

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

```
1  static int anZahl(Vector v);
2      // Verlaesst sich darauf, dass v nur String-Objekte enthaelt (0 oder
3      // mehr). Zaehlt, wie oft in diesen Strings unmittelbar nach einem
4      // grossen Buchstaben ('A' bis 'Z') ein kleiner Buchstabe ('a' bis
5      // 'z') steht und liefert diese Anzahl als Ergebnis. Beispiele:
6      //
7      // Sei v1 ein Vector-Objekt, welches die drei Strings "Hallo Susi!",
8      // "123 Euro" und "ABCabc" enthaelt.
9      // Sei v2 ein Vector-Objekt, welches den einen String
10     // "Aa Zz zZ aA !$$%&/()=?" enthaelt.
11     // Sei v3 ein Vector-Objekt, welches den einen String "abcABC123"
12     // enthaelt.
13     // Sei v4 ein Vector-Objekt, welches 0 Strings enthaelt.
14     //
15     // Dann gilt:
16     // anZahl(v1) ist gleich 4 (wegen "Ha", "Su", "Eu" und "Ca"),
17     // anZahl(v2) ist gleich 2 (wegen "Aa" und "Zz")
18     // anZahl(v3) ist gleich 0
19     // anZahl(v4) ist gleich 0
```

Aufgabe 2 (15 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```
1  static String als7erZahl(int n);
2      // Liefert einen String, der n als Zahl im 7-er-System darstellt.
3      // Der String beginnt mit einem Minuszeichen '-', falls n negativ
4      // ist. Falls n gleich 0 oder positiv ist, beginnt der String mit
5      // einem Pluszeichen '+'.
6      // Beispiele:
7      //
8      // als7erZahl( 13) ist gleich "+16"
9      // als7erZahl(-13) ist gleich "-16"
10     // als7erZahl( 18) ist gleich "+30"
11     // als7erZahl( 48) ist gleich "+66"
12     // als7erZahl(  0) ist gleich "+0"
13     // als7erZahl(-0) ist gleich "+0"
```

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      // Schleife1:
2      for (int i=5; i<=15; i++) {
3          System.out.print(i + " ");
4          i += 3;
5      }
6      System.out.println();
7
8      // Schleife2:
9      for (int i=1; i<100; i++) {
10         if (i%2 == 0) continue;
11         if (i%17 == 0) break;
12         System.out.print(i + " ");
13     }
14     System.out.println();
```

```

15
16     // Schleife 3:
17     for (int i=1; i<=4; i++) {
18         for (int j=1; j<i; j++) {
19             System.out.print(" " + i + j + " ");
20         }
21         System.out.println("<-");
22     }

```

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

```

1  // -----
2  class K1 {
3      static K2 ob1 = new K2(11);
4      static K2 ob2;
5
6      K2 ob3;
7      K2 ob4;
8      K2 ob5 = new K2(55);
9
10     K1(int n) {
11         ob4 = new K2(n);
12     } // Konstruktor K1
13
14 } // class K1
15 // -----
16 class K2 {
17     public int g;
18     K2(int i) { g = i;}
19 } // class K2
20 // -----
21 class K3 {
22     static public void main(String[] susi) {
23         K1 ob6;
24         K1 ob7 = new K1(77);
25         ...
26     } // main
27 } // class K3
28 // -----

```

Nehmen Sie an, dass der Ausführer das Programm **K3** bis **Zeile 24** (einschliesslich) ausgeführt hat. Was gilt in diesem Moment? Beantworten Sie die folgenden Fragen entsprechend:

- 4.1. Wieviele **Klassen** hat der Ausführer bis dahin geladen und wie heissen diese Klassen?
- 4.2. Wieviele **Objekte** hat der Ausführer bis dahin erzeugt und wie heissen diese Objekte?
- 4.3. Wieviele **Module** gehören in diesem Moment zum Programm **K3**?
- 4.4. 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.Emil** enthält **2** Konstruktoren, **3** Attribute namens **x, y** und **summe** und **2** Methoden namens **berechneSumme** und **halbiere**".

Aufgabe 5 (15 Punkte): Betrachten Sie das folgende Java-Programm:

```

1 class Ausnahmen {
2     static class A1 extends Exception {};
3     static class A11 extends A1      {};
4     static class A12 extends A1      {};
5     // -----
6     static void upro01(int n) throws A11, A12 {
7         if (n%2 == 0) throw new A11();
8         if (n%3 == 0) throw new A12();
9         System.out.println("A: n ist gleich " + n);
10    } // upro01
11    // -----
12    static void upro02(int n) {
13        try {
14            upro01(n);
15            System.out.println("B: n ist gleich " + n);
16        }
17
18        catch (A11 a) {
19            System.out.println("C: n ist gleich " + n);
20        }
21
22        catch (Throwable t) {
23            System.out.println("D: n ist gleich " + n);
24        }
25
26        finally {
27            System.out.println("E: n ist gleich " + n);
28        }
29    } // upro02
30    // -----
31    static public void main(String[] susi) {
32        upro02(1);
33        upro02(6);
34        upro02(9);
35    } // main
36    // -----
37 } // class Ausnahmen

```

Geben Sie an, was dieses Programm **zur Standardausgabe** (d.h. zum Bildschirm) **ausgibt**.

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

```

1 class Person {
2     String vorName;
3     String zuName;
4     Person(String vorName, String zuName) {
5         this.vorName = vorName;
6         this.zuName = zuName;
7     } // Konstruktor Person
8
9     static public void main(String[] susi) {
10        String s1 = new String("Blum");
11        Person p1 = new Person(new String("Anna"), s1);
12        Person p2 = new Person(new String("Bert"), s1);
13        ...
14    } // main
15 } // class Person

```

Wie sehen die Variablen **s1**, **p1** und **p2** in dem Moment aus, in dem der Ausführer die **Zeile 13** erreicht? Zeichnen Sie die **Bojen** der 3 Variablen und verwenden Sie dabei die Zahlen [1], [2], [3], ... (in dieser Reihenfolge) als **Referenzen**.

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

Lösung 1:

```
1  static int anzahl(Vector v) {
2      // Verlaesst sich darauf, dass v nur String-Objekte enthaelt (0 oder
3      // mehr). Zaehlt, wie oft in diesen Strings unmittelbar nach einem
4      // grossen Buchstaben ('A' bis 'Z') ein kleiner Buchstabe ('a' bis
5      // 'z') steht und liefert diese Anzahl als Ergebnis.
6
7      int erg = 0; // Das Ergebnis dieser Funktion
8
9      for (int vi=0; vi<v.size(); vi++) {
10         String s = (String) v.get(vi);
11         for (int si=0; si<s.length()-1; si++) {
12             char    c1 = s.charAt(si);
13             char    c2 = s.charAt(si+1);
14             boolean b1 = ('A' <= c1 && c1 <= 'Z');
15             boolean b2 = ('a' <= c2 && c2 <= 'z');
16             if (b1 && b2) erg++;
17         } // for
18     } // for
19
20     return erg;
21 } // anzahl
```

Lösung 2:

```
1  static String als7erZahl(int n) {
2      // Liefert einen String, der n als Zahl im 7-er-System darstellt.
3      // Der String beginnt mit einem Minuszeichen '-', falls n negativ
4      // ist. Falls n gleich 0 oder positiv ist, beginnt der String mit
5      // einem Pluszeichen '+'.
6
7      StringBuffer erg = new StringBuffer();
8
9      char vorzeichen = '+';
10     if (n<0) {
11         vorzeichen = '-';
12         n          = -n;
13     }
14
15     do {
16         char eineZiffer = (char) (n%7 + '0');
17         erg.insert(0, eineZiffer);
18         n /= 7;
19     } while (n>0);
20
21     erg.insert(0, vorzeichen);
22
23     return erg.toString();
24 } // als7erZahl
```

Lösung 3:

```
1 schleifel:
2 5 9 13
3 schleife2:
4 1 3 5 7 9 11 13 15
5 schleife3:
6 <-
7 21 <-
8 31 32 <-
9 41 42 43 <-
```

Lösung 4:4.1. **Drei** Klassen, **K1**, **K2** und **K3**4.2. **Vier** Objekte, **K1.ob1**, **ob7**, **ob7.ob4**, **ob7.ob5**4.3. **Sieben** Module.

4.4.

Der Modul **K1** enthält 2 Attribute namens **ob1** und **ob2** und 1 Konstruktor.Der Modul **K2** enthält 1 Konstruktor.Der Modul **K3** enthält 1 Methode namens **main**.Der Modul **K1.ob1** enthält 1 Attribut namens **g**.Der Modul **ob7** enthält 3 Attribute namens **ob3**, **ob4** und **ob5**.Der Modul **ob7.ob4** enthält 1 Attribut namens **g**.Der Modul **ob7.ob5** enthält 1 Attribut namens **g**.**Lösung 5:** Ausgabe des Programms **Ausnahmen:**

```

1 A: n ist gleich 1
2 B: n ist gleich 1
3 E: n ist gleich 1
4 C: n ist gleich 6
5 E: n ist gleich 6
6 D: n ist gleich 9
7 E: n ist gleich 9

```

Lösung 6: Die Variablen **s1**, **p1** und **p2** als Bojen dargestellt