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.

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.

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.

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;`

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.

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**.

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.

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.

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.

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.

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.

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.**"