

2.1 Logische Verknüpfungen

Neben den einfachen logischen Operatoren und, oder und nicht gibt es auch weitere, die sich aus diesen zusammensetzen lassen.

- aus a folgt b = NICHT a ODER b
- a NOR b = NICHT (a ODER b)
- a XOR b = (a UND NICHT b) oder (NICHT a und b)
- a XNOR b = NICHT (a XOR b)

Definiere zwei boolesche Variablen, überprüfe die oben genannten Fälle und gib entsprechend Text aus, wenn sie zutreffen (z.B. a XOR b trifft zu). Überprüfe deinen Code durch Ausprobieren verschiedener Belegungen.

Tip: Sieh dir im Skript den Teil *if mit mehr Möglickeiten* aus dem Kapitel *Kontrollstrukturen* an.

2.2 Mehr logische Verknüpfungen

In einem Zimmer gibt es eine Lampe und drei Schalter. Gesucht ist nun ein boolscher Ausdruck, so dass die Lampe leuchtet wenn die Mehrheit der drei Schalter gedrückt ist. Schreibe ein Programm, welches für von dir definierte Variablen ausgibt ob die Lampe leuchtet oder nicht.

2.3 */** Beweis über Wahrheitsbelegung

Für logische Ausdrücke `boolean a, b, c` gelten folgende Gesetze:

$$a \wedge b = b \wedge a$$

$$(a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$$

$$(a \oplus b) \wedge c = a \oplus b \oplus c \quad (*)$$

$$\neg(a \wedge b) = \neg a \vee \neg b$$

$$((\neg a \wedge (a \vee b)) \vee b) \wedge c = (a \wedge c) \vee (b \wedge c) \quad (**)$$

Dabei ist mit $\neg$ das Nicht, mit $\wedge$ das logische Und, mit $\vee$ das logische Oder und mit $\oplus$ das exklusive Oder gemeint.

Diese Gesetze kann man beweisen, indem man alle möglichen Kombinationen von wahren und falschen `a`, `b` und `c` einsetzt und jeweils überprüft, ob die Ergebnisse auf beiden Gleichungsseiten auch gleich sind. Schreibe ein Java Programm, dass alle Fälle ausprobiert und eine Warnung gibt, falls beide Seiten nicht übereinstimmen (dann gilt die Gleichung nämlich nicht).

3 Für das nächste Blatt

Nächste Woche beginnen wir mit Schleifen. Lies dazu im Skript das Kapitel *Schleifen* und *Kommentieren, Einrücken und Formatieren* aus der Einleitung.