

Vorname	Nachname	Matrikel-Nr

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

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 public int lexiko(short[] ra, short[] rb) {
2      // Vergleicht ra und rb lexikografisch und liefert
3      // eine negative Zahl wenn ra kleiner rb ist
4      // die Zahl 0 wenn ra gleich rb ist
5      // eine positive Zahl wenn ra groesser rb ist. Beispiele:
6      //
7      // short[] sr01 = {7, 2, 5};
8      // short[] sr02 = {7, 2};
9      // short[] sr03 = {7, 2, 5};
10     // short[] sr04 = {7, 2, 5, 0};
11     // short[] sr05 = {7, 2, 2, 2};
12     // short[] sr06 = {7, 3};
13     // short[] sr07 = {};
14     //
15     // lexiko(sr01, sr02) ist eine pos. Zahl, z.B. +1
16     // lexiko(sr01, sr03) ist gleich 0
17     // lexiko(sr01, sr04) ist eine neg. Zahl, z.B. -1
18     // lexiko(sr01, sr05) ist eine pos. Zahl, z.B. +3
19     // lexiko(sr01, sr06) ist eine neg. Zahl, z.B. -1
20     // lexiko(sr01, sr07) ist eine pos. Zahl, z.B. +3
21     // lexiko(sr07, sr01) ist eine neg. Zahl, z.B. -3
22     ...
23 } // lexiko

```

Zur Erinnerung: Bei einem lexikografischen Vergleich werden (von „links nach rechts“) entsprechende Komponenten von **ra** und **rb** miteinander verglichen bis zur ersten Stelle, an der die entsprechenden Komponenten *ungleich* sind ...

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

```

1  static public String expa(String s) {
2      // Verlaesst sich darauf, dass s.length eine gerade Zahl ist.
3      // Interpretiert s.charAt(0) als ein Zeichen und s.charAt(1) als
4      // seine Haeufigkeit (und ebenso fuer s.charAt(2) und s.charAt(3),
5      // s.charAt(4) und s.charAt(5) etc.). Liefert einen String, in
6      // dem jedes Zeichen aus so oft vorkommt, wie seine Haeufigkeit
7      // angibt. Beispiele:
8      //
9      // expa("X\3Y\2Z\5") ist gleich "XXXXYYZZZZZ"
10     // expa("+\5X\5") ist gleich "+++++XXXXX"
11     // expa("#\17") ist gleich "#####"
12     ...
13 } // expa

```

Zur Erinnerung: Ein String wie z. B. "#\17" besteht aus dem Zeichen '#' und der Zahl \17. Nach dem Schrägstrich \ steht eine Zahl im 8-er-System (Oktalzahl), die Zahl \17 entspricht also der Dezimalzahl 15. Der String "X\3Z\2Y\5" hat die Länge 6.

Aufgabe 3 (15 Punkte): Geben Sie für jede der drei Befehlsfolgen an, welche Zeile(n) sie zum Bildschirm ausgibt.

```

1  Teilaufgabe 3.1.:
2  int a = -2;
3  for (int b=0, c=10; a>-16; b++, c-=3) {
4      pln(a+1 + ", " + b + ", " + c%2);
5      a *= 2;
6  }
7
8  Teilaufgabe 3.2.:
9  int d=7, e;
10 while (d > 0) {
11     e = d;
12     while (e > 0) {
13         p(' ');
14         e--;
15     }
16     pln();
17     d-=2;
18 }
19
20 Teilaufgabe 3.3.:
21 int f = 17;
22 int g = 25;
23 pln(++f + ", " + g--);
24 pln(++f + ", " + g--);
25 pln(++f + ", " + g--);

```

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

```

1  int [] anna = {};
2  Integer[] bert = new Integer[2];
3  String [] carl = {"Hallo!", "Sonja!"};
4  Object [] dora = {anna, bert, carl};

```

Stellen Sie die Variable **dora** als Boje dar. Es ist erlaubt, die *vereinfachte* Bojendarstellung zu benutzen. Wenn Sie sich nicht sicher sind, was damit gemeint ist, sollten Sie wahrscheinlich die ausführliche Bojendarstellung benutzen (die ist immer erlaubt).

Aufgabe 5 (15 Punkte): Beantworten Sie die folgenden Fragen möglichst kurz, aber genau (etwa so, wie in den Wiederholungen vor den Vorlesungen)

5.1. Welche charakteristischen Eigenschaften hat ein **Grabo-Objekt**?

5.2. Welche Klassen gelten in Java als **Grabo-Klassen**?

5.3. Was kann man mit einem **Sammlungsobjekt** machen? Nennen Sie möglichst nur charakteristische Möglichkeiten („kaufen“ und „verschenken“ sind **nicht** charakteristisch für Sammlungsobjekte)

5.4. Welche Klassen gelten in Java als **Sammlungs-Klassen**?

5.5. Was kann man mit einem Behälterobjekt machen?

5.6. Welche Klassen gelten in Java als Behälterklassen?

Aufgabe 6 Betrachten Sie die folgenden Klassenvereinbarungen:

```

1  // =====
2  class Perle {
3      int wert;
4      Perle next;
5
6      Perle(int wert) {
7          this.wert = wert;
8          anz++;
9      }
10
11     static int anz;
12     static int getAnz() {return anz;}
13 } // class Perle
14 // =====
15 class PerleTst {
16     //-----
17     static public void main(String[] _) {
18         ...
19         Perle p1 = new Perle(3);
20         ...
21         Perle p2 = new Perle(2);
22         p2.next = new Perle(4);
23         ...
24         p1.next = p2;
25         ...
26     } // main
27     //-----
28     // Eine Methode mit einem kurzen Namen:
29     static void pln(Object ob) {System.out.println(ob);}
30     //-----
31 } //class PerleTst

```

Stellen Sie sich die vier *Zeitpunkte* vor, an denen der Ausführer die Zeile 18, 20, 23 bzw. 25 erreicht. Beantworten Sie die folgenden Fragen für jeden dieser vier Zeitpunkte einmal:

Fragen (Was gilt in diesem Moment?)	Zeile 18	Zeile 20	Zeile 23	Zeile 25
Wieviele Objekte des Typs Perle gibt es?				
Wieviele <i>Module</i> gibt es?				
Wieviele <i>int</i> - <i>Variablen</i> gibt es?				
Wieviele <i>Klassenelemente</i> gibt es?				
Wieviele <i>Objektelemente</i> gibt es?				
Welchen <i>Wert</i> hat die Variable <code>Perle. anz</code> ?				

Tragen Sie Ihre Antworten möglichst gleich in diese Tabelle ein.

Lösung 1 (20 Punkte):

```

1  static public int lexiko(short[] ra, short[] rb) {
2
3      int minLen = Math.min(ra.length, rb.length);
4
5      for (int i=0; i<minLen; i++) {
6          if (ra[i] != rb[i]) return ra[i] - rb[i];
7      }
8      return ra.length - rb.length;
9  } // lexiko

```

Lösung 2 (20 Punkte):

```

1  static public String expa(String s) {
2
3      StringBuilder sb = new StringBuilder();
4
5      for (int i=0; i<s.length()-1; i+=2) {
6          char c = s.charAt(i);
7          char anz = s.charAt(i+1);
8
9          for (char j=1; j<=anz; j++) {
10             sb.append(c);
11         }
12     }
13     return sb.toString();
14
15 } // expa

```

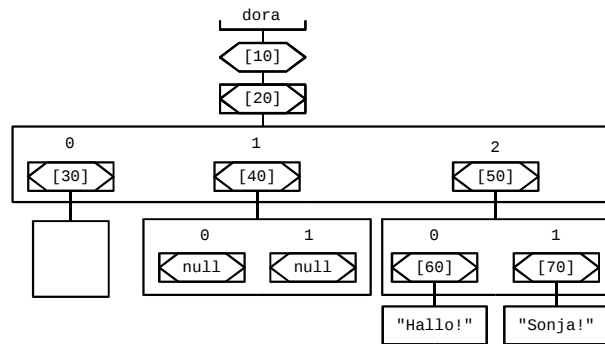
Lösung 3 (15 Punkte):

```

1  .....3.1.
2  -1, 0, 0
3  -3, 1, 1
4  -7, 2, 0
5  .....3.2.
6  *****
7  *****
8  ***
9  *
10 .....3.3.
11 18, 25
12 19, 24
13 20, 23
14 .....

```

Lösung 4: Die Variable dora als Boje dargestellt



Lösung 5 (15 Punkte): Beantworten Sie die folgenden Fragen möglichst kurz, aber genau (etwa so, wie in den Wiederholungen vor den Vorlesungen)

5.1. Welche charakteristischen Eigenschaften hat ein **Grabo-Objekt**?

Ein **Grabo-Objekt besitzt eine grafische Darstellung und wenn man es erzeugt, wird es (mehr oder weniger automatisch) auf dem Bildschirm grafisch dargestellt.**

5.2. Welche Klassen gelten in Java als **Grabo-Klassen**?

Alle Unterklassen der Klasse java.awt.Component.

5.3. Was kann man mit einem **Sammlungsobjekt** machen?

In ein Sammlungsobjekt sob kann man andere Objekte hineintun, man kann darin nach Objekten suchen und man Objekte aus sob wieder entfernen („in sob kann man Objekte sammeln“).

5.4. Welche Klassen gelten in Java als **Sammlungs-Klassen**?

Alle Klassen, die die Schnittstelle Collection implementieren.

5.5. Was kann man mit einem Behälterobjekt machen?

In ein Behälterobjekt bob kann man Grabo-Objekte hineintun.

5.6. Welche Klassen gelten in Java als Behälterklassen?

Alle Unterklassen der Klasse java.awt.Container.

Lösung 6 (15 Punkte): Die Klassen Perle und PerleTst mit (mit richtigen Zeilen-Nrn 1 bis 31):

Fragen (Was gilt in diesem Moment?)	Zeile 18	Zeile 20	Zeile 23	Zeile 25
Wieviele Objekte des Typs Perle gibt es?	0	1	3	3
Wieviele <i>Module</i> gibt es?	2	3	5	5
Wieviele <i>int</i> -Variablen gibt es?	1	2	4	4
Wieviele Klassenelemente gibt es?	3	3	3	3
Wieviele Objektelemente gibt es?	0	2	6	6
Welchen <i>Wert</i> hat die Variable <i>Perle.anz</i> ?	0	1	3	3