Programmierung 1 (I)

Aufgabenblatt 5 (Kontrollstrukturen)

Prof. Dr.-Ing. Heiko Tapken / Programmier-Team

Wintersemester 2018/19

Testat KW 46

Erreichbar: 18 Punkte, Bestehensgrenze: 14 Punkte

Hochschule Osnabrück
University of Applied Sciences

Aufgabe 1 (`switch`) [2 Punkte]

Wandeln Sie die folgende `if`-Anweisung in eine äquivalente `switch`-Anweisung um:

```
int zahl = IO.readInt();
if (zahl > 3 && zahl < 7) {
if (zahl > 4 && zahl < 10) {
   System.out.println("ja");
} else {
   System.out.println("weiss nicht");
}
} else if (zahl >= -1 && zahl <= 1) {
   System.out.println("nein");
} else {
   System.out.println("vielleicht");
}
```

Aufgabe 2 (einfache Methoden) [5 Punkte]

Implementieren Sie in Java folgende Prozeduren/Funktionen! Achten Sie auf Randfälle und nicht korrekte Parameterübergaben! Überprüfen Sie aber zunächst für jede Funktion, ob sie überhaupt mit den (bisher bekannten) Konzepten in Java implementiert werden kann und wenn nicht, begründen Sie, wieso nicht!

(1) eine Funktion, die testet, ob eine als Parameter übergebene natürliche Zahl eine Fibonacci-Zahl ist oder nicht (Beispiel: f(8) == true),

(2) eine Prozedur, die die Werte zweier als Parameter übergebener double-Variablen vertauscht (Beispiel: double a=2.0; double b=5.9; f(a, b); Ergebnis: a==5.9; b==2.0;),

(3) eine Funktion, die einen übergebenen double-Wert rundet und als positiven int-Wert zurückliefert (Beispiel: f(-2.6) == 3)

(4) eine Funktion, die die nächst kleinere Primzahl einer als Parameter übergebenen natürlichen Zahl (größer als 2) liefert (Beispiel: f(11) == 7),

(5) eine Funktion, die als Parameter einen int-Wert übergeben bekommt und die überprüft, ob die Ziffer 7 in dem int-Wert vorkommt (Beispiel: f(-2578) == true),

(6) eine Funktion, die als ersten Parameter eine Funktion g:char->int und als zweiten Parameter einen char-Wert *zeichen* übergeben bekommt und die als Ergebnis den Wert g(zeichen) liefert (Beispiel: public static int pos(char zeichen) {return zeichen - ‘a‘ ;} und f(pos, ‘b‘) == 2),

(7) eine Funktion, die als Parameter einen int-Wert n übergeben bekommt und die als Ergebnis die Summe der Zahlen zwischen 0 und n zurückliefert; ist der Wert des übergebenen Parameters jedoch kleiner als 0, soll die Funktion den Wert false liefern (Beispiel: f(4) = 10, f(-2) == false)

(8) eine Funktion, die als ersten Parameter einen int-Wert n (Anzahl der zu übergebenen Float-Werte) und daraufhin n float-Parameter übergeben bekommt, deren Summe geliefert werden soll (Beispiel: f(3, 1.1f, 2.2f, 3.3f) == 6.6f).

(9) eine Funktion, die als ersten Parameter einen float-Wert *x* (zwischen 0 und 1000) und als zweiten Parameter einen positiven int-Wert *n* (zwischen 1 und 5) übergeben bekommt. Die Funktion soll den Wert x auf n Nachkommastellen runden (Beispiel: f(2.2576F, 3) == 2.258F)

(10) eine Funktion, die die Summe zweier Uhrzeiten als Ergebnis liefert; Uhrzeiten werden dabei als float-Werte realisiert, wobei die Vorkommastellen die Stunden und die Nachkommastellen die Minuten darstellen (Beispiel: f(22.13f, 3.48f) == 2.01f)

(11) eine Funktion, die die Summe und die Differenz zweier als Parameter übergebener int-Werte zurückliefert (Beispiel: f(4, 3) = (7, 1))

Schreiben Sie ein Programm, das einem Benutzer eine Auswahl zur Ausführung der implementierbaren Funktionen anbietet, anschließend jeweils passende Werte für die aktuellen Parameter einliest, die ausgewählte Funktion aufruft und ein Ergebnis auf dem Bildschirm ausgibt. Achten Sie darauf, dass die Funktionen nur mit zulässigen Werten aufgerufen werden.

Aufgabe 3 [4 Punkte]

Ein im Studiengang Informatik – Medieninformatik sehr beliebtes Thema der Mathematikvorlesung ist die Berechnung des Pascalschen Dreiecks. Das P. Dreieck berechnet sich wie folgt:

Beispiel für n=4:

$$\begin{array}{ccccccccc}
 & & & & \binom{0}{0} & & & & \\
 & & & \binom{1}{0} & & \binom{1}{1} & & & \\
 & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & \\
 & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & \\
\binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4}
\end{array}$$

a) Schreiben Sie eine Methode, dass die Berechnung beliebiger Binomialkoeffizienten ermöglicht. Identifizieren Sie geeignete Parameter.
b) Erweitern Sie Ihr Programm, so dass die Berechnung und Ausgabe Pascalscher Dreiecke ermöglicht wird. Dabei sollten Sie geeignete Methoden identifizieren und implementieren.

c) Steuern Sie Ihr Programm durch ein geeignetes Text-Menü.

Nähere Infos finden Sie unter: http://de.wikipedia.org/wiki/Pascalsches_Dreieck

Fortgeschrittene: Der Binomialkoeffizient für n+1 über k+1 lässt sich auch rekursiv wie folgt berechnen.

$$\binom{n}{n} = \binom{n}{0} = 1; \quad \binom{n+1}{k+1} = \binom{n}{k} + \binom{n}{k+1}.$$

Nutzen diesen Ansatz zu Lösung des Problems.

Aufgabe 4 (Zocker-Automat) [4 Punkte]

Jetzt wird gezockt. Wir programmieren unser zweites Spiel:

Spielablauf:
Sie haben einen initialen Geldbetrag von 1000. Nun können Sie eine Summe auf eine gerade oder eine ungerade Summe setzen. Drücken Sie auf Weiter. Ihre Eingaben werden nun geprüft. Klicken Sie nun auf das Bild; es wird gewürfelt und Ihr aktuelles Guthaben wird aktualisiert. Nun geht das Ganze von vorne los ;-)

Machen Sie sich mit dem Spiel vertraut. In OSCA finden Sie eine Datei Wuerfel.zip. Entpacken Sie diese Datei und starten Sie das Spiel in der Eingabeaufforderung mit `java -jar Wuerfel.jar`. Spielen Sie das Spiel solange, bis Sie es verstanden haben (oder Sie keine Lust mehr haben ;-)).

Nun zur eigentlichen Aufgabe:
Leider hat der Programmierer nach Erstellen der Jar-Datei einen Festplattenausfall zu beklagen gehabt. Es existiert aber noch das letzte erstellte Backup. Dieses finden Sie in OSCA.

Laden Sie Datei WuerfelWetteMitGUI.zip aus dem Osca herunter. Dann importieren Sie das Projekt durch File -> Import … -> General -> Existing Projects into Workspace -> Browse … -> Auswählen des Root-Directories des Projekts (WuerfelWetteMitGUI).

Zum Glück hat der Programmierer zumindest dahingehend gute Arbeit geleistet, indem er den Code aussagekräftig kommentiert hat. Es sind nun einige Schritte notwendig, um das Programm wieder korrekt lauffähig zu machen. Die entsprechenden Stellen hat der Programmierer mit dem Kommentar `//todo` eingeleitet.

Programmierung 1 (I)

Aufgabenblatt 5 (Kontrollstrukturen)

Prof. Dr.-Ing. Heiko Tapken / Programmier-Team

Wintersemester 2018/19

Testat KW 46

Erreichbar: 18 Punkte, Bestehensgrenze: 14 Punkte

Hochschule Osnabrück
University of Applied Sciences

Programmieren Sie die in den Kommentaren beschriebenen Funktionalitäten in den Klassen Spiel.java und Wuerfel.java.

Ihr Programm können Sie starten durch Rechtsklick auf Main.java -> Run as -> Java Application.

Hinweis: Die Erweiterung und Korrektur eines von anderen Programmierern geschriebenen Programms ist eine Standardaufgabe im Berufsleben von MedieninformatikerInnen.

Aufgabe 5 (eindimensionale Arrays) [3 Punkte]

Bei dieser Aufgabe sollen Sie einen einfachen Fahrtroutenplaner implementieren. Stellen Sie sich dazu ein Netz von Städten vor, das durch Straßen miteinander verbunden ist. Schreiben Sie ein Programm, was genau Folgendes tut:

- Zunächst wird die Anzahl *anzahl* an Städten eingelesen.
- Anschließend werden die Namen von *anzahl* Städten eingelesen.
- Danach wird die Anzahl *direkt* der Direktverbindungen zwischen einzelnen Städten eingelesen (Verbindungen sind bidirektional)
- Anschließend werden die *direkt* Direktverbindungen eingelesen, und zwar in der Form Ausgangsstadt, Zielstadt.
- Danach soll das Programm in einer Endlosschleife als Auskunftssystem dienen. Für jede Auskunft sollen jeweils zwei Städtenamen eingelesen werden. Das Programm berechnet dann, ob eine Verbindung (Minimal direkte, gern auch indirekte!) zwischen den beiden Städten existiert.

Achten Sie bei den Nutzereingaben auf mögliche Fehler!

Beispiel:

```
Vorbereitung:
Anzahl Städte6
Stadt 1
Bremen
Stadt 2
Osnabrück
Stadt 3
Münster
Stadt 4
Bielefeld
Stadt 5
München
Stadt 6
Heiligenhafen
Anzahl Direktverbindungen4
Direktverbindung 1:
StartBremen
ZielOsnabrück
Direktverbindung 2:
StartBiele
StartBielefeld
ZielMünchen
Direktverbindung 3:
StartMünchen
StartHeiligenhafen
```

Programmierung 1 (I)

Aufgabenblatt 5 (Kontrollstrukturen)

Prof. Dr.-Ing. Heiko Tapken / Programmier-Team

Wintersemester 2018/19

Testat KW 46

Erreichbar: 18 Punkte, Bestehensgrenze: 14 Punkte

Hochschule Osnabrück
University of Applied Sciences

```
ZielMünchen
Direktverbindung 4:
Start Bremen
ZielBielefeld
Auskunft
StartBremen
ZielMünchen
Zwischen Bremen und München ist eine Verbindung nicht vorhanden
StartBremen
ZielOsnabrück
Zwischen Bremen und Osnabrück ist eine Verbindung vorhanden
```

Aufgabe 6 (Array, Achtung: schwer, OPTIONAL für fortgeschrittene Programmierer!!!) [6 Punkte]

Sie kennen sicher das Spiel *Sudoku*: Das Spiel besteht aus einem Gitterfeld mit 3 × 3 Blöcken, die jeweils in 3 × 3 Felder unterteilt sind, insgesamt also 81 Felder in 9 Reihen und 9 Spalten. In einige dieser Felder sind schon zu Beginn Ziffern zwischen 1 und 9 eingetragen. Typischerweise sind 22 bis 36 Felder von 81 möglichen vorgegeben. Ziel des Spiels ist es nun, die leeren Felder des Puzzles so zu vervollständigen, dass in jeder der je neun Zeilen, Spalten und Blöcke jede Ziffer von 1 bis 9 genau einmal auftritt.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

1. Schreiben Sie ein Programm, das ein vorgegebenes Sudoku löst und die Lösung auf den Bildschirm ausgibt. Sie können dabei das gegebene Sudoku als fest kodierte Matrix im Sourcecode repräsentieren.
2. Schreiben Sie ein Programm, das ein neues Sudoku erstellt.