

Vorname

Nachname

Matrikel-Nr

Diese Klausur ist mein **dritter Prüfungsversuch** (bitte ankreuzen): Ja ☐ Nein ☐

Ich studiere im **Diplom- bzw. Bachelorstudiengang** Medien: Diplom ☐ Bachelor ☐

Schreiben Sie jede Lösung auf die Vorderseite eines *neuen Blattes* (und lassen Sie die Rückseiten Ihrer Lösungsblätter *leer*). Die Aufgaben 5 und 6 stehen auf der Rückseite dieses Blattes!

(15/12 Punkte) bedeutet: 15 Punkte für Studiengang **MB** (Bachelor), 12 für **MD** (Diplom).

Aufgabe 1 (15/12 Punkte): Betrachten Sie die folgenden drei Klassenvereinbarungen:

```

1   class K1 {
2       private int wert;
3       public int getWert() {return wert;}
4       public void setWert(int wert) {
5           if (wert >= 0) this.wert = wert;
6       }
7       public K1(int wert) {setWert(wert);}
8   } // class K1
9
10  class K2 {
11      K1 anna = new K1(10);
12      K1 bert = new K1(20);
13      K1 carl = new K1(30);
14  } // class K2
15
16  class K3 {
17      static public void main(String[] sonja) {
18          K1 dora = new K1(15);
19          K1 emil = new K1(25);
20          K2 fred = new K2();
21          ...
22      } // class main
23
24      static void pln(Object ob) {System.out.println(ob);}
25  } // class K3

```

1.1. Analysieren Sie die Klassen und füllen Sie dann die folgende Tabelle aus. Tragen Sie jeweils nur die (einfachen) *Namen* der betreffenden Elemente in die Kästchen der Tabelle ein:

	class K1	class K2	class K3
Klassenattribute			
Klassenmethoden			
Objektattribute			
Objektmethoden			

1.2. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 19 ausgeführt hat. *Wieviele* Module existieren in diesem Moment und wie *heißen* diese Module?

1.3. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 20 ausgeführt hat. *Wieviele* Module existieren in diesem Moment und wie *heißen* diese Module? Geben Sie die Namen an, mit denen man in der main-Methode auf die Module zugreifen kann (falls nötig also Namen der Form mmmm.eeee).

1.4. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 20 ausgeführt hat (wie bei der vorigen Teilaufgabe). *Wieviele* int-Variablen existieren in diesem Moment?

Aufgabe 2 (15/12 Punkte) In der main-Methode der Klasse K3 der vorigen Aufgabe werden drei Variablen namens dora, emil und fred vereinbart. Stellen Sie diese drei Variablen als Bojen dar. Benützen Sie dabei die *ausführliche* Bojendarstellung (nicht die vereinfachte Darstellung). Wenn Sie Referenzen "erfinden müssen", dann nehmen Sie dafür die Zahlen 110, 120, 130, 140 ... in dieser Reihenfolge.

Aufgabe 3 (20/20 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```

1      static String als2er(char zahl) {
2          // Liefert einen String der Laenge 16, der die zahl als 2-er-Zahl
3          // (Binaerzahl) enthaelt. Achtung: Die Komponenten des Ergebnis-
4          // strings sollen Zeichen ('0' bzw. '1') sein, nicht die
5          // Zahlen 0 bzw. 1!
6          //
7          // Beispiele:
8          // als2er('\u0000') ist gleich "0000000000000000"
9          // als2er('\uFFFF') ist gleich "1111111111111111"
10         // als2er('\u0041') ist gleich "0000000001000001"
11         // als2er('A')      ist gleich "0000000001000001"
12         ...
13     } // als2er

```

Aufgabe 4 (20/20 Punkte): Schreiben Sie ein Unterprogramm entspr. der folgenden Spezifikation:

```

1      static int anzAB(String[][] srr) {
2          // Wie oft kommt die Zeichenfolge "AB" in den elementaren
3          // Komponenten der Reihung srr ("in den String-Komponenten
4          // von srr") vor? Diese Funktion liefert die Antwort.
5          //
6          // Beispiele: Seien folgende Reihungen vereinbart:
7          // String[][] srr01 = {{ "ABCAB", "XABX", "AB"}, { "XX", "A", "" }, { }};
8          // String[][] srr02 = {{ "ABABAB", "ABABABA", "AABABABX", "AA" }};
9          // String[][] srr03 = { };
10         //
11         // Dann ist anzAB(srr01) gleich 4
12         //           anzAB(srr02) gleich 9
13         //           anzAB(srr03) gleich 0
14         ...
15     } // anzAB

```

Aufgabe 5 (15/13 Punkte): Betrachten Sie die folgenden for-Schleifen:

```
1      // Teilaufgabe 5.1.
2      for (int i=0; i<5; i++) {
3          for (int j=0; j<i; j++) p("XX");
4          for (int j=i; j<5; j++) p("OO");
5          pln();
6      }
7
8      // Teilaufgabe 5.2.
9      for (int i=5; i>0; i--) {
10         for (int j=0; j<i; j++) p("XX");
11         for (int j=i; j<5; j++) p("OO");
12         pln();
13     }
14
15     // Teilaufgabe 5.3.
16     for (int i=1; i<=5; i++) {
17         for (int j=5; j>i; j--) p("XX");
18         for (int j=i; j>0; j--) p("OO");
19         pln();
20     }
```

Geben Sie für jede Teilaufgabe an, was durch sie zum Bildschirm ausgegeben wird. Die Prozedur `p` gibt (wie in zahlreichen Beispielprogrammen) ihren `String`-Parameter zum Bildschirm aus, *ohne* den Bildschirmzeiger (engl. cursor) zum Anfang der folgenden Zeile vorzuschieben. Die parameterlose Prozedur `pln` schiebt den Bildschirmzeiger zum Anfang der nächsten Zeile vor.

Aufgabe 6: (15/13 Punkte)

- 6.1. Erläutern Sie kurz den Unterschied zwischen einem *Sammlungsobjekt* (engl. collection object) und einem *Behälterobjekt* (engl. container object).
- 6.2. Geben Sie eine *Art* und eine *Oberart* von Ereignissen an, die in einem Java-Programm im Zusammenhang mit einem *Grabo-Objekt* (oder: einem *Component-Objekt*) auftreten können.
- 6.3. Geben Sie mit dem Befehl `pln` ein *anonymes Objekt* aus. Den Typ des Objekts dürfen Sie völlig frei wählen.
- 6.4. Vereinbaren Sie eine Variable namens `berthold`, die zu irgendeinem *anonymen Typ* gehört.

Lösung 1 (15/12 Punkte):

1.1. Analysieren Sie die Klassen und füllen Sie dann die folgende Tabelle aus. Tragen Sie jeweils nur die (einfachen) *Namen* der betreffenden Elemente in die Kästchen der Tabelle ein:

	class K1	class K2	class K3
Klassenattribute	--	--	--
Klassenmethoden	--	--	main, pln
Objektattribute	wert	anna, bert, carl	--
Objektmethoden	getWert, setWert	--	--

1.2. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 19 ausgeführt hat. *Wieviele* Module existieren in diesem Moment und wie *heißen* diese Module?

4 Module, 2 Klassen namens K1 und K3 und

2 Objekte namens dora, emil.

1.3. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 20 ausgeführt hat. *Wieviele* Module existieren in diesem Moment und wie *heißen* diese Module? Geben Sie die Namen an, mit denen man in der main-Methode auf die Module zugreifen kann (falls nötig also Namen der Form mmmm.eeee).

9 Module, 3 Klassen namens K1, K2 und K3 und

6 Objekte namens dora, emil, fred, fred.anna, fred.bert, fred.carl.

1.4. Stellen Sie sich vor, dass der Ausführer die main-Methode von K3 bis einschließlich Zeile 20 ausgeführt hat. *Wieviele* int-Variablen existieren in diesem Moment?

5

Lösung 2 (15/12 Punkte): Die Variablen dora, emil und fred als Bojen dargestellt:

```

1  |dora|--<110>--[<120>]--[ wert <130>--[15] ]
2
3  |emil|--<140>--[<150>]--[ wert <160>--[25] ]
4
5
6      +-----+
7      |  anna   bert   carl  |
8  |fred|--<170>--[<180>]--+<190>  <220>  <250>
9      |         |         |
10     |<200>| <230>| <260>|
11     |         |         |
12     +-----+
13
14     +---+---+ +---+---+ +---+---+
15     |wert| |wert| |wert|
16     |<210>| |<240>| |<270>|
17     |         |         |
18     |[10]| |[20]| |[30]|
19     +---+---+ +---+---+ +---+---+
20

```

Lösung 3 (20/20 Punkte):

```

1      static String als2er(char zahl) {
2          // Liefert einen String der Laenge 16, der die zahl als 2-er-Zahl
3          // (Binaerzahl) enthaelt. Achtung: Die Komponenten des Ergebnisstrings
4          // sollen Zeichen sein ('0' bzw. '1'), nicht die Zahlen 0 bzw. 1!
5
6          StringBuilder sb = new StringBuilder();
7          char          ziff;
8
9          for (int i=0; i<16; i++) {
10             ziff = (char) (zahl%2 + '0');
11             zahl = (char) (zahl/2);
12             sb.insert(0, ziff);
13         }
14         return sb.toString();
15     } // als2er

```

Lösung 4 (20/20 Punkte):

```

1      static int anzAB(String[][] srr) {
2          // Wie oft kommt die Zeichenfolge "AB" in den elementaren
3          // Komponenten der Reihung srr ("in den String-Komponenten
4          // von srr") vor? Diese Funktion liefert die Antwort.
5
6          int anz = 0;
7
8          for (String[] sr : srr) {
9              for (String s : sr) {
10                 for (int i=0; i<s.length()-1; i++) {
11                     char c1 = s.charAt(i);
12                     char c2 = s.charAt(i+1);
13                     if (c1 == 'A' && c2 == 'B') {
14                         anz++;
15                         i++; // effizient, aber nicht unbedingt noetig
16                     }
17                 }
18             }
19         }
20         return anz;
21     } // anzAB

```

Lösung 5 (15/13 Punkte): Die Ausgaben der drei Teilaufgaben (for-Schleifen):

```

1      // Teilaufgabe 5.1.
2      0000000000
3      XX00000000
4      XXXX000000
5      XXXXXX0000
6      XXXXXXXX00
7
1     // Teilaufgabe 5.2.
2     XXXXXXXXXX
3     XXXXXXXX00
4     XXXXXX0000
5     XXXX000000
6     XX00000000
7
1     // Teilaufgabe 5.3.
2     XXXXXXXX00
3     XXXXXX0000
4     XXXX000000
5     XX00000000
6     0000000000

```

Lösung 6 (15/13 Punkte):

6.1. Erläutern Sie kurz den Unterschied zwischen einem *Sammlungsobjekt* (engl. collection object) und einem *Behälterobjekt* (engl. container object).

In einem Sammlungsobjekt darf man beliebige Objekte (des Komponententyps) sammeln und ein Objekt kann gleichzeitig zu mehreren Sammlungsobjekten gehören.

In ein Behälterobjekt darf man nur Grabo-Objekte (Component-Objekte, GUI objects) hineintun und ein Grabo-Objekt kann in jedem Moment zu höchstens einem Behälterobjekt gehören.

6.2. Geben Sie eine *Art* und eine *Oberart* von Ereignissen an, die in einem Java-Programm im Zusammenhang mit einem Grabo-Objekt (oder: einem Component-Objekt) auftreten können.

Arten von Ereignissen: windowClosing, windowOpened, ..., mouseClicked, mouseMoved, ...

Oberarten: Fensterereignisse, Mausereignisse, Mausbewegungsergebnisse, ...

6.3. Geben Sie mit dem Befehl `println` ein *anonymes Objekt* aus. Den Typ des Objekts dürfen Sie völlig frei wählen.

```
println(new StringBuilder("Hallo Sonja!"));
```

6.4. Vereinbaren Sie eine Variable namens `berthold`, die zu irgendeinem *anonymen Typ* gehört.

```
WindowListener berthold = new WindowAdapter() {  
    public void windowClosing(WindowEvent we) {  
        println("Hallo!");  
    }  
};
```