

Vorname

Nachname

Matrikel-Nr

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

Aufgabe 1 (20 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```

1 static void printAnzahlUndMax(ArrayList[] ral) {
2     // Verlaesst sich darauf, dass die Komponenten der Reihung ral
3     // Sammlungen sind, die nur Long-Objekte enthalten. Gibt die
4     // Anzahl dieser Long-Objekte und den primitiven long-Wert des
5     // groessten Long-Objekts aus, etwa so:
6
7     // Anzahl der Long-Objekte: 17, maximaler long-Wert: 125.
8
9     // Falls die Reihung ral leer ist oder nur leere Sammlungen
10    // enthaelt, soll nur die Anzahl 0, aber kein maximaler long-Wert
11    // ausgegeben werden, etwa so:
12
13    // Anzahl der Long-Objekte: 0.
14
15    ...
16
17 }
```

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

```

1 static int quersumme8(long zahl) {
2     // Wenn man die zahl im 8-er-System ("als Oktalzahl") darstellt
3     // und die Quersumme dieser 8-er-Zahl berechnet, erhaelt man das
4     // Ergebnis dieser Funktion. Beispiele:
5     //
6     // zahl      zahl      Quersumme der
7     // als 10-er-Zahl als 8-er-Zahl 8-er-Zahl zahl
8     //      7          7          7
9     //      8          10         1
10    //     16          20         2
11    //     17          21         3
12    //     35          43         7
13    //     45          55        10
14    //     -7          -7         7
15    //     -8          -10        1
16    //
17    // Die letzten beiden Beispiele sollen verdeutlichen, dass eine
18    // negative Zahl (z.B. -7 oder -8) immer die gleiche Quersumme
19    // hat wie die entsprechende positive Zahl (7 bzw. 1 etc.).
20
21    ...
22
23 }
```

Aufgabe 3 (15 Punkte): Betrachten Sie die folgenden vier Befehlsfolgen (Teilaufgaben):

```

1 // Teilaufgabe 3.1.:
2 int n1;
3 char c1 = 'A';
4 for (n1 = 0; n1 < 10; n1 += 2) {
5     if (n1 % 3 == 1) continue;
6     System.out.println("n1: " + n1 + ", c1: " + ++c1);
7 }
8
9 // Teilaufgabe 3.2.:
10 int n21 = 60;
11 int n22 = 105;
12 while (n21 != n22) {
13     if (n21 > n22) {
14         n21 = n21 - n22;
15     } else {
16         n22 = n22 - n21;
17     }
18 } // while
19 System.out.println("n21: " + n21 + ", n22: " + n22);
20
21 // Teilaufgabe 3.3.:
22 for (int n3=4; Math.abs(n3) < 30; n3 *= -3) {
23     n3 /= 2;
24     System.out.print(n3 + " ");
25 }
26 System.out.println();
27
28 // Teilaufgabe 3.4.:
29 int n41 = 1;
30 int n42 = 2;
31 int n43 = 0;
32 char c4 = 'A';
33 do {
34     n43 = n41 + n42;
35     n41 = n42;
36     n42 = n43;
37     System.out.println(c4++ + " " + n43);
38 } while (n43 < 50);
```

Geben Sie für jede der vier Befehlsfolgen an, welche Zeile(n) sie zum Bildschirm ausgibt.

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

```

1  // -----
2  class Exa extends Exception {};
3  class Exb extends Exa {};
4  class Exc extends Exb {};
5  // -----
6  class Aufgabe04 {
7      static void main(String[] sonja) {
8          for (int n=1; n<=3; ++n) {
9              try {
10                 met01(n);
11             } catch (Exa exa) {
12                 System.out.println("A exa!");
13             } finally {
14                 System.out.println("B ----");
15             } // try-catch-finally
16         } // for
17     } // main
18     // -----
19     static void met01(int n) throws Exa {
20         try {
21             met02(n);
22         } catch (Exb exb) {
23             System.out.println("C exb!");
24             throw exb;
25         } finally {
26             System.out.println("D ----");
27         } // try-catch-finally
28     } // met01
29     // -----
30     static void met02(int n) throws Exa {
31         try {
32             switch (n) {
33                 case 1: throw new Exa();
34                 case 2: throw new Exb();
35                 case 3: throw new Exc();
36             } // switch
37         } catch (Exc exc) {
38             System.out.println("E exc!");
39             throw exc;
40         } // try-catch
41     } // met02
42     // -----
43 } // class Aufgabe04
44 // -----

```

Was gibt das Programm Aufgabe04 zum Bildschirm aus?

Aufgabe 5 (15 Punkte):

```

45 class Knoten {
46     static int anzahl = 0;
47     static int getAnzahl() {return anzahl;}
48
49     Knoten oben;
50     Knoten links;
51     Knoten rechts;
52
53     Knoten() {
54         anzahl++;
55     } // Konstruktor Knoten
56 } // class Knoten
57
58 class TestKnoten {
59     static Knoten k1 = new Knoten();
60     static Knoten k2 = new Knoten();
61     static Knoten k3 = new Knoten();
62
63     static void main(String[] sonja) {
64         k1.oben = k1;
65         k2.oben = k1;
66         k3.oben = k1;
67
68         k1.links = k2;
69         k1.rechts = k3;
70         ...
71         pln("Knoten.getAnzahl(): " + Knoten.getAnzahl());
72
73     } // main
74 } // class TestKnoten

```

Stellen Sie sich vor, dass der Ausführer das Programm TestKnoten ausführt und dabei die Zeile 71 erreicht. Wie sehen die Variablen k1, k2 und k3, dargestellt als Bojen, in diesem Moment aus?

Aufgabe 6: (15 Punkte) Betrachten Sie noch einmal das Programm **TestKnoten** aus der vorigen Aufgabe und stellen Sie sich noch einmal den Moment vor, in dem der Ausführer die **Zeile 71** erreicht. Was gilt in diesem Moment? Beantworten Sie die folgenden Fragen entsprechend:

- 5.1. Wieviele **Klassen** hat der Ausführer bis dahin erzeugt und wie heissen diese Klassen?
- 5.2. Wieviele **Objekte** hat der Ausführer bis dahin erzeugt und wie heissen diese Objekte?
- 5.3. Wieviele **Module** gehören in diesem Moment zum Programm **Test**?
- 5.4. Geben Sie für jeden **Modul** an, **wieviele Konstruktoren**, **Attribute** und **Methoden** er enthält, und wie die Attribute und Methoden **heissen**, etwa so:
 "Der Modul **otto** enthält **2** Konstruktoren, **3** Attribute namens **x**, **y** und **summe** und **2** Methoden namens **berechneSumme** und **halbiere**".
- 5.5. Was gibt der Ausführer zum Bildschirm aus, wenn er den Befehl in Zeile 71 ausführt?

Lösung 1 (20 Punkte):

```

1  static void printAnzahlUndMax(ArrayList[] ral) {
2      int  anz = 0;
3      long max = Long.MIN_VALUE;
4      for (int i=0; i<ral.length; i++) {
5          ArrayList ali = ral[i];
6          anz += ali.size();
7
8          for (int j=0; j<ali.size(); j++) {
9              Long wob = (Long) ali.get(j);
10             long wert = wob.longValue();
11             if (max < wert) max = wert;
12         } // for j
13     } // for i
14
15     System.out.print("Anzahl der Long-Objekte: " + anz);
16     if (anz > 0) System.out.print(", maximaler long-Wert: " + max);
17     System.out.println(".");
18 } // printAnzahlUndMax

```

Lösung 2 (20 Punkte):

```

19  static int quersumme8(long zahl) {
20      int quersumme = 0;
21      int ziffer;
22
23      if (zahl < 0) zahl = -zahl;
24
25      while (zahl > 0) {
26          ziffer = (int) zahl % 8;
27          zahl = zahl / 8;
28          quersumme += ziffer;
29      }
30      return quersumme;
31  } // quersumme8

```

Lösung 3 (15 Punkte):

```

32 ----- Schleife1:
33 n1: 0, c1: B
34 n1: 2, c1: C
35 n1: 6, c1: D
36 n1: 8, c1: E
37 -----
38 n21: 15, n22: 15
39 ----- Schleife3:
40 2 -3 4 -6 9 -13
41 ----- Schleife4:
42 A 3
43 B 5
44 C 8
45 D 13
46 E 21
47 F 34
48 G 55

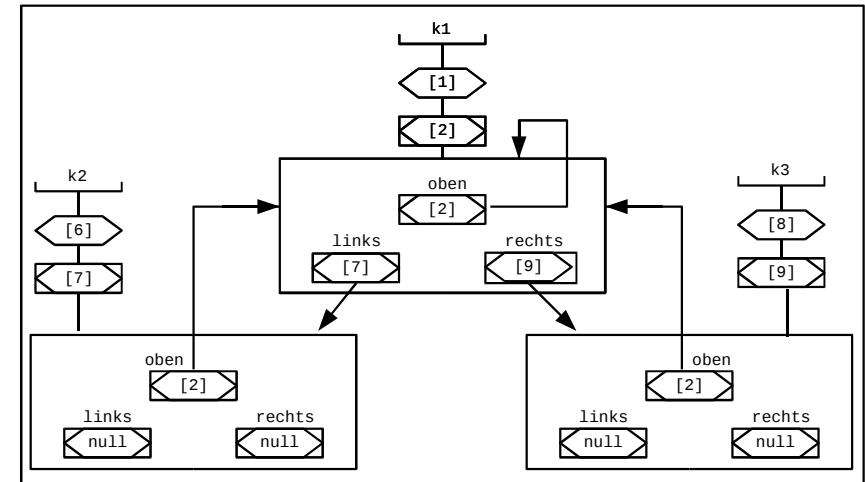
```

Lösung 4 (15 Punkte):

```

49 D ----
50 A exa!
51 B ----
52 C exb!
53 D ----
54 A exa!
55 B ----
56 E exc!
57 C exb!
58 D ----
59 A exa!
60 B ----

```

Lösung 5 (15 Punkte):**Lösung 6 (15 Punkte):**

6.1. Wieviele Klassen hat der Ausführer bis dahin geladen und wie heissen diese Klassen?

Zwei Klassen, Knoten und TestKnoten.

6.2. Wieviele Objekte hat der Ausführer bis dahin erzeugt? Geben Sie für jedes dieser Objekte einen Namen an.

Drei Objekte, TestKnoten.k1, TestKnoten.k2 und TestKnoten.k3.

6.3. Wieviele Module gehören in diesem Moment zum Programm TestKnoten?

Fünf Module.

6.4. Geben Sie für jeden Modul an, wieviele Konstruktoren, Attribute und Methoden er enthält, und wie die Attribute und Methoden heissen:

Der Modul TestKnoten enthält einen (geschenkten Standard-) Konstruktor, drei Attribute namens k1, k2 und k3 und eine Methode namens main.

Der Modul Knoten enthält einen Konstruktor, eine Methode namens getAnzahl und ein Attribut namens anzahl.

Der Modul TestKnoten.k1 enthält drei Attribute namens oben, links und rechts.

Der Modul TestKnoten.k2 enthält drei Attribute namens oben, links und rechts.

Der Modul TestKnoten.k3 enthält drei Attribute namens oben, links und rechts.

6.5. Was gibt der Ausführer zum Bildschirm aus wenn er den Befehl in Zeile 71 ausführt?

Knoten.getAnzahl(): 3