Aufgaben Programmiervorkurs
Übungsblatt 2

TECHNISCHE UNIVERSITÄT DARMSTADT

von Uli Fahrer & Dennis Albrecht

Wintersemester 2014/2015

Aufgabe 1 Multiple Choice

Kreuze zu jeder Antwort an, ob sie zutrifft (**w**) oder nicht (**f**).

Aufgabe 1.1 Anweisungen (Stufe 1)

w f

■ □ Eine if-Anweisung beeinflusst den Programmablauf.

□ ■ Eine if-Anweisung benötigt eine else-Klausel.

Antwort

Die erste Aussage trifft zu.

Aufgabe 1.2 Schleifen (Stufe 1)

w f

■ □ Eine for-Schleife lässt sich durch eine while-Schleife ersetzten.

■ □ Eine while-Schleife lässt sich durch eine do-while-Schleife ersetzten.

■ □ Eine do-while-Schleife lässt sich durch eine for-Schleife ersetzten.

□ ■ Der continue-Befehl beendet die Schleifenausführung.

■ □ Jeder der drei Teile des for-Schleifen-Kopfes kann prinzipiell leer gelassen werden.

□ ■ Die while-Schleife wird mindestens 1 mal ausgeführt.

■ □ Die do-while-Schleife wird mindestens 1 mal ausgeführt.

□ ■ switch-case bietet die gleiche Funktionalität wie if-else.

Antwort

Die zutreffenden Antworten sind die Aussagen 1 bis 3, 5 und 7.
Anmerkung zu Antwort 5: Die Abbruchbedingung (der mittlere Teil des Schleifenkopfes) muss nicht angegeben werden und wird falls nicht vorhanden als `true` *angenommen, die Schleife terminiert dann nicht selbstständig (also braucht man dann irgendwo ein* `break`*).*

Aufgabe 2 Theoriefragen

Aufgabe 2.1 switch-case-Anweisung (Stufe 1)

Betrachte ein Beispiel der switch-case-Anweisung (beispielsweise das aus den Folien). Überlege dir zuerst, was passiert, wenn du einen (oder mehrere) der break-Befehle weglässt und probiere es danach selbst aus. Was fällt dir auf (du musst gegebenenfalls mehrere Werte für die Variable ausprobieren)? Erkläre dieses Verhalten.

Antwort

Der break-Befehl beendet die Ausführung des switch-Blockes. Ohne diesen Befehl wird der Block auch über eine case-Grenze hinweg weiter ausgeführt. Dieses Verhalten ist auch leicht zu erklären, da das case-Statement nur eine Ansprungmarke ist und Ansprungmarken bei Programmausführung überlesen wird. In unserem Beispiel führt das Weglassen des break-Befehls zu einem semantischen Fehler, (da die Zahl eins nicht gleichzeitig eins und zwei ist,) aber in anderen Beispielen kannst du dieses Verhalten auch zu deinem Vorteil ausnutzen.

Aufgabe 3 Anweisungen

Aufgabe 3.1 Operatoren (Stufe 1)

Es seien die folgenden Variablen deklariert und initialisiert:
`int x = 6; int y = 7; int z = 0; boolean a = false; boolean b;`
Welchen Wert enthält die Variable b jeweils nach Ausführung der folgenden Anweisungen?

a) `b = x > 5 || y < 7 && z != 0;`

b) `b = x * y != y * x && x / z == 0;`

c) `b = (a = !a) != a;`

Antwort

a) b = true; *b) b = false;* *c) b = false;*

Aufgabe 3.2 Taschenrechner (Stufe 2)

In dieser Aufgabe soll ein sehr einfacher Taschenrechner implementiert werden, wobei wir uns dabei auf die Grundrechenarten beschränken wollen. Dein Programm soll nacheinander drei Eingaben entgegennehmen, zwei Operanden (**ganze Zahlen**) und einen Operator (+, -, *, /). Schließlich soll der Taschenrechner eine Ausgabe liefern mit dem entsprechenden Ergebnis. Vergiss nicht, dein Programm zu testen.

Antwort

```
1  public class Taschenrechner {
2      public static void main(String[] args) {
3          System.out.print("Wie lautet der erste Operand: ");
4          int op1 = Input.readInt();
5          System.out.print("Wie lautet der zweite Operand: ");
6          int op2 = Input.readInt();
7          System.out.print("Wie lautet der Operator: ");
8          char op = Input.readChar();
9          switch(op){
10             case '+':
11                 System.out.println("Das Ergebnis lautet: " + (op1 + op2));
12                 break;
13             case '-':
14                 System.out.println("Das Ergebnis lautet: " + (op1 - op2));
15                 break;
16             case '*':
17                 System.out.println("Das Ergebnis lautet: " + (op1 * op2));
18                 break;
19             case '/':
20                 System.out.println("Das Ergebnis lautet: " + (op1 / op2));
21                 break;
22             default:
23                 System.out.println("Keine gueltige Eingabe!");
24         }
25     }
26 }
```

Aufgabe 4 Schleifen

Aufgabe 4.1 Gaußsche Summenformel Reloaded (Stufe 1)

Erinnert euch an die Gaußsche Summenformel aus Übungsblatt 1:

$$1+2+3+\ldots+n=\sum_{k=1}^{n}k=\frac{n\cdot(n+1)}{2}$$

Nun soll die Summe nicht mit Hilfe der geschlossenen Darstellung der Reihe berechnet werden, sondern unter Verwendung einer **for-Schleife**.

Antwort

```
1 public class Summenformel {
2     public static void main(String[] args) {
3         int sum = 0;
4         for(int i=1; i<=100; i++)
5             sum += i;
6         System.out.println("Die Summe der ersten 100 Zahlen ist " + sum);
7     }
8 }
```

Aufgabe 4.2 Fakultät (Stufe 1)

a) Schreibe ein Programm, das den Wert des Ausdrucks $1*2*3*\ldots*15 = 15!$ (Fakultät von 15) berechnet, und das Ergebnis auf der Konsole ausgibt. Achte auf einen passenden Datentyp für die Variablen.

b) Erweitere dein Programm so, dass es von beliebigen Eingaben in der Konsole die Fakultät berechnet.

Antwort

```
1  public class Fakultaet {
2      public static void main(String[] args) {
3          System.out.print("Welche Fakultaet soll berechnet werden: ");
4          int fak = Input.readInt();
5          long erg = 1;
6          for(int i=1;i<=fak;i++)
7              erg *= i;
8          System.out.println("Die " + fak + ". Fakultaet ist " + erg);
9      }
10 }
```

Aufgabe 4.3 FizzBuzz (Stufe 3)

FizzBuzz ist ein bekanntes Trinkspiel. Schreibe ein Programm, das (zeilenweise) die Zahlen von 1 bis 100 ausgibt, aber für jedes Vielfaches von 3 das Wort **Fizz** und für jedes Vielfaches von 5 das Wort **Buzz** anstelle der Zahl ausgibt. Für Zahlen, die Vielfaches von 3 und 5 sind, soll **FizzBuzz** ausgegeben werden.

Antwort

```
1  public class FizzBuzz {
2      public static void main(String[] args) {
3          for(int i=1; i<=100; i++){
4              if(i%3 == 0 && i%5 == 0)
5                  System.out.println("FizzBuzz");
6              else if(i%3 == 0)
7                  System.out.println("Fizz");
8              else if(i%5 == 0)
9                  System.out.println("Buzz");
10             else
11                 System.out.println(i);
12         }
13     }
14 }
```

Aufgabe 4.4 Primzahl (Stufe 3)

Gegeben ist folgendes Listing:

```
1  public class Primzahl{
2      public static void main(String[] args){
3          int prim = 20;
4          boolean isPrim;
5          //TODO: Implement me
6
7          if(isPrim)
8          //TODO: Implement me
9          else
10         //TODO: Implement me
11     }
12 }
```

a) Ergänze das Programm. Es soll erkennen, ob es sich bei der Variable **prim** um eine Primzahl handelt. Teste dein Programm mit verschiedenen Werten.

b) Erweitere das Programm um eine manuelle Eingabe der Primzahl. Nutze dazu die Input-Klasse.

Antwort

```
1  public class Primzahl {
2      public static void main(String[] args) {
3          System.out.print("Gib␣bitte␣eine␣Zahl␣ein:␣");
4          int prim = Input.readInt();
5          boolean isPrim = true;
6          for(int i=2; i<prim; i++)
7              if(prim % i == 0)
8                  isPrim = false;
9          if(isPrim)
10             System.out.println(prim + "␣ist␣eine␣Primzahl!");
11         else
12             System.out.println(prim + "␣ist␣keine␣Primzahl!");
13     }
14 }
```

Aufgabe 5 Schleifen und Strings

Aufgabe 5.1 String umkehren (Stufe 2)

Für ein Wort $w = a_1 \dots a_n \in \Sigma^*$ wird w^{-1} durch $a_n \dots a_1$ definiert. D.h. w wird rückwärts gelesen. Schreibe ein Programm, dass die Zeichenkette "**ssapS thcam nereimmargorP**" umkehrt und das Ergebnis auf der Konsole ausgibt.

Antwort

```
1 public class StringUmkehrer {
2     public static void main(String[] args) {
3         String string = "ssapS␣thcam␣nereimmargorP";
4         for(int i = string.length()-1; i>=0; i--)
5             System.out.print(string.charAt(i));
6         System.out.println();
7     }
8 }
```

Aufgabe 5.2 Palindrom (Stufe 2)

Ein Palindrom ist eine Zeichenkette, die von vorne und hinten gelesen gleich bleibt.
Beispielsweise: **Anna, Otto, Reittier, Rotor**.
Schreibe ein Programm, das erkennt ob es sich bei der Zeichenkette:

`Ein Neger mit Gazelle zagt im Regen nie.`

um ein Palindrom handelt. Leerstellen, Satzzeichen und Groß-Klein-Schreibung sollen ignoriert werden, daher sollten diese zu Beginn manuell entfernt/korrigiert werden.

Antwort

```
1  public class Palindrom {
2      public static void main(String[] args) {
3          String string = "einnegermitgazellezagtimregennie";
4          boolean palindrom = true;
5          for(int i=0; i<string.length()/2; i++)
6              if(string.charAt(i) != string.charAt(string.length()-1-i))
7                  palindrom = false;
8          if(palindrom)
9              System.out.println(string + "␣ist␣ein␣Palindrom!");
10         else
11             System.out.println(string + "␣ist␣kein␣Palindrom!");
12     }
13 }
```

Aufgabe 5.3 Zeichen zählen (Stufe 2)

Schreibe ein Programm, das ermittelt, wie oft der Buchstabe 'n' in der folgenden Zeichenkette enthalten ist: "**In diesem sinnlosen String kommen viele n vor.**"

Antwort

```
1  public class Buchstabenzaehler {
2      public static void main(String[] args) {
3          String string = "In␣diesem␣sinnlosen␣String␣kommen␣viele␣n␣vor.";
4          int counter = 0;
5          for(int i=0; i<string.length(); i ++)
6              if(string.charAt(i) == 'n')
7                  counter++;
8          System.out.println("In␣dem␣Satz␣\""+ string
9              + "\"␣kommt␣der␣Buchstabe␣n␣" + counter + "␣mal␣vor.");
10     }
11 }
```