

Vorname

Nachname

Matrikel-Nr.

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

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

Aufgabe 1 (20 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1  static public int anzDurchEnTeilbar(int[][] irr, int n) {
2      // Die elementaren Komponenten der Reihung irr sind vom Typ int.
3      // Wie viele dieser elementaren Komponenten sind durch n teilbar?
4      // Diese Funktion liefert die Antwort.
5
6      // Beispiel: Sei die Reihungsvariable irr01 vereinbart wie folgt:
7      // int[][] irr01 = { {0, 1, 4, 6}, {3, 6, 9}, {}, {31, 5} };
8      // Dann gilt:
9      // anzDurchEnTeilbar(irr01, 3) ist gleich 5
10     // (weil die 5 int-Werte 0, 6, 3, 6 und 9 durch 3 teilbar sind).
11     ...
12 }
```

Aufgabe 2 (20 Punkte) Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1  static public long maxDezimalZiffer(long zahl) {
2      // Wie lautet die groesste Ziffer, die in der dezimalen Darstellung
3      // der Zahl zahl vorkommt? Diese Funktion liefert die Antwort.
4      //
5      // Beispiele:
6      // maxDezimalZiffer(    3152L) ist gleich 5
7      // maxDezimalZiffer(   -3152L) ist gleich 5
8      // maxDezimalZiffer(    3773L) ist gleich 7
9      // maxDezimalZiffer(   -3773L) ist gleich 7
10     // maxDezimalZiffer( 987654321L) ist gleich 9
11     // maxDezimalZiffer(-987654321L) ist gleich 9
12     // maxDezimalZiffer( 101010101L) ist gleich 1
13     // maxDezimalZiffer(-101010101L) ist gleich 1
14     // maxDezimalZiffer(      0L) ist gleich 0
15     ...
16 }
```

Aufgabe 3 (15 Punkte): Betrachten Sie die folgende Java-Klasse:

```
1  class Klodwig {
2      static private Integer[] ir;
3
4      static public void main(String[] _) {
5          ir = new Integer[2];
6          ir[1] = new Integer(17);
7          ...
8      } // main
9  } // class Klodwig
10
```

Wie sieht die Variable **ir** in den folgenden Momenten als Boje dargestellt aus

1. wenn die Zeile 2 fertig ausgeführt ist?
2. wenn die Zeile 5 fertig ausgeführt ist?
3. wenn die Zeile 6 fertig ausgeführt ist?

Beantworten Sie diese Fragen, indem Sie die Variable **ir** *dreimal als Boje darstellen* (in vereinfachter oder ausführlicher Darstellung, wie Sie wollen).

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

```

1  class KA {
2      String name;
3      KA(String name) {this.name = name;}
4  }
5
6  class KB {
7      static public KA v1 = new KA("v1");
8
9      KA v2 = new KA("v2");
10     KA v3 = new KA("v3");
11 }
12
13 class KC {
14     static public void main(String[] _) {
15         KA v4 = new KA("v4");
16         System.out.println(KB.v1);
17         KB w1 = new KB();
18         KB w2;
19     } // main
20 }

```

Angenommen, der Benutzer startet das Programm (mit der Hauptklasse) KC. Sei M15 der Moment, in dem der Ausführer die Zeile 15 gerade fertig ausgeführt hat. Die Momente M16, M17 und M18 seien entsprechend definiert. Füllen Sie die folgende Tabelle aus. Tragen Sie in jeder Zeile ein, wieviele Module in dem betreffenden Moment existieren und wie diese Module heissen.

Im Moment	Anzahl der Module	Namen der Module
M15		
M16		
M17		
M18		

Aufgabe 5 (15 Punkte): Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

```

1      // Schleife 5.1.:
2      int[] zal = {4, 1, 4, 9, 4, 0, 1};
3      int[] anz = new int[10];
4      for (int z : zal) anz[z]++;
5      for (int a : anz) p(a + " ");
6      pln();

7      // Schleife 5.2:
8      for (int i=0; i<=4; i++) {
9          for (int j=0; j<i; j++) p("X");
10         p("O");
11         for (int j=i; j<4; j++) p("X");
12         pln();
13     }

14     // Schleife 5.3:
15     for (int i=0; i<=5; i++) {
16         for (int j=i; j<5; j++) p("O");
17         p("X");
18         for (int j=0; j<i; j++) p("O");
19         pln();
20     }

21     // Schleife 5.4:
22     String s = "XXXXX";
23
24     for (int i=0; i<=s.length(); i++) {
25         String sa = s.substring(i);
26         String sb = s.substring(0, i);
27         pln(sa + "O" + sb);
28     }

```

`p` und `pln` sind auch hier Abkürzungen für `System.out.print` und `System.out.println`.

Aufgabe 6: (15 Punkte)

6.1. Was ist (im Zusammenhang mit der Programmiersprache Java) ein *Behälterobjekt* (engl. container object)?

6.2. Was ist ein *Grabo-Objekt* (engl. GUI object)? Geben Sie die wichtigste Eigenschaft eines Grabo-Objektes an.

6.3. Mit was für *Datenquellen* bzw. *Datensenken* kann man Java-Stromobjekte (engl. stream objects) verbinden?

6.4. Welche Frage stellt sich der Java-Ausführer immer dann, wenn während der Ausführung eines Programms eine Ausnahme `a` geworfen wird?

6.5. Angenommen, ein Java-Programm besteht aus der Hauptklasse `H1` und den Nebenklassen `N1`, `N2` und `N3`. Wann wird die Hauptklasse geladen? Und wann wird die Nebenkategorie `N2` geladen?

6.6. Geben Sie von jedem der folgenden drei Befehle an, zu welcher Art (von Befehlen) er gehört:

```

6.6.1.: class Karl { static long n = 123L; }
6.6.2.: 123L
6.6.3.: n = 123L;

```


Lösung 1 (20 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
29 static public int anzDurchEnTeilbar(int[][] irr, int n) {
30     // Die elementaren Komponenten der Reihung irr sind vom Typ int.
31     // Wie viele dieser elementaren Komponenten sind durch n teilbar?
32     // Diese Funktion liefert die Antwort.
33
34     int erg = 0;
35
36     for (int[] ir : irr) {
37         for (int i : ir) {
38             if (i%n==0) erg++;
39         }
40     }
41
42     return erg;
43 } // anzDurchEnTeilbar
```

Lösung 2 (20 Punkte) Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1 static public long maxDezimalZiffer(long zahl) {
2     // Wie lautet die groesste Ziffer, die in der dezimalen Darstellung
3     // der Zahl zahl vorkommt? Diese Funktion liefert die Antwort.
4
5     long erg = 0;
6     long ziff;
7     zahl = Math.abs(zahl);
8
9     while (true) {
10         if (zahl==0) break;
11         ziff = zahl%10;
12         zahl = zahl/10;
13         if (ziff>erg) erg=ziff;
14     }
15
16     return erg;
17 } // maxDezimalZiffer
```

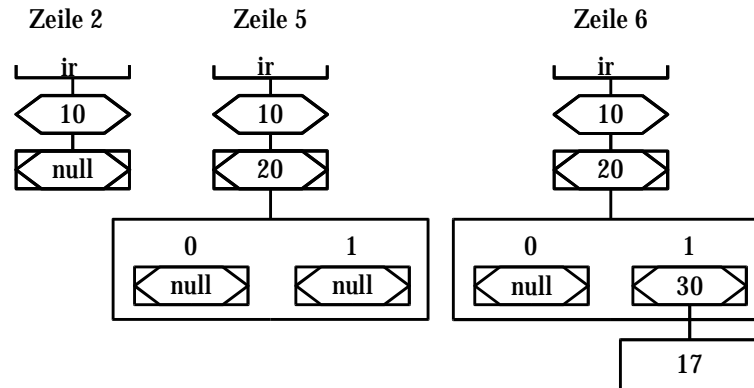
Lösung 3 (15 Punkte): Betrachten Sie die folgende Java-Klasse:

```
1 class Klodwig {
2     static private Integer[] ir;
3
4     static public void main(String[] _) {
5         ir = new Integer[2];
6         ir[1] = new Integer(17);
7         ...
8     } // main
9 } // class Klodwig
```

Wie sieht die Variable **ir** in den folgenden Momenten als Boje dargestellt aus

1. wenn die Zeile 2 fertig ausgeführt ist?
2. wenn die Zeile 5 fertig ausgeführt ist?
3. wenn die Zeile 6 fertig ausgeführt ist?

Beantworten Sie diese Fragen, indem Sie die Variable **ir** *dreimal als Boje darstellen* (in vereinfachter oder ausführlicher Darstellung, wie Sie wollen).



Lösung 4 (15 Punkte): Betrachten Sie die folgenden drei Klassenvereinbarungen:

```

1  class KA {
2      String name;
3      KA(String name) {this.name = name;}
4  }
5
6  class KB {
7      static public KA v1 = new KA("v1");
8
9      KA v2 = new KA("v2");
10     KA v3 = new KA("v3");
11 }
12
13 class KC {
14     static public void main(String[] _) {
15         KA v4 = new KA("v4");
16         System.out.println(KB.v1);
17         KB w1 = new KB();
18         KB w2;
19     } // main
20 }

```

Angenommen, der Benutzer startet das Programm (mit der Hauptklasse) `KC`. Sei `M15` der Moment, in dem der Ausführer die Zeile 15 gerade fertig ausgeführt hat. Die Momente `M16`, `M17` und `M18` seien entsprechend definiert. Füllen Sie die folgende Tabelle aus. Tragen Sie in jeder Zeile ein, wieviele Module in dem betreffenden Moment existieren und wie diese Module heissen.

Im Moment	Anzahl der Module	Namen der Module
M15	3	KC, KA, v4
M16	5	KC, KA, KB, v4, KB.v1
M17	8	KC, KA, KB, v4, KB.v1, w1, w1.v2, w1.v3
M18	8	KC, KA, KB, v4, KB.v1, w1, w1.v2, w1.v3

Lösung 5 (15 Punkte): Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

```

1      // Schleife 5.1.:
2      int[] zal = {4, 1, 4, 9, 4, 0, 1};
3      int[] anz = new int[10];
4      for (int z : zal) anz[z]++;
5      for (int a : anz) p(a + " ");
6      pln();

7      // Schleife 5.2:
8      for (int i=0; i<=4; i++) {
9          for (int j=0; j<i; j++) p("X");
10         p("O");
11         for (int j=i; j<4; j++) p("X");
12         pln();
13     }

14     // Schleife 5.3:
15     for (int i=0; i<=5; i++) {
16         for (int j=i; j<5; j++) p("O");
17         p("X");
18         for (int j=0; j<i; j++) p("O");
19         pln();
20     }

21     // Schleife 5.4:
22     String s = "XXXXX";
23
24     for (int i=0; i<=s.length(); i++) {
25         String sa = s.substring(i);
26         String sb = s.substring(0, i);
27         pln(sa + "O" + sb);
28     }

```

p und pln sind auch hier Abkürzungen für System.out.print und System.out.println.

```

Schleife 5.1: -----
1 2 0 0 3 0 0 0 0 1
Schleife 5.2: -----
OXXXX
XOXXX
XXOXX
XXXOX
XXXXO
Schleife 5.3:-----
OXXXXX
OXXXXO
OOXXOO
OOXOOO
OXOOOO
XOOOOO
Schleife 5.4: -----
XXXXXO
XXXXOX
XXXOXX
XXOXXX
XOXXXX
OXXXXX
-----

```

Lösung 6: (15 Punkte)

6.1. Was ist (im Zusammenhang mit der Programmiersprache Java) ein *Behälterobjekt* (engl. container object)?

Ein Behälterobjekt ist ein Grabo-Objekt, in welches man andere Grabo-Objekte hineintun kann.

6.2. Was ist ein *Grabo-Objekt* (engl. GUI object)? Geben Sie die wichtigste Eigenschaft eines Grabo-Objektes an.

Ein Grabo-Objekt ist ein Objekt, welches nach seiner Erzeugung (mehr oder weniger) automatisch auf dem Bildschirm dargestellt wird.

6.3. Mit was für *Datenquellen* bzw. *Datensenken* kann man Java-Stromobjekte (engl. stream objects) verbinden?

Mit Dateien (engl. files), mit Reihungen der Typen `byte[]` und `char[]` und mit String-Objekten.

6.4. Welche Frage stellt sich der Java-Ausführer immer dann, wenn während der Ausführung eines Programms eine Ausnahme `a` geworfen wird?

Der Ausführer fragt sich: "Trat die Ausnahme `a` in einem try-Block auf, dem ein zum Fangen von `a` geeigneter catch-Block folgt?".

6.5. Angenommen, ein Java-Programm besteht aus der Hauptklasse `H1` und den Nebenklassen `N1`, `N2` und `N3`. Wann wird die Hauptklasse geladen? Und wann wird die Nebenkasse `N2` geladen?

Die Hauptklasse `H1` wird geladen, wenn das Programm `H1` gestartet wird.

Die Nebenkasse `N2` wird geladen, wenn sie zum ersten Mal gebraucht wird.

6.6. Geben Sie von jedem der folgenden drei Befehle an, zu welcher Art (von Befehlen) er gehört:

6.6.1.: `class Karl { static long n = 123L; }`

Vereinbarung (einer Klasse)

6.6.2.: `123L`

Ausdruck (vom Typ `long`)

6.6.3.: `n = 123L;`

Anweisung (Zuweisung)