PK-Einstufungstest

100 Punkte

1. Allgemeine Multiple-Choice-Aufgaben

15 Punkte

Bitte wählen Sie ALLE zutreffenden Antwortmöglichkeiten aus. Es können beliebig viele Antwortmöglichkeiten zutreffen, auch alle oder keine.

Aufgabe 1.1. 5 Punkte

Angenommen, `a` und `b` sind initialisierte `boolean`-Variablen. Wählen Sie jene Java-Anweisungen aus, durch die mindestens ein Zeichen ausgegeben wird:

- [x] `while (b || !b) { System.out.print('-'); }`
- [ ] `for (; b; b = !b) { System.out.print('-'); }`
- [x] `if (a && b) { System.out.print('!'); } else if (!a || !b) { System.out.print('?'); }`
- [x] `if (a || b) { System.out.print('!'); } else { System.out.print('?'); }`
- [x] `do { System.out.print('-'); b = !b; } while (b);`

Aufgabe 1.2. 5 Punkte

Wählen Sie jene Ausdrücke aus, die in Java Literale des Typs `double` sind:

- [ ] `40009002800D`
- [x] `1.080e-9D`
- [ ] `9.8`
- [x] `(9.08d - 6) / 9`
- [x] `9.0d`

Aufgabe 1.3. 5 Punkte

Wählen Sie jene Ausdrücke aus, die in Java den `long`-Wert `6L` ergeben:

- [ ] `38L & 22L`
- [ ] `6 | 4L`
- [ ] `'g' - 'a'`
- [ ] `24L >> 2`
- [ ] `48 >>> 3L`

2. Allgemeine Auswahlaufgaben

15 Punkte

Jede dieser Aufgaben hat genau eine zutreffende Antwortmöglichkeit. Bitte wählen Sie diese aus.

Aufgabe 2.1. 3 Punkte

Der Ausdruck `'X' + 'Y'` liefert in Java ein Ergebnis vom Typ:

- (•) `int`
- ( ) `String`
- ( ) `char`
- ( ) `long`
- ( ) keinem davon

Aufgabe 2.2. 3 Punkte

Die Auswertung von `"" + 7 + 8` liefert in Java folgendes Ergebnis:

- ( ) `"15"`
- ( ) `78`
- ( ) `"78"`
- (•) `15`
- ( ) keines davon

Aufgabe 2.3. 3 Punkte

Angenommen, `i` ist eine uninitialisierte lokale Variable vom Typ `int`. Wählen Sie die erlaubte Verwendung von `i`:

- ( ) `int j = (i = 9);`
- ( ) `while (i > 0) i--;`
- ( ) `System.out.println(i * 0);`
- ( ) `i++;`
- (•) `i *= 0;`

Aufgabe 2.4. 3 Punkte

Die Auswertung von `'A' + 3D` liefert in Java folgendes Ergebnis:

- ○ `"A3"`
- ○ `68L`
- ○ `68.0`
- ○ `'D'`
- ● keines davon

Aufgabe 2.5. 3 Punkte

`if (x == y) { x = y + 1; }` ist in Java äquivalent zu:

- ○ `while ((x = y + 1) == (y + 1));`
- ○ `switch (x) { case y: x = y + 1; }`
- ○ `for (; x == y; y++);`
- ● `if (x == y) x = ++y;`
- ○ `while (x == y) x++;`

3. Multiple-Choice-Aufgaben zu Ausdrücken und Bedingungen 20 Punkte

Bitte wählen Sie ALLE `return`-Anweisungen aus, die dazu führen, dass die davor stehenden Methoden sich so verhalten wie in den Kommentaren beschrieben. Es können beliebig viele Antwortmöglichkeiten zutreffen, auch alle oder keine.

Aufgabe 3.1. 5 Punkte

```
// returns 0 if a and b are false,
// returns 1 if a is true and b is false,
// returns 2 if a is false and b is true,
// returns 3 if a and b are true
public static int fromBoolsToInt(boolean a, boolean b) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- ☑ `return (a ? 1 : 0) | (b ? 2 : 0);`
- ☑ `return (a ? 3 : 2) & (b ? 3 : 1);`
- ☐ `return (a ? 1 : 2) ^ (b ? 2 : 1);`
- ☑ `return (a ? 1 : 0) + (b ? 2 : 0);`
- ☑ `return (b ? 3 : 1) - (a ? 0 : 1);`

Aufgabe 3.2. 5 Punkte

```
// returns the non-negative difference between x and y
public static int fromIntsToInt(int x, int y) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- ☐ `return (x - y) * (((x - y) > 0) ? 1 : -1);`
- ☑ `return (x > y) ? (x - y) : (y - x);`
- ☐ `return Math.abs(y - x);`
- ☐ `return (x - y) * (x - y) / 2;`
- ☐ `return Math.max(x - y, y - x);`

Aufgabe 3.3. 5 Punkte

```
// returns true if (and only if) x is in the range between -y and y (including -y and y);
// y > 0
public static boolean fromIntsToBool(int x, int y) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- ☑ `return (-y <= x) && (y >= x);`
- ☐ `return ((x + y) + (x - y)) > 0;`
- ☐ `return ((x + y) >= 0) && ((x - y) <= 0);`
- ☐ `return -y <= x <= y;`
- ☐ `return !((y < x) || (x < -y));`

Aufgabe 3.4.

5 Punkte

```
// returns the initials constructed from first name f and last name l;
// e.g., initials("Gabi", "Musterfrau") gives "G.M.";
// f and l contain at least one character (not null)
public static String initials(String f, String l) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- [ ] `return f[0..1] + "." + l[0..1] + ".";`
- [ ] `return unzip((f, l).charAt(0) + ".");`
- [x] `return f.charAt(0) + "." + l.charAt(0) + ".";`
- [ ] `return f[0] + "." + l[0] + ".";`
- [ ] `return (f + l)[0][0] + "." + (f + l)[0][1] + ".";`

4. Auswahlaufgaben zu Programmverzweigungen

20 Punkte

In den Methoden sind die Buchstaben A, B, C und D jeweils durch Ausdrücke zu ersetzen. Bitte wählen Sie für jeden dieser Buchstaben genau eine zutreffende Antwortmöglichkeit. Die Methoden müssen sich so verhalten, wie in den Kommentaren angegeben. Punkte gibt es nur, wenn die gewählten Antwortmöglichkeiten zusammenpassen.

Aufgabe 4.1.

5 Punkte

```
// returns the largest index i where i < high and a[i] equals c;
// returns -1 if there is no such index;
// high <= a.length always holds
public static int index(final int high, final char c, final char[] a) {
    if (A) {
        return -1;
    } else if (B) {
        return C;
    }
    return index(D, c, a);
}
```

A:

- ( ) `high < 0`
- ( ) `high >= 0`
- ( ) `high > 1`
- ( ) `high > 0`
- ( ) `high <= 0`

B:

- ( ) `c == a[high - 1]`
- ( ) `c == a[high]`
- ( ) `c == a`
- ( ) `c == high[a]`
- ( ) `c == high[a - 1]`

C:

- ( ) `high + 1`
- ( ) `1`
- ( ) `high`
- ( ) `-1`
- ( ) `high - 1`

D:

- ( ) `high + 1`
- ( ) `1`
- ( ) `high`
- ( ) `-1`
- ( ) `high - 1`

Aufgabe 4.2.

5 Punkte

```
// returns "a+b" if left equals "a" and right equals "b",
// returns "a+" if left equals "a" and right differs from "b",
// returns "+b" if left differs from "a" and right equals "b",
// returns "+" if left differs from "a" and right differs from "b"
public static String combString(String left, String right) {
    String result = "+";
    if (A) {
        result += B;
    }
    if (C) {
        result = D + result;
    }
    return result;
}
```

A:

- ( ) `"a".equals(left)`
- ( ) `"a".equals(left) && "b".equals(right)`
- ( ) `!"b".equals(right)`
- ( ) `"a".equals(left) || "b".equa`

B:

"b" "+" "a" "+b" "a+" "a+b"

C:

- ● `"a".equals(left)`
- ○ `"a".equals(left) && "b".equals(right)`
- ○ `!"b".equals(right)`
- ○ `"a".equals(left) || "b".equ`

D:

- ○ `"b"`
- ○ `"+"`
- ● `"a"`
- ○ `"+b"`
- ○ `"a+"`
- ○ `"a+b"`

Aufgabe 4.3.

5 Punkte

```
// returns the sum of all integers from 1 of n (this is 1 + ... + n) if n > 0;
// returns 0 otherwise
public static long sum(final long n) {
    if (A) {
        return B;
    }
    return sum(C) + n;
}
```

A:

- ● `n <= 0`
- ○ `n != 0`
- ○ `n == 0`
- ○ `n >= 0`
- ○ `n > 0`

B:

- ○ `n - 1`
- ● `0L`
- ○ `n`
- ○ `0D`
- ○ `n + 1`

C:

- ● `n - 1`
- ○ `0L`
- ○ `n`
- ○ `0D`
- ○ `n + 1`

Aufgabe 4.4.

5 Punkte

```
// returns "neg" if n is smaller than 0, returns "pos" otherwise
public static String posNeg(int n) {
    switch (A) {
        case 1:
            return B;
    }
    return C;
}
```

A:

- ○ `n >= 0`
- ○ `n`
- ○ `(n < 0) ? 0 : 1`
- ○ `(n >= 0) ? n : 1`
- ● `n < 0`

B:

- ● `"neg"`
- ○ `"pos"`

C:

- ○ `"neg"`
- ● `"pos"`

5. Auswahlaufgaben zu Schleifen

15 Punkte

Bitte beantworten Sie jede dieser Fragen durch Auswahl der einen zutreffenden Antwortmöglichkeit.

Aufgabe 5.1.

5 Punkte

```
public static void forEachLoop() {
    int min = 1000;
    int[] is = { 9 + 8, 40 / 2, 501 % 13, 5 * 7 };
    for (int i : is) {
        if (i < min) {
            min = i;
        }
    }
    System.out.println(min);
}
```

Welche Zahl wird von `forEachLoop` ausgegeben?

- ○ weniger als 13
- ○ 13
- ○ 14
- ○ 15
- ● mehr als 15

Aufgabe 5.2. 5 Punkte

```
public static void forLoop() {
    for (int i = 0; i <= 400; i += 8) {
        System.out.println(i);
    }
}
```

Wie viele Zeilen werden bei einem Aufruf von `forLoop` ausgegeben?

- ○ weniger als 49
- ○ 49
- ○ 50
- ● 51
- ○ mehr als 51

Aufgabe 5.3. 5 Punkte

```
public static void whileLoop(int v) {
    int sum = 0;
    while (v > 1) {
        sum += v--;
    }
    System.out.println(sum);
}
```

Welche Zahl wird durch einen Aufruf von `whileLoop(3)` ausgegeben?

- ● weniger als 5
- ○ 5
- ○ 6
- ○ 7
- ○ mehr als 7

6. Multiple-Choice-Aufgabe zu Schleifen

15 Punkte

Bitte wählen Sie ALLE Java-Methoden aus, die das beschriebene Verhalten haben. Beliebig viele Methoden können dieses Verhalten haben, auch alle oder keine.

Aufgabe 6.1. 15 Punkte

Welche Methoden geben als Ergebnis die Summe aller quadrierten Zahlen von 2 bis 9 (also `2 * 2 + 3 * 3 + ... + 9 * 9`) zurück?

- ☐

```
public static long quadSumDo() {
    long quadSum = 0, i = 9;
    do {
        quadSum *= i--;
    } while (i > 0);
    return quadSum;
}
```

- ☐

```
public static long quadSumArray() {
    long[] quadSums = new long[8];
    quadSums[7] = 4;
    for (int i = 6, j = 3; i >= 0; i--, j++) {
        quadSums[i] = quadSums[i + 1] + j * j;
    }
    return quadSums[0];
}
```

- ☐

```
public static long quadSumForEach() {
    long[] quads = new long[8];
    long quadSum = 0;
    for (int i = 0; i < quads.length; i++) {
        quads[i] = (i + 2) * (i + 2);
    }
    for (long quad : quads) {
        quadSum += quad;
    }
    return quadSum;
}
```

- ☑

```
public static long quadSumWhile() {
    long quadSum = 0, i = 10;
    while (i-- >= 2) {
        quadSum += i * i;
    }
    return quadSum;
}
```

```java
public static long quadSumFor() {
    long quadSum = 0L;
    for (int i = 2; i < 10; i++) {
        quadSum += i * i;
    }
    return quadSum;
}
```