

Vorname (bitte deutlich und lesbar)

Nachname (bitte deutlich und lesbar)

Matrikel-Nr (bitte deutlich und lesbar)

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

Schreiben Sie jede Lösung auf die Vorderseite eines *neuen Blattes* (und lassen Sie die Rückseiten Ihrer Lösungsblätter *leer*). Diese Klausur besteht aus 6 Aufgaben.

**Aufgabe 1 (20 Punkte):** Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1  static public boolean istGroesser(ArrayList<String> als, String[] rs) {
2      // Liefert true genau dann wenn
3      // als mehr Komponenten enthaelt als rs oder wenn
4      // als gleich viele Komponenten enthaelt wie rs und jede Komponente
5      // von als groesser ist als die entsprechende Komponenten von rs.
6      // Beispiele:
7      //
8      // String[]      rs1  = {"AA", "BBB"};
9      // String[]      rs2  = {"BB", "CCC"};
10     // String[]      rs3  = {"AA", "AAA", "A"};
11     // ArrayList<String> als1 = new ArrayList<String>(Arrays.asList(rs1));
12     // ArrayList<String> als2 = new ArrayList<String>(Arrays.asList(rs2));
13     // ArrayList<String> als3 = new ArrayList<String>(Arrays.asList(rs3));
14     //
15     // istGroesser(als3, rs1) ist gleich true
16     // istGroesser(als1, rs2) ist gleich false
17     // istGroesser(als2, rs1) ist gleich true
18     // istGroesser(als3, rs3) ist gleich false
19     ...
20 } // istGroesser
```

**Aufgabe 2 (20 Punkte)** Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1  static int anzZiffern(int[] ri) {
2      // Wie viele Ziffern braucht man, um alle Komponenten von ri als
3      // Dezimalzahlen darzustellen? Diese Funktion liefert die Antwort.
4      // Beispiele:
5      // anzZiffern(new int[]{10, 20, 30}) ist gleich 6
6      // anzZiffern(new int[]{123, 12, 1234}) ist gleich 9
7      // anzZiffern(new int[]{0, 5, 5, 1}) ist gleich 4
8      // anzZiffern(new int[]{}) ist gleich 0
9      ...
10 } // anzZiffern
```

**Aufgabe 3 (15 Punkte):** Stellen Sie die folgende Variable `rrs` als Boje dar (wahlweise in vereinfachter oder in ausführlicher Form):

```
1  String[][] rrs = {
2      {"AB", null, "C"},
3      null,
4      {null},
5      {}
6  };
```

**Aufgabe 4 (15 Punkte):** Was gibt das folgende Programm Aufgabe04 zur Standardausgabe (zum Bildschirm) aus?

```
1 class Aufgabe04 {
2     // -----
3     static public void main(String[] _) {
4         for (int i=1; i<=2; i++) {
5             try {
6                 pln("AAA" + i);
7                 machWas(i);
8             } catch (ArithmeticException ex) {
9                 pln("BBB");
10            } finally {
11                pln("CCC");
12            }
13        }
14    }
15    // -----
16    static public void machWas(int n) {
17        try {
18            switch (n) {
19                case 1: throw new ArithmeticException("XXX");
20                case 2: throw new NumberFormatException("YYY");
21            }
22        } catch (NumberFormatException ex) {
23            pln("DDD");
24        } finally {
25            pln("EEE");
26        }
27    }
28    // -----
29    static void pln(Object ob) {System.out.println(ob);}
30    // -----
31 } // class Aufgabe04
```

**Aufgabe 5 (15 Punkte):** Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

```
1     // Schleife 5.1:
2     int n = -3;
3     for (int i=0; i<=10; i+=n) {
4         n +=2;
5         p(i + " ");
6     }
7     pln();
8
9     // Schleife 5.2:
10    int[] rei = {2, 0, 3, 1};
11    for (int i : rei) rei[i] = 3-i;
12    pln(Arrays.toString(rei));
13
14    // Schleife 5.3:
15    StringBuilder[] sbr = {
16        new StringBuilder("A+"),
17        new StringBuilder("B+"),
18        new StringBuilder("C+"),
19    };
20
21    for (int i=1; i<sbr.length; i++) {
22        sbr[i].append(sbr[i-1]);
23    }
24
25    for (StringBuilder sb : sbr) p(sb);
26    pln();
```

```
27
28      // Schleife 5.4:
29      for (char c1='A'; c1<='H'; c1+=2) {
30          for (char c2=c1; c2<='H'; c2++) {
31              p(c2);
32          }
33          pln();
34      }
```

Die Namen `p` und `pln` sind auch hier Abkürzungen für die Namen `System.out.print` und `System.out.println`.

### Aufgabe 6: (15 Punkte)

#### 6.1. Was ist ein *Modul*?

#### 6.2. Geben Sie 7 *Literale* an, je eines von jedem der folgenden 7 Typen:

`int`, `long`, `float`, `double`, `char`, `String`, `boolean`.

#### 6.3. Betrachten Sie die folgende Befehlsfolge:

```
1      int a1 = 10;
2      int a2 = 20;
3      int b1 = ++a1;
4      int b2 = a2--;
```

Welche *Werte* enthalten die Variablen `a1`, `a2`, `b1`, `b2` nachdem der Ausführer die Zeilen 1 bis 4 fertig ausgeführt hat?

6.4. Schreiben Sie eine (kleine, sehr einfache) Methode namens `gibAus17`, die den `int`-Wert 17 *ausgibt* und eine zweite Methode namens `gibZurueck17`, die den `int`-Wert 17 *zurückgibt*.

#### 6.5. Betrachten Sie die folgende Variablen-Vereinbarung:

```
5      int[][][] otto = new int[3][2][5];
```

6.5.1. Welche *Stufigkeit* hat die Reihung `otto`? (Geben Sie eine Ganzzahl wie 0 oder 17 etc. an).

6.5.2. Zu welchem *Typ* gehört `otto`? Geben Sie den Typ so an, wie man ihn *ausspricht*.

6.5.3. Zu welchem *Typ* gehören die *Komponenten* der Reihung `otto`?

6.5.4. Zu welchem *Typ* gehören die *elementaren* Komponenten der Reihung `otto`?

6.5.5. *Wie viele* elementare Komponenten enthält die Reihung `otto`?

6.6. Geben Sie für jedes der folgenden Konstrukte an, mit welchem Befehl man es beenden kann:

6.6.1. Eine *Methode*?

6.6.2. Eine *Ausführung des Rumpfes* einer Schleife?

6.6.3. Eine *Schleife*?

6.6.4. Ein *Programm*?

**Beurteilung dieser Klausur:**

A1	
A2	
A3	
A4	
A5	
A6	
Summe	
Note	
Datum	

**Korrigierte Beurteilung:**

A1	
A2	
A3	
A4	
A5	
A6	
Summe	
Note	
Datum	

**Lösung 1 (20 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:**

```
1  static public boolean istGroesser(ArrayList<String> als, String[] rs) {
2      // Liefert true genau dann wenn
3      // als mehr Komponenten enthaelt als rs   oder wenn
4      // als gleich viele Komponenten enthaelt wie rs und jede Komponente
5      // von als groesser ist als die entsprechende Komponenten von rs.
6      // Beispiele:
7      //
8      // String[]      rs1  = {"AA", "BBB"};
9      // String[]      rs2  = {"BB", "CCC"};
10     // String[]      rs3  = {"AA", "AAA", "A"};
11     // ArrayList<String> als1 = new ArrayList<String>(Arrays.asList(rs1));
12     // ArrayList<String> als2 = new ArrayList<String>(Arrays.asList(rs2));
13     // ArrayList<String> als3 = new ArrayList<String>(Arrays.asList(rs3));
14     //
15     // istGroesser(als3, rs1) ist gleich true
16     // istGroesser(als1, rs2) ist gleich false
17     // istGroesser(als2, rs1) ist gleich true
18     // istGroesser(als3, rs3) ist gleich false
19
20     if (als.size() > rs.length) return true;
21     if (als.size() < rs.length) return false;
22
23     // Strings darf man nicht mit den Operatoren <, <=, > etc. vergleichen,
24     // aber mit der Methode compareTo (siehe dazu die Dokumentation der
25     // Klasse String).
26     for (int i=0; i<rs.length; i++) {
27         if (als.get(i).compareTo(rs[i]) <= 0) return false;
28     }
29     return true;
30
31 }
```

**Lösung 2 (20 Punkte) Schreiben Sie eine Methode entsprechend den folgenden Spezifikationen:**

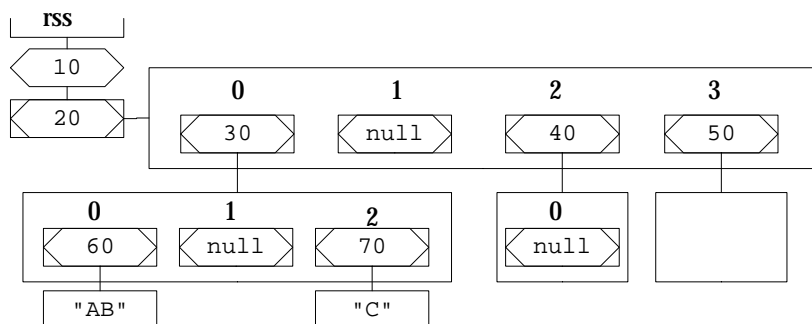
```
1  static int anzZiffern(int[] ri) {
2      // Wie viele Ziffern braucht man, um alle Komponenten von ri als
3      // Dezimalzahlen darzustellen? Diese Funktion liefert die Antwort.
4      // Beispiele:
5      // anzZiffern(new int[]{10, 20, 30})    ist gleich 6
6      // anzZiffern(new int[]{123, 12, 1234}) ist gleich 9
7      // anzZiffern(new int[]{0, 5, 5, 1})    ist gleich 4
8      // anzZiffern(new int[]{})             ist gleich 0
9
10     int anz = 0;
11
12     for (int n : ri) {
13         while (true) {
14             anz++;
15             n = n / 10;
16             if (n == 0) break;
17         }
18     }
19
20     return anz;
21 }
```

**Lösung 3 (15 Punkte):** Stellen Sie die folgende Reihung `rrs` als Boje dar:

```

1  static void aufgabe3() {
2      String[][] rrs = {
3          {"AB", null, "C"},
4          null,
5          {null},
6          {}
7      };

```



**Lösung 5 (15 Punkte):** Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

```

1      // Schleife 5.1:
2      int n = -3;
3      for (int i=0; i<=10; i+=n) {
4          n +=2;
5          p(i + " ");
6      }
7      pln();
8
9      // Schleife 5.2:
10     int[] rei = {2, 0, 3, 1};
11     for (int i : rei) rei[i] = 3-i;
12     pln(Arrays.toString(rei));
13
14     // Schleife 5.3:
15     StringBuilder[] sbr = {
16         new StringBuilder("A+"),
17         new StringBuilder("B+"),
18         new StringBuilder("C+"),
19     };
20
21     for (int i=1; i<sbr.length; i++) {
22         sbr[i].append(sbr[i-1]);
23     }
24
25     for (StringBuilder sb : sbr) p(sb);
26     pln();
27
28     // Schleife 5.4:
29     for (char c1='A'; c1<='H'; c1+=2) {
30         for (char c2=c1; c2<='H'; c2++) {
31             p(c2);
32         }
33         pln();
34     }

```

**Die Ausgaben der Schleifen:**

```

// Schleife 5.1: 0 -1 0 3 8
// Schleife 5.2: [3, 2, 1, 1]
// Schleife 5.3: A+B+A+C+B+A+
// Schleife 5.4:
ABCDEFGH
CDEFGH
EFGH
GH

```

**Lösung 6: (15 Punkte)****6.1. Was ist ein *Modul*?**

Ein Modul ist ein Behälter für Variablen, Unterprogramme, Typen, Module, ... etc., der aus mindestens 2 Teilen besteht, einem öffentlichen Teil und einem privaten Teil.

**6.2. Geben Sie 7 *Literale* an, je eines von jedem der folgenden 7 Typen:**

int, long, float, double, char, String, boolean.

123, 456L, 12.3F, 45.6, 'X', "Hallo", true

**6.3. Betrachten Sie die folgende Befehlsfolge:**

```
1      int a1 = 10;
2      int a2 = 20;
3      int b1 = ++a1;
4      int b2 = a2--;
```

Welche *Werte* enthalten die Variablen a1, a2, b1, b2 nachdem der Ausführer die Zeilen 1 bis 4 fertig ausgeführt hat?

11, 19, 11, 20

**6.4. Schreiben Sie eine (kleine, sehr einfache) Methode namens gibAus17, die den int-Wert 17 *ausgibt* und eine zweite Methode namens gibZurueck17, die den int-Wert 17 *zurückgibt*.**

```
5      void gibAus17()      {println(17);}
6      int  gibZurueck17(){return 17;}
```

**6.5. Betrachten Sie die folgende Variablen-Vereinbarung:**

```
7      int[][][] otto = new int[3][2][5];
```

**6.5.1. Welche *Stufigkeit* hat die Reihung otto? (Geben Sie nur eine Ganzzahl wie 0 oder 17 etc. an).**

**Die Reihung otto hat die Stufigkeit 3.**

**6.5.2. Zu welchem *Typ* gehört otto? Geben Sie den Typ so an, wie man ihn ausspricht.**

**Zum Typ Reihung von Reihungen von Reihungen von int (-Variablen).**

**6.5.3. Zu welchem *Typ* gehören die *Komponenten* der Reihung otto?**

**Zum Typ Reihung von Reihungen von int (-Variablen).**

**6.5.4. Zu welchem *Typ* gehören die *elementaren* Komponenten der Reihung otto?**

**Zum Typ int.**

**6.5.5. *Wie viele* elementare Komponenten enthält die Reihung otto?**

**Die Reihung otto enthält (3 x 2 x 5 gleich) 30 elementare Komponenten.**

**6.6. Geben Sie für jedes der folgenden Konstrukte an, mit welchem Befehl man es beenden kann:**

- |   |                |
|---|----------------|
| 6.6.1. Eine <i>Methode</i> ?                              | return         |
| 6.6.2. Eine <i>Ausführung des Rumpfes</i> einer Schleife? | continue       |
| 6.6.3. Eine <i>Schleife</i> ?                             | break          |
| 6.6.4. Ein <i>Programm</i> ?                              | System.exit(3) |