

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): Führen Sie die folgende Befehlsfolge mit Papier und Bleistift aus. Geben Sie als Lösung (nur) an, was zum Bildschirm ausgegeben wird:

```
1      int anna = 1;
2      int bert = 127;
3      int carl = 1;
4      while (anna < 2*bert) {
5          carl++;
6          anna *= 3;
7          bert /= 2;
8          AM01.p(anna + " " + bert + " ");
9      }
10     AM01.pln();
```

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

```
1      static StringBuffer als7erZahl(long g) {
2          // Liefert einen StringBuffer, der die Zahl g als Zahl im 7-er-System
3          // ("als Septalzahl") mit einem Vorzeichen davor enthaelt. Beispiele:
4          //
5          // Funktionsaufruf:      Inhalt des StringBuffer-Ergebnisses:
6          // als7erZahl(25)        "+34"
7          // als7erZahl(-7)       "-10"
8          // als7erZahl(48)       "+66"
9          // als7erZahl(-49)      "-100"
10         // als7erZahl(0)        "+0"
11         // als7erZahl(987654321) "+33321631443"
12         ...
13     }
```

Aufgabe 3 (20 Punkte): Schreiben Sie ein Unterprogramm entsprechend der folgenden Spezifikation:

```
1      static void gibDreieckAus(char n) {
2          // Gibt eine Art Dreieck zur Standardausgabe aus, welches aus
3          // "Doppelsternchen" besteht. Beispiel: Der Aufruf
4          // gibDreieckAus(5); gibt die folgenden 5 Zeilen aus:
5          //
6          //          **
7          //         ****
8          //        *****
9          //       ********
10         //      *********
11         //
12         // Noch ein Beispiel: Der Aufruf
13         // gibDreieckAus(2); gibt die folgenden 2 Zeilen aus:
14         //
15         //          **
16         //         ****
17         //
18     }
```

Hinweis: Sie können sicher sein, dass der Parameter n keinen negativen Wert enthält, denn zum Ganzzahltyp char gehören keine negativen Zahlen.

Aufgabe 4 (30 Punkte): Betrachten Sie die folgenden Klassenvereinbarungen:

```
1 class Haupt4 {
2     static public void main(String[] susi) {
3         Neben41 arno = new Neben41(17);
4         Neben42 bert = new Neben42(arno);
5         Neben43 carl = new Neben43(bert);
6         ...
7     }
8 } // class Haupt4
9 // -----
10 class Neben41 {
11     static boolean a = true;
12     int anina;
13     Neben41(int anina) {this.anina = anina;}
14 } // class Neben41
15 // -----
16 class Neben42 {
17     static boolean b = true;
18     Neben41 berta;
19     Neben42(Neben41 berta) {this.berta = berta;}
20 } // class Neben42
21 // -----
22 class Neben43 {
23     static boolean c = true;
24     Neben42 celia;
25     Neben43(Neben42 celia) {this.celia = celia;}
26 } // class Neben43T
```

Stellen Sie sich den Moment vor, in dem der Ausführer die Zeile 5 fertig ausgeführt hat.

4.1. Wie sehen in diesem Moment die Variablen **arno**, **bert** und **carl** aus? Stellen Sie die drei Variablen als Bojen dar (wahlweise in ausführlicher oder in abgekürzter Darstellung).

4.2. Wieviele Module existieren in diesem Moment und wie heissen diese Module?

4.3. Geben Sie für jeden der Module an, wieviele Elemente er enthält und wie diese Elemente heissen.

Aufgabe 5 (15 Punkte): Was gibt das Programm Haupt5 zur Standardausgabe aus?

```
1 import de.tfh_berlin.einaus.AM01;
2
3 class Aus1 extends Exception {};
4 class Aus2 extends Exception {};
5
6 class Haupt5 {
7     static public void main(String[] susi) {
8         for (int i=1; i<=3; i++) {
9             try {
10                 met1(i);
11                 met2(i);
12             } catch (Aus1 a) {
13                 AM01.pln("main, Ausnahme: " + i);
14             } catch (Aus2 a) {
15                 AM01.pln("main, Ausnahme: " + i);
16             } // try/catch
17             AM01.pln("main, normal:   " + i);
18         } // for
19     } // meinMain
20
21     static public void met1(int n) throws Aus2 {
22         try {
23             switch(n) {
24                 case 1: throw new Aus1();
25                 case 2: throw new Aus2();
26             }
27             AM01.pln("met1, normal:   " + n);
28         } catch (Aus1 a) {
29             AM01.pln("met1, Ausnahme: " + n);
30         } // try/catch
31     } // met1
32
33     static public void met2(int n) throws Aus1 {
34         try {
35             switch(n) {
36                 case 1: throw new Aus1();
37                 case 2: throw new Aus2();
38             }
39             AM01.pln("met2, normal:   " + n);
40         } catch (Aus2 a) {
41             AM01.pln("met2, Ausnahme: " + n);
42         } // try/catch
43     } // met2
44
45 } // class Haupt5
```

Lösung 1: (15 Punkte) Ausgabe der Befehlsfolge zur Standardausgabe:

```
1  3 63 9 31 27 15 81 7
```

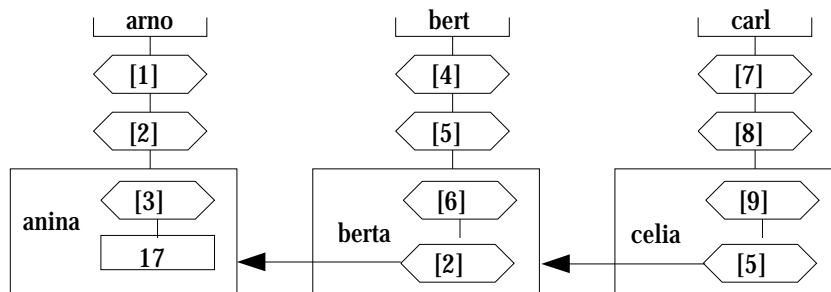
Lösung 2: (20 Punkte) Die Methode als7erZahl:

```
1  static StringBuffer als7erZahl(long g) {
2      // Liefert einen StringBuffer, der die Zahl g als Zahl im 7-er-System
3      // ("als Septalzahl") mit einem Vorzeichen davor enthaelt. Beispiele:
4      //
5      // Funktionsaufruf:          Inhalt des StringBuffer-Ergebnisses:
6      // als7erZahl(25)             "+34"
7      // als7erZahl(-7)            "-10"
8      // als7erZahl(48)            "+66"
9      // als7erZahl(-49)           "-100"
10     // als7erZahl(0)             "+0"
11     // als7erZahl(987654321)     "+33321631443"
12
13     StringBuffer erg = new StringBuffer();
14
15     if (g < 0) {
16         erg.append('-');
17         g = -g;
18     } else {
19         erg.append('+');
20     }
21
22     do {
23         char c = (char) (g % 7 + '0');
24         g = g / 7;
25         erg.insert(1, c);
26     } while (g > 0);
27
28     return erg;
29 } // als7erZahl(
```

Lösung 3: (20 Punkte) Die Methode gibDreieckAus:

```
1  static void gibDreieckAus(char n) {
2      // Gibt eine Art Dreieck zur Standardausgabe aus, welches aus
3      // "Doppelsternchen" besteht. Beispiel:
4      // Der Aufruf gibDreieckAus(5); gibt die folgenden 5 Zeilen aus:
5      //
6      //      **
7      //      ****
8      //      *****
9      //      *******
10     //      *********
11     //
12     // Noch ein Beispiel:
13     // Der Aufruf gibDreieckAus(2); gibt die folgenden 2 Zeilen aus:
14     //
15     //      **
16     //      ****
17
18     for (int zeile=1; zeile<=n; zeile++) {
19         int anzDoppelBlanks = n-zeile;
20         int anzDoppelSterne = zeile;
21         for (int i=1; i<=anzDoppelBlanks; i++) System.out.print(" ");
22         for (int i=1; i<=anzDoppelSterne; i++) System.out.print("***");
23         System.out.println();
24     } // for
25
26 } // gibDreieckAus
```

Lösung 4.1: (15 Punkte) Die Variablen arno, bert und carl in Bojendarstellung:



Lösung 4.2.: (5 Punkte) Wieviele Module existieren in diesem Moment und wie heissen diese Module?

7 Module:

Haupt4, Neben41, Neben42, Neben43, arno , bert und carl.

(arno heisst auch bert.berta und carl.celia.berta, bert heisst auch carl.celia).

Lösung 4.3.: (10 Punkte) Geben Sie für jeden der Module an, wieviele Elemente er enthält und wie diese Elemente heissen.

Modul Haupt4:	1 Element namens	Haupt4.main.
Modul Neben41:	1 Element namens	Neben41.a.
Modul Neben42:	1 Element namens	Neben42.b.
Modul Neben43:	1 Element namens	Neben42.c.
Modul arno:	1 Elemente namens	arno.anina.
Modul bert:	1 Elemente namens	bert.berta.
Modul carl:	1 Element namens	carl.celia.

Lösung 5: (15 Punkte) Ausgabe des Programms Haupt5:

```

1 met1, Ausnahme: 1
2 main, Ausnahme: 1
3 main, normal: 1
4 main, Ausnahme: 2
5 main, normal: 2
6 met1, normal: 3
7 met2, normal: 3
8 main, normal: 3

```