PK-Einstufungstest

69 / 100 Punkte

Alle Aufgaben beziehen sich auf Java.

1. Allgemeine Multiple-Choice-Aufgaben

11 / 15 Punkte

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

Aufgabe 1.1. 4 / 5 Punkte

Angenommen, `a` und `b` sind initialisierte `boolean`-Variablen. Wählen Sie jene Java-Anweisungen aus, durch die es (mit bestimmten Variablenwerten) möglich ist, dass die Ausgabe `xy` (ohne weitere Ausgaben) erzeugt wird:

- [ ]
```
do {
    if(a) {
        System.out.print("y");
    }
    a = !a;
    System.out.print("x");
} while (a || b);
```

- [ ]
```
for(b = !a; a != b; a = !a) {
    System.out.print("x");
}
System.out.print("y");
```

- [x]
```
if(!a) {
    System.out.print("x");
    a = !a;
}
if (a) {
    System.out.print("y");
}
```

- [x]
```
if(!a) {
    System.out.print("x");
} else if (a) {
    System.out.print("y");
}
System.out.print("y");
```

- [ ]
```
while(a) {
    a = !a;
    System.out.print("x");
}
while(a) {
    a = !a;
    System.out.println("y");
}
```

Aufgabe 1.2. 5 / 5 Punkte

Angenommen, `x` ist eine initialisierte `int`-Variable. Wählen Sie jene Ausdrücke aus, die in Java zu `++x + x` hinsichtlich Seiteneffekten und Ergebnissen äquivalent sind:

- [ ] `x + x + 1`
- [x] `(x += 1) + x`
- [x] `2 * (++x)`
- [ ] `2 * (x++) + 1`
- [ ] `(x + 1) * 2`

Aufgabe 1.3. 2 / 5 Punkte

Wählen Sie jene Variablendeklarationen mit Initialisierungen aus, für die der Java-Compiler keine Fehlermeldung liefert:

- [x] `int i = 0xD * 0;`
- [ ] `long x = '\2';`
- [x] `long x = 0xEF + 0xL;`
- [ ] `float f = f + 0f;`
- [ ] `char c = 00;`

2. Allgemeine Auswahlaufgaben

12 / 15 Punkte

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

Aufgabe 2.1.

3 / 3 Punkte

Die Auswertung von `('1' + "2") + 3 + 4` liefert in Java folgendes Ergebnis:

- ○ `10`
- ● `"1234"`
- ○ `1234`
- ○ `"127"`
- ○ keines davon

Aufgabe 2.2.

0 / 3 Punkte

`for (int i = n; i != 0;) { f(i/=2); }` ist in Java äquivalent zu:

- ● `for (int i = n; i > 0; i /= 2) { f(i); }`
- ○ `{ int i; do { i = 1; f(i); } while ((i /= 2) != 0); };`
- ○ `for (int i = 1; i <= n; i *= 2) { f(i); };`
- ○ `{ int i; for (i = n; (i /= 2) != 0; ) { f(i); } }`
- ○ `{ int i = n; while (i != 0) { i = i / 2; f(i); } }`

Aufgabe 2.3.

3 / 3 Punkte

Der Ausdruck `0xb00C + (1f + '0')` liefert in Java ein Ergebnis vom Typ:

- ● `float`
- ○ `int`
- ○ `long`
- ○ `char`
- ○ `double`

Aufgabe 2.4.

3 / 3 Punkte

Die Auswertung von `0.0 + (char)('a' + 0.6)` liefert in Java folgendes Ergebnis (wobei 97 der ASCII-Wert von `'a'` ist):

- ○ `'b'`
- ○ `98.0`
- ○ `'a'`
- ● `97.0`
- ○ keines davon

Aufgabe 2.5.

3 / 3 Punkte

Angenommen, `i` ist eine uninitialisierte lokale Variable vom Typ `int`. Wählen Sie jene Anweisung bzw. Anweisungsfolge aus, für die der Java-Compiler keine Fehlermeldung liefert:

- ○ `i = 1; int i = 2;`
- ○ `i += -1; i = 2;`
- ○ `while (i < 0) i++;`
- ○ `System.out.println(i);`
- ● `for(i = 1; i <= 10; i++) i++;`

3. Multiple-Choice-Aufgaben zu Ausdrücken und Bedingungen

16 / 20 Punkte

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

Aufgabe 3.1.

3 / 5 Punkte

```
// liefert true dann und nur dann wenn x im Intervall -y bis y
// (einschließlich -y und y) liegt;
// Annahme: y > 0
public static boolean fromIntsToBool(int x, int y) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

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

Aufgabe 3.2.

5 / 5 Punkte

```
// liefert true, wenn das letzte Zeichen des letzten Elements aus array dem Zeic
// z.B. last(new String[] {"raj","penny","leonard"}, 'd') liefert true;
// Es gilt: array != null und das letzte Element ist nicht null.
public static boolean last(String[] array, char c) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- [x] `return array[array.length-1].endsWith(""+c);`
- [x] `return array[array.length-1].charAt(array[array.length-1].length()-1) == c;`
- [ ] `return array[array.length-1].charAt(array[0].length()-1) == c;`
- [ ] `return array[array.length()-1].charAt(array[array.length()-1].length-1) == c;`
- [ ] `return array[array.length-1].getChar(array[array.length-1].length-1) == c;`

Aufgabe 3.3.

3 / 5 Punkte

```java
// gibt x / y zurück, wenn gilt: x >= y
// gibt y / x zurück, wenn gilt: x <= y
// Annahme: x != 0 und y != 0, also weder x noch y dürfen 0 sein.
public static int fromIntsToInt(int x, int y) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- [ ] `return x / y > 0 ? x / y : y / x;`
- [ ] `return y < x ? x / y : y / x ;`
- [x] `return Math.max(x/y, y/x);`
- [x] `return (x > y ? x : y) / (x > y ? y : x);`
- [ ] `return x > y ? x / y : x / -y;`

Aufgabe 3.4.

5 / 5 Punkte

```java
// Gibt folgenden Wert in Abhängigkeit von Parametern a und b zurück:
//   a      b      Rückgabe
//   false  false  0
//   true   false  1
//   false  true   2
//   true   true   3
public static int fromBoolsToInt(boolean a, boolean b) {
    // TODO: Welche return-Anweisung kann hier stehen?
}
```

- [ ] `return (a ? 3 : 1) - (b ? 0 : 1);`
- [x] `return (a ? 1 : 0) + (b ? 2 : 0);`
- [ ] `return (a ? 2 : 0) ^ (b ? 1 : 0);`
- [ ] `return (a ? 2 : 0) | (b ? 1 : 0);`
- [x] `return (a ? 3 : 2) & (b ? 3 : 1);`

4. Auswahlaufgaben zu Programmverzweigungen

0 / 20 Punkte

In den Methoden sind die Buchstaben A, B, C bis höchstens F 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.

0 / 5 Punkte

```java
// liefert die beiden Zeichen in alphabetischer Reihenfolge als String verkettet
// Wenn a == b enthält der String das Zeichen nur einmal.
// Beispiele:
// Wenn a = 'l' und b = 'y' wird "ly" geliefert.
// Wenn a = 'y' und b = 'x' wird "xy" geliefert.
// Wenn a = 'w' und b = 'w' wird "w" geliefert.
// Annahme: a und b sind Kleinbuchstaben.
public static String asString(char a, char b) {
    String result = A;
    if (B) {
        if (C) {
            result = D;
        } else {
            result = E;
        }
    }
    return result;
}
```

A:

- (●) `""`
- ( ) `b + result`
- ( ) `result + b`
- ( ) `result + a`
- ( ) `"" + a`
- ( ) `"" + b`
- ( ) `a`
- ( ) `b`

B:

- ( ) `a == b`
- ( ) `a > b`
- (●) `a != b`

C:

- ( ) `a == b`
- (●) `a > b`
- ( ) `a != b`

D:

- ( ) `""`
- ( ) `b + result`
- (●) `result + b`
- ( ) `result + a`
- ( ) `"" + a`
- ( ) `"" + b`
- ( ) `a`
- ( ) `b`

E:

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

Aufgabe 4.2.

0 / 5 Punkte

```
// liefert für n > 0 einen String mit der absteigenden Zahlenfolge n bis 1 durch
// also z.B "5_4_3_2_1" wenn n = 5.
// liefert sonst "1"
public static String sequence(int n) {
    if(A) {
        return B + "_" + C;
    }
    return D;
}
```

A:

- ○ `n != 1`
- ● `n >= 1`
- ○ `n == 1`
- ○ `n <= 1`
- ○ `n > 1`

B:

- ● `sequence(n)`
- ○ `1`
- ○ `n`
- ○ `"1"`
- ○ `sequence(n+1)`
- ○ `sequence(n-1)`

C:

- ○ `sequence(n)`
- ○ `1`
- ○ `n`
- ○ `"1"`
- ○ `sequence(n+1)`
- ● `sequence(n-1)`

D:

- ○ `sequence(n)`
- ○ `1`
- ○ `n`
- ○ `"1"`
- ○ `sequence(n+1)`
- ● `sequence(n-1)`

Aufgabe 4.3.

0 / 5 Punkte

```
// gibt an wie oft toFind als Element in a im Bereich vom Index from zum Index t
// Annahme: from und to sind gültige Indizes von a und from <= to
public static int count(final String[] a, final String toFind, final int from, f
    if(A) {
        if (B) {
            return C;
        } else {
            return D;
        }
    }
    return 0;
}
```

A:

- [x] `from <= to`
- [ ] `toFind.equals(a[from])`
- [ ] `from > to`
- [ ] `to > 0`
- [ ] `from == to`
- [ ] `from > 0`

B:

- [ ] `from <= to`
- [x] `toFind.equals(a[from])`
- [ ] `from > to`
- [ ] `to > 0`
- [x] `from == to`
- [ ] `from > 0`

C:

- [ ] `count(a, toFind, from + 1, to)`
- [x] `1`
- [ ] `1 + count(a, toFind, from + 1, to)`

D:

- [ ] `count(a, toFind, from + 1, to)`
- [ ] `1`
- [x] `1 + count(a, toFind, from + 1, to)`

Aufgabe 4.4.

0 / 5 Punkte

```
// liefert 1 wenn n größer als 0 ist, 0 wenn n gleich 0 ist und -1 sonst
// (entspricht der Vorzeichenfunktion)
public static int sign(int n) {
    switch (A) {
        case B:
            return C;
    }
    return -1;
}
```

A:

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

B:

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

C:

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

5. Auswahlaufgaben zu Schleifen und Rekursion

15 / 15 Punkte

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

Aufgabe 5.1.

5 / 5 Punkte

```
public static void forEachLoop(int n){
    int[] a = {n*87, n+34, n/2, n%3, 4*n+1, 2*n%2};
    int c = 0;
    for(int s : a){
        if(s > n) {
            c++;
        }
    }
    System.out.println(c);
}
```

Welche Zahl wird bei einem Aufruf von `forEachLoop(8)` ausgegeben?

- ○ 0
- ○ 1
- ○ 2
- ● 3
- ○ 4

Aufgabe 5.2.

5 / 5 Punkte

```
public static void recursion(int x){
    if(x > 0){
        System.out.println(x%3);
        recursion(x/3);
    }
}
```

Wie viele Zeilen werden bei einem Aufruf `recursion(600)` ausgegeben?

- ○ 0
- ○ 3
- ● 6
- ○ 200
- ○ 300

Aufgabe 5.3.

5 / 5 Punkte

```
public static int whileLoop(int e){
    int res = 0;
    int v = 0;
    while(v <= e){
        res += e+v++;
    }
    return res;
}
```

Welche Zahl wird bei einem Aufruf von `whileLoop(5)` retourniert?

- ○ 15
- ○ 25
- ○ 35
- ● 45
- ○ 55

6. Multiple-Choice-Aufgabe zu Ausdrücken, Schleifen und Rekursion

15 / 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 / 15 Punkte

Welche Methoden liefern als Ergebnis für `a` >= 0 und `b` > 0 denselben Wert wie der Ausdruck `a/b`?

- [x]
```
public static int div(int a, int b) {
    return a >= b ? 1 + div(a-b, b) : 0;
}
```

- [x]
```
public static int div(int a, int b) {
    int i = 1;
    for (; i*b <= a; i++);
    return i-1;
}
```

- [ ]
```
public static int div(int a, int b) {
    return (a*=1)/b;
}
```

- [ ]
```
public static int div(int a, int b) {
    int i = 1;
    do {
        i++;
    } while(i*b <= a);
    return i-1;
}
```

- [x]
```
public static int div(int a, int b) {
    int i = 1;
    while(i*b <= a) {
        i++;
    }
    return --i;
}
```