

Vorname

Nachname

Matrikel-Nr

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

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!

**Aufgabe 1 (15 Punkte):** Schreiben Sie 2 Methoden, die den folgenden Spezifikationen entsprechen:

```
1  static boolean istGrossbuchstabe(char c) {
2      // Liefert true, wenn c zwischen 'A' und 'Z' (einschliesslich) liegt.
3      // Der Rumpf darf nur eine einfache Anweisung enthalten!
4      ...
5  }
6
7  static int anzPaare(ArrayList<String[]> alsa) {
8      // Die elementaren Komponenten der Sammlung alsa sind Strings.
9      // Wie viele Paare von nebeneinanderstehenden Grossbuchstaben
10     // (z.B. "AK" oder "IC" oder "ZZ" etc.) enthalten diese Strings?
11     // Diese Methode liefert die Antwort.
12     //
13     // Achtung: Gezaehlt werden sollen nur nicht-ueberlappende Paare
14     // von Grossbuchstaben. Z.B. enthaelt der String "abCDEfg" nur ein
15     // Paar ("CD") und einen einzelnen Grossbuchstaben ("E"). Der String
16     // "aBCDEfGH" enthaelt 3 Paare ("BC", "DE" und "GH").
17     //
18     // Als Grossbuchstaben sollen alle Zeichen zwischen 'A' und 'Z'
19     // (einschliesslich) gelten.
20     ...
21 } // anzPaare
```

**Aufgabe 2 (15 Punkte):** Schreiben Sie eine Methode, die der folgenden Spezifikation entspricht:

```
1  static boolean alleZiffernSindGerade(long n) {
2      // Liefert true, wenn die Darstellung von n als Dezimalzahl
3      // nur gerade Ziffern (0, 2, 4, 6, 8) enthaelt, sonst false.
4      // Beispiele:
5      // alleZiffernSindGerade(    0L) ist gleich true
6      // alleZiffernSindGerade(    1L) ist gleich false
7      // alleZiffernSindGerade(    8L) ist gleich true
8      // alleZiffernSindGerade( 8124L) ist gleich false
9      // alleZiffernSindGerade( 7246L) ist gleich false
10     // alleZiffernSindGerade(-8821L) ist gleich false
11     // alleZiffernSindGerade(-2808L) ist gleich true
12     // alleZiffernSindGerade(-1732L) ist gleich false
13     ...
14 }
```

**Aufgabe 3 (15 Punkte):** Betrachten Sie das folgende Codestück:

```
1 class Carola {
2     static Character[] carl;
3
4     static public void main(String[] _) {
5         carl = new Character[3];
6         carl[0] = 'X';
7         carl[1] = 'Y';
8         ...
9     }
10    ...
11 } // class Carola
```

Zeichnen Sie drei Bojen die darstellen, wie die Variable `carl` in den folgenden drei Momenten aussieht:

1. Nachdem die Vereinbarung in Zeile 2 ausgeführt wurde.
2. Nachdem die Zuweisung in Zeile 5 ausgeführt wurde.
3. Nachdem die Zuweisung in Zeile 7 ausgeführt wurde.

**Aufgabe 4 (15 Punkte)** Betrachten Sie die folgenden drei Codestücke:

```
1 // Codestück A:
2 int a1 = 3;
3 int a2 = -1;
4 int a3 = 5;
5 for (int a4=0; a4<10; a4=a4+3) {
6     a1 = a1 * a2;
7     a2 = a2 * -1;
8     a3 = a1 - a3 + a4;
9     println("a3: " + a3);
10 }
11
12 // Codestück B:
13 String s = "A B C D E";
14 int n = 0;
15 while (true) {
16     n = s.indexOf(' ', n+1);
17     if (n==-1) break;
18     printf("Ab %d: %s%n", n, s.substring(n));
19 }
20
21 // Codestück C:
22 final int MAX = 5;
23 for (int i=1; i<=MAX; i++) {
24     for (int j=1; j<=i; j++) p("X");
25     p(' ');
26     for (int j=i; j<=MAX; j++) p("O");
27     println();
28 }
```

Geben Sie für jedes Codestück an, was es (zur Standardausgabe) ausgibt.

**Aufgabe 5 (15 Punkte):** Betrachten Sie die folgenden Klassenvereinbarungen:

```

1 class K1 {
2     double d = 7.7;
3     int i;
4
5     K1(double d) {
6         this.d = d;
7         this.i = 0;
8     }
9
10    K1(int i) {
11        this.d = i;
12        this.i = 2*i;
13    }
14
15    public String toString() {
16        return "d: " + d + ", i: " + i;
17    }
18 } // class K1
19 // -----
20 class K2 extends K1 {
21     String s;
22
23     K2() {
24         super(3.5);
25         s = "333";
26     }
27
28     K2(String s) {
29         super(4);
30         this.s = s;
31     }
32
33     public String toString() {
34         return super.toString() + ", s: " + s;
35     }
36 } // class K2
37 // -----
38 class K3 {
39     static public void main(String[] sonja) {
40         K1 k1a = new K1(1.5);
41         K1 k1b = new K1(1);
42
43         K2 k2a = new K2();
44         K2 k2b = new K2("ABC");
45
46         pln("k1a: " + k1a);
47         pln("k1b: " + k1b);
48         pln("k2a: " + k2a);
49         pln("k2b: " + k2b);
50     } // main
51
52     // Eine Methode mit einem kurzen Namen:
53     static void pln(Object ob) {System.out.println(ob);}
54 } // class K3

```

- 5.1. Wieviele *Elemente* werden in jedes K1-Objekt eingebaut und wie heißen diese Elemente?
- 5.2. Wieviele *Elemente* werden in jedes K2-Objekt eingebaut und wie heißen diese Elemente?
- 5.3. Wieviele *Module* existieren in dem Moment, in dem der Ausführer die Zeile 44 fertig ausgeführt hat, und wie heißen diese Module?
- 5.4. Was gibt das Programm K3 (zur Standardausgabe) aus?

**Aufgabe 6: (15 Punkte)** Schreiben Sie eine Klasse namens `MeinFenster` als Erweiterung der Klasse `javax.swing.JFrame`. Jedes Objekt Ihrer Klasse `MeinFenster` soll folgende Eigenschaften haben:

1. Es soll (sobald man es mit `new` erzeugt hat) auf dem Bildschirm als Fenster mit einer Breite von 400 Pixel und einer Höhe von 200 Pixel sichtbar werden.
2. In der Titelleiste soll immer der Titel "`MeinFenster`" stehen (der Programmierer, der die Klasse benutzt, soll keinen anderen Titel festlegen können).
3. Wenn der Benutzer auf den Fenster-Schließen-Knopf klickt, soll das umgebende Programm beendet werden (`EXIT_ON_CLOSE`).
4. Wenn der Benutzer ein `windowIconified`-Ereignis auslöst (indem er z.B. auf den Ikonifizieren-Knopf klickt) soll die Meldung "Hallo" zur Standardausgabe ausgegeben werden.

**Lösung 1 (15 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:**

```
1  static boolean istGrossbuchstabe(char c) {
2      // Liefert true, wenn c zwischen 'A' und 'Z' (einschliesslich) liegt.
3      // Der Rumpf darf nur eine einfache Anweisung enthalten!
4      return 'A' <= c && c <= 'Z';
5  } // istGrossbuchstabe
6
7  static int anzPaare(ArrayList<String[]> alsa) {
8      // Die elementaren Komponenten der Sammlung alsa sind Strings.
9      // Wie viele Paare von nebeneinanderstehenden Grossbuchstaben
10     // (z.B. "AK" oder "IC" oder "ZZ" etc.) enthalten diese Strings?
11     // Diese Methode liefert die Antwort.
12     //
13     // Achtung: Gezaehlt werden sollen nur nicht-ueberlappende Paare
14     // von Grossbuchstaben. Z.B. enthaelt der String "abCDEfg" nur ein
15     // Paar ("CD") und einen einzelnen Grossbuchstaben ("E"). Der String
16     // "aBCDEfGH" enthaelt 3 Paare ("BC", "DE" und "GH").
17     //
18     // Als Grossbuchstaben sollen alle Zeichen zwischen 'A' und 'Z'
19     // (einschliesslich) gelten.
20
21     int anzPaare = 0;
22
23     for (String[] sa : alsa) {
24         for (String s : sa) {
25             for (int i=0; i<s.length()-1; i++) {
26                 char c1 = s.charAt(i);
27                 char c2 = s.charAt(i+1);
28                 if (istGrossbuchstabe(c1) &&
29                     istGrossbuchstabe(c2)) {
30                     anzPaare++;
31                     i++;
32                 }
33             }
34         }
35     }
36     return anzPaare;
37 } // anzPaare
```

**Lösung 2 (15 Punkte): Schreiben Sie eine Methode, die der folgenden Spezifikation entspricht:**

```
1  static boolean alleZiffernSindGerade(long n) {
2      // Liefert true, wenn die Darstellung von n als Dezimalzahl
3      // nur gerade Ziffern (0, 2, 4, 6, 8) enthaelt, sonst false.
4
5      while (true) {
6          long ziff = n%10;
7          if (ziff%2 != 0) return false;
8          n = n/10;
9          if (n == 0) break;
10     }
11     return true;
12 } // alleZiffernSindGerade
```

**Lösung 3 (15 Punkte)** Betrachten Sie das folgende Codestück:

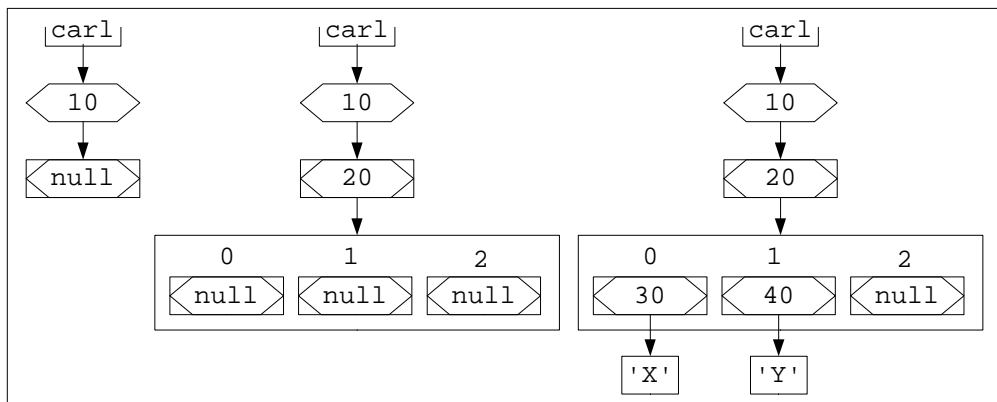
```

1 class Carola {
2     static Character[] carl;
3
4     static public void main(String[] _) {
5         carl = new Character[3];
6         carl[0] = 'X';
7         carl[1] = 'Y';
8         ...
9     }
10    ...
11 } // class Carola

```

Zeichnen Sie drei Bojen die darstellen, wie die Variable `carl` in den folgenden drei Momenten aussieht:

1. Nachdem die Vereinbarung in Zeile 2 ausgeführt wurde.
2. Nachdem die Zuweisung in Zeile 5 ausgeführt wurde.
3. Nachdem die Zuweisung in Zeile 7 ausgeführt wurde.

**Lösung 4 (15 Punkte):** Geben Sie für jedes Codestück an, was es (zur Standardausgabe) ausgibt.

```

-----
// Codestueck A:

```

```

a3: -8
a3: 8
a3: 1
a3: 11

```

```

-----
// Codestueck B:

```

```

Ab 1:  B C D E
Ab 3:  C D E
Ab 5:  D E
Ab 7:  E

```

```

-----
// Codestueck C:

```

```

X 00000
XX 0000
XXX 000
XXXX 00
XXXXX 0

```

**Lösung 5 (15 Punkte):** Betrachten Sie die folgenden Klassenvereinbarungen ...

5.1. Wieviele *Elemente* werden in jedes  $K1$ -Objekt eingebaut und wie heißen diese Elemente?

**3 Elemente:** `d`, `i`, `toString`

5.2. Wieviele *Elemente* werden in jedes  $K2$ -Objekt eingebaut und wie heißen diese Elemente?

**5 Elemente:** `d`, `i`, `s`, `toString`, `toString`

auch akzeptabel:

**4 Elemente:** `d`, `i`, `s`, `toString`

5.3. Wieviele *Module* existieren in dem Moment, in dem der Ausfühler die Zeile 33 fertig ausgeführt hat, und wie heißen diese Module?

**7 Module:** 3 Klassen ( $K1$ ,  $K2$ ,  $K3$ ) und 4 Objekte (`k1a`, `k1b`, `k2a`, `k2b`)

5.4. Was gibt das Programm  $K3$  (zur Standardausgabe) aus?

```
k1a: d: 1.5, i: 0
k1b: d: 1.0, i: 2
k2a: d: 3.5, i: 0, s: 333
k2b: d: 4.0, i: 8, s: ABC
```

**Lösung 6: (15 Punkte)** Schreiben Sie eine Klasse namens `MeinFenster` als Erweiterung der Klasse `javax.swing.JFrame`. Jedes Objekt dieser Klasse soll folgende Eigenschaften haben:

1. Sobald es (mit `new` erzeugt) wurde, soll es auf dem Bildschirm als ein Fenster mit einer Breite von 400 Pixel und einer Höhe von 200 Pixel sichtbar werden.

2. In der Titelleiste soll immer der Titel "`MeinFenster`" stehen (der Programmierer, der die Klasse benutzt, soll keinen anderen Titel festlegen können).

3. Wenn der Benutzer auf den Fenster-Schließen-Knopf klickt, soll das umgebende Programm beendet werden (`EXIT_ON_CLOSE`).

4. Wenn der Benutzer ein `windowIconified`-Ereignis auslöst (indem er z.B. auf den Ikonifizieren-Knopf klickt) soll die Meldung "`Hallo`" zur Standardausgabe ausgegeben werden.

```
1  import javax.swing.JFrame;
2  import java.awt.event.WindowAdapter;
3  import java.awt.event.WindowEvent;
4
5  class MeinFenster extends JFrame {
6
7      MeinFenster(String titel) {
8          super(titel);
9          this.setBounds(100, 50, 400, 200);
10         this.setDefaultCloseOperation(EXIT_ON_CLOSE);
11         this.addWindowListener(new WindowAdapter() {
12             public void windowIconified(WindowEvent we) {
13                 System.out.println("Hallo!");
14             }
15         });
16         this.setVisible(true);
17     } // Konstruktor MeinFenster
18 } // class MeinFenster
```