Fakultät IV
NI & CV

Probeklausur – Java

Einführung in die Informatik

Wintersemester 2014/2015

Musterlösung

Hinweis: Diese Probeklausur ist eine kleine Aufgabensammlung, die etwa dem Schwierigkeitsgrad der schriftlichen Prüfung des Moduls Einführung in die Informatik entspricht. Die hier zur Verfügung gestellten Aufgaben decken jedoch nicht alle behandelten Themenbereiche ab. Darüber hinaus wird es in der Klausur Wissensaufgaben geben, welche aus allen Themen und sämtlichen Teilen der Veranstaltung (Vorlesung, Tutorium und Hausaufgaben) kommen können. In den Klausuren für dieses Semester wird die Gewichtung sein: 40 % Rechneraufbau und 60 % Programmieren

Aufgabe 1 (Zahlensysteme).

1. Rechnen Sie die Zahl $423_{(6)}$ in eine Zahl zur Basis 16 um. Der Lösungsweg muss erkennbar sein.

   Lösung:

   $$423_{(6)} = 4 \cdot 6^2 + 2 \cdot 6^1 + 3 \cdot 6^0 = 144 + 12 + 3 = 159_{(10)}$$
   $$159 : 16 = 9 \text{ Rest } 15 \Rightarrow 423_{(6)} = 9F_{(16)}$$

2. Berechnen Sie die folgenden Aufgaben unter Verwendung der Zweierkomplementdarstellung mit 4 Bit. Der Lösungsweg muss erkennbar sein. Bei welchen Aufgaben findet ein Über- bzw. Unterlauf statt? Woran erkennen Sie jeweils den Über- bzw. Unterlauf?

   (a) $6 + (-2)$

   (b) $-7 + (-2)$

   (c) $4 + 7$

   Lösung:

   (a) $6_{(10)} = 0110_{(2)}$
   $2_{(10)} = 0010_{(2)} \Rightarrow -2_{(10)} = 1110_{(2)}$

	0	0	1	1	0
+	1_1	1_1	1_1	1	0
	0	0	1	0	0

   Ergebnis: $6 + (-2) = 4$: Korrekt, da die ersten beiden Bits im (erweiterten) Ergebnis gleich sind.

   (b) $7_{(10)} = 0111_{(2)} \Rightarrow -7_{(10)} = 1001_{(2)}$
   $2_{(10)} = 0010_{(2)} \Rightarrow -2_{(10)} = 1110_{(2)}$

	1	1	0	0	1
+	1_1	1	1	1	0
	1	0	1	1	1

   Ergebnis: $-7 + (-2) = 7$: Unterlauf, da die ersten beiden Bits im (erweiterten) Ergebnis 10 sind.

   (c) $4_{(10)} = 0100_{(2)}$
   $7_{(10)} = 0111_{(2)}$

	0	0	1	0	0
+	0	0_1	1	1	1
	0	1	0	1	1

   Ergebnis: $4 + 7 = -5$: Überlauf, da die ersten beiden Bits im (erweiterten) Ergebnis 01 sind.

Aufgabe 2 (Fließkommazahlen).

1. Addieren Sie die Zahlen 45_{10} und 0.0625_{10} im angegebenen Gleitkommaformat und konvertieren Sie das Ergebnis zurück in eine Dezimalzahl.

Vorzeichen	Exponent	Mantisse
1 Bit	4 Bits	5 Bits

Der Exzess beträgt: $7_{10} = 0111_2$

Lösung:

(a) Dualzahlen: $45_{10} = 101101_2$, $0.0625_{10} = 0.0001_2$
(b) Normalisierung: $1.01101 \cdot 10_2^{101_2}$, $1.00000000_2 \cdot 10_2^{-100_2}$
(c) Exponent in Exzessdarstellung: $0101 + 0111 = 1100$, $-0100 + 0111 = 0011$)

Vorzeichen	Exponent	Mantisse
0	1100	01101
0	0011	00000

(d) Addition:
- kleineren Exponenten anpassen: $1.00000 \cdot 10_2^{-100_2} = 0.00000 \cdot 10_2^{101_2}$
- Addition der Mantissen

	1.01101
+	0.00000
	1.01101

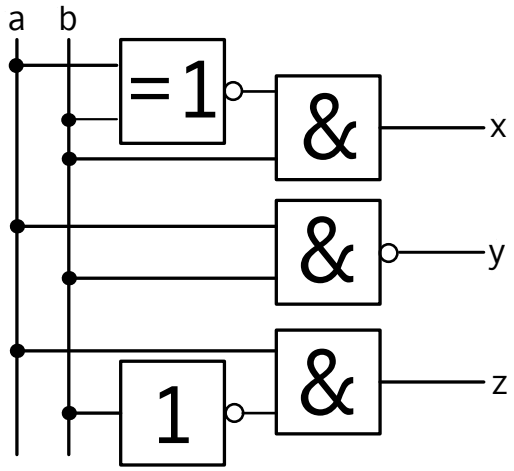
- Zusammenfügen: 0 1100 01101

2. Wie groß ist der Rundungsfehler, der bei der Addition in der ersten Teilaufgabe entsteht. Geben Sie den Rundungsfehler als Dezimalzahl an.

Lösung:
Die Differenz der Exponenten ist 9_{10}, daher fällt das Bit heraus und das Ergebnis ist wieder 45_{10}. Der Fehler beträgt demzufolge 0.0625_{10}.

Aufgabe 3 (Schaltungen).

1. Stellen Sie die Wertetabelle für folgende Schaltung auf:



(a und b sind die Eingänge, x, y und z sind die Ausgänge der Schaltung.)

Lösung:
Zunächst werden die Zwischenpunkte $m := \overline{a \oplus b}$ und $n := \overline{b}$ eingeführt. Dann ergibt sich folgende Wertetabelle:

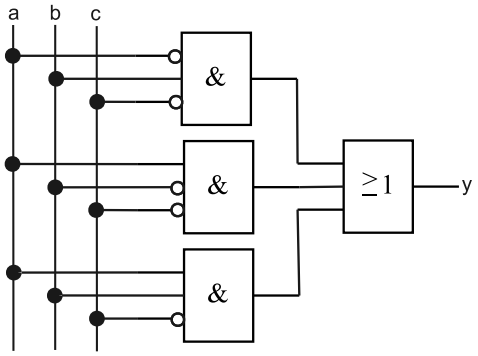
a	b	m	n	x	y	z
0	0	1	1	0	1	0
0	1	0	0	0	1	0
1	0	0	1	0	1	1
1	1	1	0	1	0	0

2. Zeichnen Sie eine Schaltung, die der folgenden Wertetabelle entspricht.

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

(a, b und c sind die Eingänge, y ist der Ausgang der Schaltung.)

Lösung:



Aufgabe 4 (Bedingte Anweisungen).

1. Betrachten Sie die folgende Java-Methode

```
boolean f(boolean x, boolean y) {
    if(x) {
        return y;
    } else if(y) {
        return x;
    } else {
        return x;
    }
}
```

(a) Stellen Sie eine Tabelle auf, die für alle möglichen Wertepaare (x, y) der Parameter den zugehörigen Rückgabewert der Methode `f()` angibt.

Lösung:

x	y	f(x, y)
true	true	true
true	false	false
false	true	false
false	false	false

(b) Implementieren Sie eine möglichst einfache Methode `boolean f2(boolean x, boolean y)`, die ohne `if`-Anweisung auskommt und für alle möglichen Wertekombinationen von `x` und `y` die gleichen Werte wie die Methode `f()` zurückliefert.

Lösung:

```
boolean f2(boolean x, boolean y) {
    return x && y;
}
```

2. Betrachten Sie das folgende Punktesystem zur Notenvergabe:

Punkte	Note
36-40	1
31-35	2
26-30	3
21-25	4
0-20	5

Implementieren Sie eine Methode `int getNote(int punkte)`, die als Wert die Note der übergebenen Punkte zurückliefert. Liefern Sie den Wert `-1` zurück, falls der Parameter `punkte` keine gültige Punktezahl darstellt.

Lösung:
siehe nächste Seite.

```
int getNote(int punkte) {
    if (punkte > 40 || punkte < 0){
        return -1;
    } else if (punkte > 35) {
        return 1;
    } else if (punkte > 30) {
        return 2;
    } else if (punkte > 25) {
        return 3;
    } else if (punkte > 20) {
        return 4;
    } else {
        return 5;
    }
 }
```

Aufgabe 5 (Schleifen).

1. Welchen Wert hat `k` nach Ausführung der folgenden Anweisungen?

```
int k = 2;
for(int i = 2; i < 7; i += 2) {
    k *= 2;
}
```

Lösung:

`k = 16`

2. Wandeln Sie die folgende `for`-Anweisung in eine `while`-Anweisung um.

```
for(int i = 533; i > -12; i -= 7) {
    System.out.println(i);
}
```

Lösung:

```
int i = 533;
while(i > -12) {
    System.out.println(i);
    i -= 7;
}
```

3. Implementieren Sie eine Methode `int zaehleNullen(int[] vektor)`, die die Anzahl der Nullen in dem übergebenen Ganzzahlen-Vektor zurückgibt.

Lösung:

```
int zaehleNullen(int[] vektor) {
    int anzahlNullen = 0;
    for (int i=0; i<vektor.length; i++)
    {
        if (vektor[i] == 0)
            anzahlNullen++;
    }
    return anzahlNullen;
}
```

Aufgabe 6 (Rekursion).

Betrachten Sie die rekursiv definierte Funktion für alle $n \geq 3$

$$\begin{aligned} f(3) &= 1 \\ f(4) &= 2 \\ f(n) &= 2 \cdot f(n-1) + 3 \cdot f(n-2) \quad \text{für } n > 4 \end{aligned}$$

Implementieren Sie eine rekursive Java-Methode, die den Wert $f(n)$ zurückliefert. Die Variable n wird hierbei als Parameter der Methode übergeben. Die Deklaration von lokalen Variablen und Verwendung von Schleifen ist nicht erlaubt.

Lösung:

```
int f(int n) {
    if(n <= 3) {
        return 1;
    } else if(n == 4) {
        return 2;
    } else {
        return 2*f(n-1) + 3*f(n-2);
    }
}
```

Aufgabe 7 (Klassen und Objekte).

1. Ergänzen Sie die folgende Java-Klasse wie in den Kommentaren beschrieben.

   Lösung:

```
public class C {

    private int a;

    public C(int a) {
        // initialisiere Attribut mit uebergebenem Parameter
        this.a = a;
    }

    public void f() {
        a *= 2;
        System.out.println("a = " + a);
    }

    public static void main(String[] args) {
        // Erzeuge ein Objekt der Klasse C. Dabei soll der Wert des
        // Attributs a mit 5 initialisiert werden.
        C c = new C(5);

        // Rufe die Methode f() auf.
        c.f();
    }
}
```

2. Implementieren Sie eine Klasse `Fahrrad`. Die Klasse soll folgenden Anforderungen genügen:

   (a) Der Zustand eines Objekts der Klasse `Fahrrad` wird durch seine Geschwindigkeit und die Anzahl der Gänge beschrieben. Beide Attribute sind vom Typ `int`. Sowohl die Geschwindigkeit als auch die Anzahl der Gänge dürfen nicht negativ werden. Ein Zugriff auf die Attribute außerhalb der Klasse ist nicht möglich.

   (b) Die Klasse besitzt einen parameterlosen Konstruktor, der die Geschwindigkeit mit 0 und die Anzahl der Gänge mit 1 initialisiert.

   (c) die Klasse besitzt einen erweiterten Konstruktor mit zwei Parametern zur Initialisierung der Attribute.

   (d) Die Klasse besitzt eine Methode zum Ändern der Geschwindigkeit. Die Differenz zur alten Geschwindigkeit wird als Parameter der Methode übergeben.

   (e) Die Klasse besitzt eine Methode zum Setzen der Anzahl der Gänge. Die Anzahl wird dabei als Parameter der Methode übergeben.

   (f) Auf die Klasse, die Konstruktoren und auf die Methoden kann von jeder anderen Klasse zugegriffen werden.

   Lösung:
   siehe nächste Seite

```
0   public class Fahrrad {
1
2       private int speed;
3       private int gears;
4
5       public Fahrrad() {
6           this(0, 1); // ruft den parametrisierten Konstruktor auf
7       }
8
9       public Fahrrad(int speed, int gears) {
10          this.speed = Math.max(0, speed);
11          this.gears = Math.max(0, gears);
12      }
13
14      public void changeSpeed(int value) {
15          this.speed += value;
16          this.speed = Math.max(0, speed);
17      }
18      public void setGears(int value) {
19          this.gears = value;
20          this.gears = Math.max(0, gears);
21      }
22  }
```

Aufgabe 8 (Vererbung).

Betrachten Sie die Klassenhierarchie in der folgenden Abbildung.

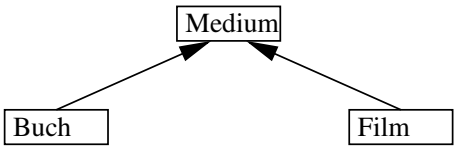


Abbildung 1: Klassenhierarchie

Beachten Sie außerdem: Bücher besitzen einen Titel und einen Autor. Filme werden durch ihren Titel und ihre Spieldauer beschrieben. Implementieren Sie die Klassenhierarchie der obigen Abbildung. Sämtliche Attribute der zu implementierenden Klassen sollen `private` sein, während Konstruktoren und Methoden `public` sein sollen. Gehen Sie folgendermaßen vor:

1. Implementieren Sie eine Klasse `Medium`. Die Klasse `Medium` besitzt als Attribut einen Titel vom Typ `String`. Implementieren Sie einen Konstruktor zur Initialisierung des Titels mit einem Parameter. Definieren Sie eine Methode `void ausgeben()`, welche den Titel auf der Konsole ausgibt.

2. Implementieren Sie eine Unterklasse `Buch` der Klasse `Medium`. Die Klasse `Buch` besitzt als zusätzliches Attribut einen Autor vom Typ `String`. Implementieren Sie einen Konstruktor zur Initialisierung von Titel und Autor mit Parametern. Definieren Sie eine Methode `void ausgeben()`, welche den Titel sowie den Autor auf der Konsole ausgibt.

3. Implementieren Sie eine Unterklasse `Film` der Klasse `Medium`. Die Klasse `Film` besitzt als zusätzliches Attribut die Spieldauer vom Typ `int`. Implementieren Sie einen Konstruktor mit Parametern zur Initialisierung von Titel und Spieldauer. Definieren Sie eine Methode `void ausgeben()`, welche den Titel sowie die Spieldauer auf der Konsole ausgibt.

   Lösung:

```
0   public class Medium {
1
2       private String titel;
3
4       public Medium(String titel) {
5           this.titel = titel;
6       }
7
8       public void ausgeben(){
9           System.out.println("Titel : "+titel);
10      };
11  }
```

```
0   public class Buch extends Medium {
1
2       private String autor;
3
4       public Buch(String titel, String autor) {
5           super(titel);
6           this.autor = autor;
7       }
8
9       public void ausgeben() {
10          super.ausgeben();
11          System.out.println("Autor     : " + autor);
12      }
13  }
```

```
0   public class Film extends Medium {
1
2       private int dauer;
3
4       public Film(String titel, int dauer) {
5           super(titel);
6           this.dauer = dauer;
7       }
8
9       public void ausgeben() {
10          super.ausgeben();
11          System.out.println("Spieldauer: " + dauer);
12      }
13  }
```