

Vorname

Nachname

Matrikel-Nr

Schreiben Sie jede **Lösung** möglichst auf die Vorderseite eines **neuen** Blattes (und lassen Sie die Rückseiten Ihrer Lösungsblätter **leer**). Insgesamt erreichbar sind **100 Punkte**.

Aufgabe 1 (15 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```
1 static int[] lottoHisto(int[] lottoZahlen) ;
2 // Verlaesst sich darauf, dass die Reihung lottoZahlen nur Zahlen
3 // zwischen 1 und 49 enthaelt (z.B. alle Zahlen, die in den letzten
4 // 3 Jahren gezogen wurden). Liefert als Ergebnis ein Histogramm,
5 // d.h. eine Reihung erg der Laenge 49. In der Komponenten erg[0]
6 // steht, wie oft die Zahl 1 in der Reihung lottoZahlen vorkommt,
7 // in der Komponenten erg[1] steht die Haeufigkeit der 2, in erg[2]
8 // die Haeufigkeit der 3, ... und in erg[48] die Haeufigkeit der 49.
```

Aufgabe 2 (20 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```
1 static Vector einMalige(String[] vieleNamen) ;
2 // Liefert als Ergebnis einen Vector, der alle Strings aus der
3 // Reihung vieleNamen enthaelt, die nur einmal (und nicht mehrmals)
4 // in der Reihung vieleNamen vorkommen. Beispiel:
5 // Wenn die Reihung vieleNamen die Strings {"Anna", "Alfred", "Bert",
6 // "Beate", "Alfred", "Celia", "Carl", "Beate", "Carl", "Carl",
7 // "Carl"} enthaelt, dann muss der Ergebnisvektor die Strings
8 // ["Anna", "Bert", "Celia"] enthalten.
```

Hinweis: Es ist durchaus erlaubt, zuerst eine **Hilfsfunktion** (z.B. eine namens **anzahlVorkommen**) zu vereinbaren und dann erst die Funktion **einMalige** zu programmieren. Es gibt aber auch Lösungen **ohne** eine solche Hilfsfunktion.

Aufgabe 3 (15 Punkte): Betrachten Sie die folgenden drei **Schleifen** und geben Sie für jede Schleife an, welche **Zeile(n)** sie zur Standardausgabe (d.h. zum Bildschirm) ausgibt:

```
1 // Schleife 1:
2 for (int i=3; i<11; i+=2) {
3     i--;
4     System.out.print(i + " ");
5     i++;
6 }
7 System.out.println();
8
9 // Schleife 2:
10 for (int i=1; i<=3; i++) {
11     for (int j=i; j<=3; j++) {
12         System.out.print(j);
13     }
14     System.out.println();
15 }
16
17 // Schleife 3:
18 S1: for (int i=1; i<=4; i++) {
19     System.out.println("X");
20     S2: for (int j=1; j<=4; j++) {
21         if (i == j) continue S1;
22         System.out.print(i+j);
23     }
24 }
25 System.out.println();
```

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

```
1 class K1 {
2     static int n1 = 1;
3     int n2 = n1++;
4 } // class K1
5 // -----
6 class K2 {
7     K1 ob1 = new K1();
8     K1 ob2 = new K1();
9     K1 ob3;
10 } // class K2
11 // -----
12 class K3 {
13     static public void main1(String[] susi) {
14         K1 ob4 = new K1();
15         K2 ob5 = new K2();
16         ...
17     } // main
18 } // class K3
```

Nehmen Sie an, dass der Ausführer das Programm **K3** bis **Zeile 15** (einschliesslich) ausgeführt hat. Zeichnen Sie **alle Variablen**, die bis dahin erzeugt wurden, als **Bojen**. Verwenden Sie dabei die Zahlen [1], [2], [3], ... (in dieser Reihenfolge) als **Referenzen**.

Tip: Zeichnen Sie (zumindest erstmal) mit Bleistift und möglichst "großzügig".

Aufgabe 5 (20 Punkte): Nehmen Sie noch einmal an, dass der Ausführer das Programm **K3** (aus der vorigen Aufgabe) bis **Zeile 15** (einschliesslich) ausgeführt hat. Was gilt in diesem Moment? Beantworten Sie die folgenden Fragen entsprechend:

4.1. In welcher **Reihenfolge** hat der Ausführer bis dahin **welche Klassen** geladen?

4.2. Wieviele **Objekte** hat der Ausführer bis dahin erzeugt und mit welchen **Namen** könnte man in Zeile 16 (siehe **main**-Methode in der Klasse **K3**) auf diese Objekte zugreifen?

4.3. Wieviele **Module** gehören in diesem Moment zum Programm **K3**?

4.4. Geben Sie für jeden **Modul** an, **wieviele Elemente (Attribute und Methoden)** er enthält, und wie die Elemente **heissen**, etwa so:

"Der Modul **Otto.Emil** enthält **3** Attribute namens **x**, **y** und **summe** und **2** Methoden namens **berechneSumme** und **halbiere**".

Aufgabe 6 (15 Punkte): Betrachten Sie das folgende Programm namens **Ausnahmen**:

```
1 class Ausnahmen {
2     static class AusA extends Exception {};
3     static class AusAX extends AusA    {};
4     static class AusAY extends AusA    {};
5
6     static public void main2(String[] susi) {
7         int n = 0;
8         while (true) {
9             try {
10                 n++;
11                 if (n > 3) throw new Exception();
12                 if (n%2 == 0) throw new AusAY();
13                 if (n%3 == 0) throw new AusAX();
14             } catch (AusAX a) {
15                 System.out.println("A: n ist gleich " + n);
16             } catch (AusA a) {
17                 System.out.println("B: n ist gleich " + n);
18             } catch (Throwable a) {
19                 System.out.println("C: n ist gleich " + n);
20                 break;
21             } finally {
22                 System.out.println("D: n ist gleich " + n);
23             } // try/catch
24         } // while
25         System.out.println("E: n ist gleich " + n);
26     } // main2
27 } // class Ausnahmen
```

Was gibt dieses Programm zur Standardausgabe (d.h. zum Bildschirm) aus?

Lösung 1:

```

1  static int[] lottoHisto(int[] lottoZahlen) {
2
3      int[] erg = new int[] {
4          0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
5          0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
6          0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
7          0, 0, 0, 0, 0, 0, 0
8      };
9
10     // int[] erg = new int[49];
11     // for (int i=0; i<erg.length; i++) erg[i] = 0;
12
13     // int[] erg = new int[49];
14     // Arrays.fill(erg, 0); // Einfacher mit fill
15
16     for (int i=0; i<lottoZahlen.length; i++) {
17         int lottoZahl = lottoZahlen[i];
18         erg[lottoZahl-1]++;
19     }
20
21     return erg;
22 } // lottoHisto

```

Lösung 2:

```

1  // Aufgabe 2: Loesung 1 (ohne Hilfsfunktion, "in einem Stueck")
2  static Vector einMalige(String[] vieleNamen) {
3
4      Vector erg = new Vector();
5      Schleife1: for (int i1=0; i1<vieleNamen.length; i1++) {
6          String name1 = vieleNamen[i1];
7          Schleife2: for (int i2=0; i2<vieleNamen.length; i2++) {
8              String name2 = vieleNamen[i2];
9              if (i1 != i2 && name1.equals(name2) ) continue Schleife1;
10             } // Schleife2
11
12             erg.add(name1);
13         } // Schleife1
14         return erg;
15     } // einMalige
16     // -----
17     // Aufgabe 2: Loesung 2 (mit Hilfsfunktion anzahlVorkommen)
18     static Vector einMalige2(String[] vieleNamen) {
19         Vector erg = new Vector();
20         for (int i=0; i<vieleNamen.length; i++) {
21             String name = vieleNamen[i];
22             if (anzahlVorkommen(name, vieleNamen) == 1) erg.add(name);
23         }
24         return erg;
25     } // import java.util.Arrays;
26     // -----
27     static int anzahlVorkommen(String einName, String[] vieleNamen) {
28         int anzahl = 0;
29         for (int i=0; i<vieleNamen.length; i++) {
30             if (einName.equals(vieleNamen[i]) ) anzahl++;
31         }
32         return anzahl;
33     } // anzahlVorkommen

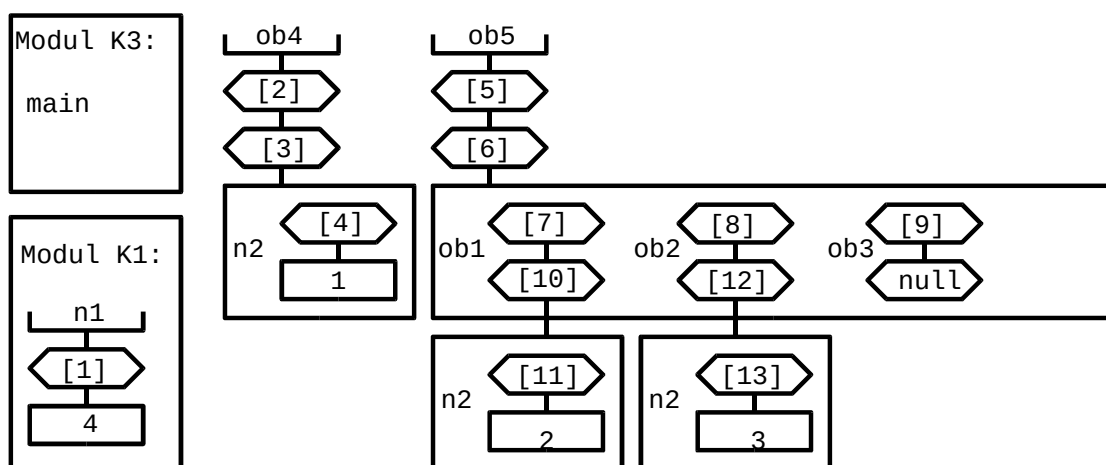
```

Lösung 3:

```

1  -----
2  2  4  6  8
3  -----
4  123
5  23
6  3
7  -----
8  X
9  X
10 3X
11 45X
12 567
13 -----

```

Lösung 4: Alle Variablen in Bojendarstellung:**Lösung 5:**

5.1. Die Klassen K3, K1, K2 (in dieser Reihenfolge)

5.2. **Vier** Objekte namens **ob4**, **ob5**, **ob5.ob1**, **ob5.ob2**

6.3. **Sieben** Module.

5.4.

Der Modul **K1** enthält **1** Attribut namens **n1**.

Der Modul **K2** enthält **0** Elemente.

Der Modul **K3** enthält **1** Methode namens **main**.

Der Modul **ob4** enthält **1** Attribut namens **n2**.

Der Modul **ob5** enthält **3** Attribute namens **ob1**, **ob2** und **ob3**.

Der Modul **ob5.ob1** enthält **1** Attribut namens **n2**.

Der Modul **ob5.ob2** enthält **1** Attribut namens **n2**.

Lösung 6: Ausgabe des Programms **Ausnahmen:**

```

1 D: n ist gleich 1
2 B: n ist gleich 2
3 D: n ist gleich 2
4 A: n ist gleich 3
5 D: n ist gleich 3
6 C: n ist gleich 4
7 D: n ist gleich 4
8 E: n ist gleich 4

```