

Vorname

Nachname

Matrikel-Nr

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

Schreiben Sie jede Lösung auf die Vorderseite eines *neuen Blattes* (und lassen Sie die Rückseiten Ihrer Lösungsblätter *leer*). Die Aufgaben 5 und 6 stehen *auf der Rückseite* dieses Blattes!

Aufgabe 1 (18 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```
1 static public String[] kopMal2(String[] sr) {
2     // Liefert eine Art Kopie von sr, in der aber "jedes Zeichen
3     // verdoppelt wurde". Beispiele:
4
5     // String[] sr01 = {"ABC", "C", "DE"};
6     // String[] sr02 = {"F", "", "G"};
7     // String[] sr03 = {"", "", ""};
8     // String[] sr04 = {};
9     //
10    // kopMal2(sr01) ist gleich {"AABBCC", "CC", "DDEE"}
11    // kopMal2(sr02) ist gleich {"FF", "", "GG"}
12    // kopMal2(sr03) ist gleich {"", "", ""}
13    // kopMal2(sr04) ist gleich {}
14    ...
15 } // kopMal2
```

Aufgabe 2 (16 Punkte): Betrachten Sie die folgenden Vereinbarung:

```
1 String s01 = "Hallo";
2 String s02 = "Hallo";
3 String s03 = new String("Hallo");
4
5 String[] rs = {s01, s02, "Hallo", s03, null, ""};
```

2.1. Stellen Sie die Reihung `rs` als Boje dar.

2.2. Beantworten Sie die folgenden Fragen (möglichst kurz):

Frage 2.2.1: Welchen Wert hat der Ausdruck `rs[0]==rs[1]`?

Frage 2.2.2: Welchen Wert hat der Ausdruck `rs[1]==rs[2]`?

Frage 2.2.3: Welchen Wert hat der Ausdruck `rs[2]==rs[3]`?

Aufgabe 3 (18 Punkte) Schreiben Sie zwei Methoden entsprechend den folgenden Spezifikationen:

```
1 static public char intNachChar(int n) {
2     // Verlaesst sich darauf, dass der Parameter n einen der 21 Werte
3     // 0, 1, 2, ..., 20 hat.
4     // Liefert in Abhaengigkeit von n eines der folgenden 21 Zeichen:
5     // '0', '1', '2', ... '9', 'A', 'B', 'C', ... 'J', 'K'
6     // Beispiele:
7     // intNachChar( 0) ist '0'
8     // intNachChar( 1) ist '1'
9     // intNachChar( 9) ist '9'
10    // intNachChar(10) ist 'A'
11    // intNachChar(11) ist 'B'
12    // intNachCahr(20) ist 'K'
13    //
14    // Der Rumpf dieser Methode darf nur Vereinbarungen und einfache
15    // Anweisungen enthalten.
16    // Tip: Sie duerfen eine Reihung vereinbaren und mit n als Index
17    // darauf zugreifen.
18    ...
19 } // intNachChar
20
```

```

21 static public String alsBerZahl(int zahl, short B) {
22     // Wandelt die zahl in das Zahlensystem mit der Basis B um und
23     // liefert diese B-er-Zahl als String.
24     // Falls B nicht zwisch 2 und 20 (einschliesslich) liegt, wird
25     // der String "Falsche Basis" als Ergebnis geliefert.
26     // Falls die zahl negativ ist, wird der String "Zahl ist negativ"
27     // als Ergebnis geliefert.
28     //
29     // Beispiele: Seien folgende short-Variablen vereinbart:
30     // short b01 = 1; // Eine zu kleine Basis
31     // short b02 = 2; // Die kleinste zulaessige Basis
32     // short b10 = 10;
33     // short b13 = 13;
34     // short b20 = 20; // Die groesste zulaessige Basis
35     // short b21 = 21; // Eine zu grosse Basis
36     //
37     // Damit gilt:
38     // alsBerZahl(123, b10) ist gleich "123"
39     // alsBerZahl( 5, b02) ist gleich "101"
40     // alsBerZahl( 15, b13) ist gleich "12"
41     // alsBerZahl( 26, b13) ist gleich "20"
42     // alsBerZahl( 16, b16) ist gleich "10"
43     // alsBerZahl(123, b01) ist gleich "Falsche Basis"
44     // alsBerZahl(123, b21) ist gleich "Falsche Basis"
45     // alsBerZahl(-12, b10) ist gleich "Zahl ist negativ"
46     ...
47 } // alsBerZahl

```

Natürlich sollen Sie in der Methode `alsBerZahl` die Methode `intNachChar` aufrufen!

Aufgabe 4 (16 Punkte): Betrachten Sie die folgenden Klassenvereinbarungen:

```

1 class K1 {
2     static int sum = 0;
3     int zal1;
4     int zal2;
5
6     public K1(int zal1, int zal2) {
7         this.zal1 = zal1;
8         this.zal2 = zal2;
9         sum += 1;
10    }
11 } // class K1
12
13 class K2 {
14     static int sum1 = 0;
15     static int sum2 = 0;
16     int zal;
17
18     public K2(int zal) {
19         this.zal = zal;
20         sum1 += 2;
21         sum2 += 3;
22    }
23 } // class K2
24
25 class K3 {
26     static public void main(String[] _) {
27         K1 ob11 = new K1(10, 20);
28         K1 ob12 = new K1(30, 40);
29         K2 ob21 = new K2(50);
30         ...
31         K1 ob13 = new K1(60, 70);
32         K2 ob22 = new K2(80);
33     } // main
34 } // class K3

```

4.1. Sei M1 der Moment, in dem der Ausführer die Zeile 29 fertig ausgeführt hat. Wie viele Module existieren in diesem Moment M1 und wie heißen diese Module?

4.2. Wie viele `int`-Variablen existieren im Moment M1 und wie heißen diese Variablen mit vollen Namen?

4.3. Sei M2 der Moment, in dem der Ausführer die Zeile 32 fertig ausgeführt hat. Wie viele Module wurden nach dem Moment M1 und vor dem Moment M2 erzeugt und wie heißen diese Module?

4.4. Wie viele `int`-Variablen wurden nach dem Moment M1 und vor dem Moment M2 erzeugt und wie heißen diese Variablen mit vollen Namen?

Aufgabe 5 (16 Punkte): Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

```

1      // Schleife 5.1.:
2      for (char c1='0'; c1<'2'; c1++) {
3          for (char c2='C'; c2>='A'; c2--) {
4              p(" " + c2 + c1 + " ");
5          }
6          pln();
7      }
8
9      // Schleife 5.2:
10     for (int i=1; i<=3; i++) {
11         switch (i) {
12             case 2 : p("A");
13             default : p("B");
14             case 1 : p("C"); break;
15         }
16     }
17     pln();
18
19     // Schleife 5.3:
20     int a = 1;
21     int b = 2;
22     int c = 3;
23     for (int i=1; i<=5; i++) {
24         pln(a + b + c);
25         a++;
26         b += a;
27         c += a + b;
28     }
29
30     // Schleife 5.4:
31     for (int i=1; i<=3; i++) {
32         for (int j=3-i; j>=1; j--) p("A");
33         for (int j=1; j<=i+(i-1); j++) p("B");
34         for (int j=3-i; j>=1; j--) p("A");
35         pln();
36     }

```

Die Prozedur `p` gibt (wie gewohnt) ihren Parameter zum Bildschirm aus, *ohne* den Bildschirmzeiger (engl. cursor) zum Anfang der nächsten Zeile vorzuschieben. Die parameterlose Prozedur `pln` schiebt den Bildschirmzeiger zum Anfang der nächsten Zeile vor.

Aufgabe 6: (16 Punkte)

6.1. Betrachten Sie die folgende Regel einer kontextfreien Grammatik:

FloatLit : Digits
 ("." [Digits] [Exponent] ["F"]
 | Exponent ["F"]
 | "F"
)
 | "." Digits [Exponent] ["F"]

Welche der folgenden Symbolfolgen sind mit dieser Regel aus dem Zwischensymbol FloatLit ableitbar und welche sind nicht ableitbar?

6.1.1. "." Digits

6.1.2. Digits Exponent Digits

6.1.3. Digits "." Digits

6.1.4. Digits

6.1.5. Digits "." Digits Exponent "F"

6.2. Wir haben grundsätzlich drei Arten von Befehlen unterschieden. Wie heißen diese Befehlsarten auf Deutsch und auf Englisch?

6.3. Geben von jedem der folgenden Java-Befehle an, zu welcher Befehlsart (siehe 6.2.) er gehört:

6.3.1. `double d01 = 34.5;`

6.3.2. `System.out.println(34.5);`

6.3.3. `double d02() {return 1.0;}`

6.3.4. `d01 == 67.8`

6.3.5. `d01 = 67.8;`

6.4. Übersetzen Sie den Befehl 6.3.1. ins Deutsche.

6.5. Übersetzen Sie den Befehl 6.3.4. ins Deutsche.

6.6. Wir haben 2 Arten von Methoden unterschieden. Wie heißen diese beiden Methodenarten? Beschreiben Sie einen wichtigen Unterschied zwischen Methoden der einen und der anderen Art.

Lösung 1 (18 Punkte): Schreiben Sie eine Methode entsprechend der folgenden Spezifikation:

```

1  static public String[] kopMal2(String[] sr) {
2      // Liefert eine Art Kopie von sr, in der aber "jedes Zeichen
3      // verdoppelt wurde".
4
5      String[] erg = new String[sr.length];
6      int e = 0;
7
8      for (String s : sr) {
9          StringBuilder sb = new StringBuilder();
10         for (int i=0; i<s.length(); i++) {
11             char c = s.charAt(i);
12             sb.append(c);
13             sb.append(c);
14         }
15         erg[e++] = sb.toString();
16     }
17     return erg;
18 } // kopMal2

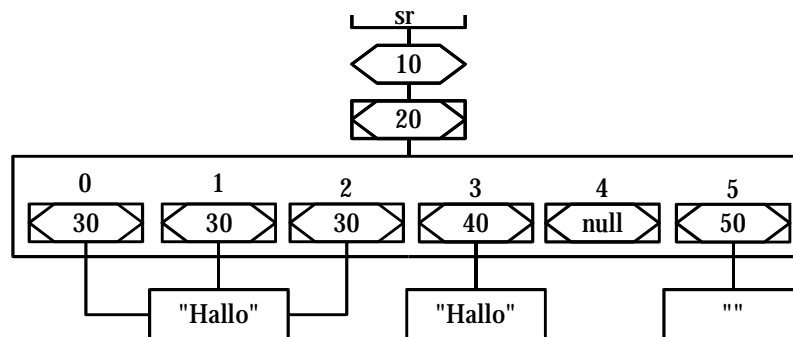
```

Lösung 2 (16 Punkte): Betrachten Sie die folgenden Vereinbarung:

```

1  String s01 = "Hallo";
2  String s02 = "Hallo";
3  String s03 = new String("Hallo");
4
5  String[] rs = {s01, s02, "Hallo", s03, null, ""};

```

3.1. Stellen Sie die Reihung rs als Boje dar.**3.2. Beantworten Sie die folgenden Fragen (möglichst kurz):**Frage 3.2.1: Welchen Wert hat der Ausdruck `rs[0]==rs[1]`? **true**Frage 3.2.2: Welchen Wert hat der Ausdruck `rs[1]==rs[2]`? **true**Frage 3.2.3: Welchen Wert hat der Ausdruck `rs[2]==rs[3]`? **false****Lösung 3 (18 Punkte) Schreiben Sie zwei Methoden entsprechend den folgenden Spezifikationen:**

```

1  static public char intNachChar(int n) {
2      // Verlaesst sich darauf, dass der Parameter n einen der 21 Werte
3      // 0, 1, 2, ..., 20 hat.
4      // Liefert in Abhaengigkeit von n eines der folgenden 21 Zeichen:
5      // '0', '1', '2', ... '9', 'A', 'B', 'C', ... 'I', 'J'
6
7      final char[] tab =
8          {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
9           'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K'}
10         };
11
12     return tab[n];
13 } // intNachChar
14

```

```
15 static public String alsBerZahl(int zahl, short B) {
16     // Wandelt die zahl in das Zahlensystem mit der Basis B um und
17     // liefert diese B-er-Zahl als String.
18     // Falls B nicht zwisch 2 und 20 (einschliesslich) liegt, wird
19     // der String "Falsche Basis" als Ergebnis geliefert.
20     // Falls die zahl negativ ist, wird der String "Zahl ist negativ"
21     // als Ergebnis geliefert.
22
23     if (B < 2 || 20 < B) return "Falsche Basis: " + B;
24     if (zahl < 0) return "Zahl ist negativ:" + zahl;
25
26     int ziffInt;
27     char ziffChar;
28     StringBuilder erg = new StringBuilder();
29
30     while (true) {
31         ziffInt = zahl % B;
32         zahl = zahl / B;
33         ziffChar = intNachChar(ziffInt);
34         erg.insert(0, ziffChar);
35         if (zahl == 0) break;
36     }
37
38     return erg.toString();
39 } // alsBerZahl
```

Lösung 4 (16 Punkte): Betrachten Sie die folgenden Klassenvereinbarungen:

```
1 class K1 {
2     static private int sum = 0;
3     private int zal1;
4     private int zal2;
5
6     public K1(int zal1, int zal2) {
7         this.zal1 = zal1;
8         this.zal2 = zal2;
9         sum += zal1;
10        sum += zal2;
11    }
12
13    public K1() {}
14 } // class K1
15
16 class K2 {
17     static private int sum1 = 0;
18     static private int sum2 = 0;
19     private int zal;
20
21     public K2(int zal) {
22         this.zal = zal;
23         sum1 += zal;
24         sum2 += zal;
25     }
26 } // class K2
27
28 class K3 {
29     static public void main(String[] _) {
30         K1 ob11 = new K1(10, 20);
31         K1 ob12 = new K1(30, 40);
32         K2 ob21 = new K2(50);
33         ...
34         K1 ob13 = new K1(60, 70);
35         K2 ob22 = new K2(80);
36     } // main
37 } // class K3
```

4.1. Sei M1 der Moment, in dem der Ausführer die Zeile 32 fertig ausgeführt hat. Wie viele Module existieren in diesem Moment M1 und wie heißen diese Module?

6 Module:

3 Klassen (K1, K2, K3) und 3 Objekte (ob11, ob12, ob21)

4.2. Wie viele `int`-Variablen existieren im Moment M1 und wie heißen diese Variablen mit vollen Namen?

8 int-Variablen:

K1.sum, K2.sum1, K2.sum2

ob11.zal1, ob11.zal2

ob12.zal1, ob12.zal2,

ob21.zal

4.3. Sei M2 der Moment, in dem der Ausführer die Zeile 35 fertig ausgeführt hat. Wie viele Module wurden nach dem Moment M1 und vor dem Moment M2 erzeugt und wie heißen diese Module?

2 Module (die Objekte ob13 und ob22)

4.4. Wie viele `int`-Variablen wurden nach dem Moment M1 und vor dem Moment M2 erzeugt und wie heißen diese Variablen mit vollen Namen?

3 int-Variablen (ob13.zal1, ob13.zal2, ob22.zal)

Lösung 5 (16 Punkte): Geben Sie von jeder der folgenden 4 Schleifen an, was sie zum Bildschirm ausgibt.

Die Ausgabe der Schleife 5.1:

C0 B0 A0

C1 B1 A1

Die Ausgabe der Schleife 5.2:

CABCBC

Die Ausgabe der Schleife 5.3:

6

15

29

49

76

Die Ausgabe der Schleife 5.4:

AABAA

ABBBB

BBBBB

Lösung 6: (16 Punkte)

6.1. Betrachten Sie die folgende Regel einer kontextfreien Grammatik:

```
FloatLit :  Digits
           ( "." [Digits] [Exponent] ["F"]
            | Exponent ["F"]
            | "F"
            )
           | "." Digits [Exponent] ["F"]
```

Welche der folgenden Symbolfolgen sind mit dieser Regel aus dem Zwischensymbol FloatLit ableitbar und welche sind nicht ableitbar?

- | | | |
|--------|--------------------------------|-----------------|
| 6.1.1. | "." Digits | ableitbar |
| 6.1.2. | Digits Exponent Digits | nicht ableitbar |
| 6.1.3. | Digits "." Digits | ableitbar |
| 6.1.4. | Digits | nicht ableitbar |
| 6.1.5. | Digits "." Digits Exponent "F" | ableitbar |

6.2. Wir haben grundsätzlich drei Arten von Befehlen unterschieden. Wie heißen diese Befehlsarten auf Deutsch und auf Englisch?

Vereinbarungen (declarations), Ausdrücke (expressions) und Anweisungen (statements)

6.3. Geben von jedem der folgenden Java-Befehle an, zu welcher Befehlsart (siehe 6.2.) er gehört:

- | | | |
|--------|---|--|
| 6.3.1. | <code>double d01 = 34.5;</code> | Vereinbarung (einer Variablen) |
| 6.3.2. | <code>System.out.println(34.5);</code> | Anweisung (Aufruf einer Prozedur) |
| 6.3.3. | <code>double d02() {return 1.0;}</code> | Vereinbarung (einer Methode) |
| 6.3.4. | <code>d01 == 67.8</code> | Ausdruck (vom Typ boolean) |
| 6.3.5. | <code>d01 = 67.8;</code> | Anweisung (eine Zuweisung) |

6.4. Übersetzen Sie den Befehl 6.3.1. ins Deutsche.

Erzeuge eine Variable namens d01 vom Typ double mit dem Anfangswert 34.5.

6.5. Übersetzen Sie den Befehl 6.3.4. ins Deutsche.

Berechne den Wert des Ausdrucks d01 == 67.8.

6.6. Wir haben 2 Arten von Methoden unterschieden. Wie heißen diese beiden Methodenarten? Beschreiben Sie einen wichtigen Unterschied zwischen Methoden der einen und der anderen Art.

Wir haben Prozeduren und Methoden unterschieden.

Jeder Aufruf einer Prozedur ist eine Anweisung.

Jeder Aufruf einer Funktion ist ein Ausdruck.