

Musterlösung

1. Kreuzen Sie zu jeder der folgenden Zeichenfolgen die richtigen Antworten an. Im Falle der syntaktischen Korrektheit gemäß der Java-Syntax, erläutern Sie die Bedeutung in Kurzform durch Vervollständigung der angegebenen Satzanfänge.

- (a) $8 /= x$ ist syntaktisch richtig
 falsch

Wenn x den Wert 4 hat,

- (b) $x /= 8$ ist syntaktisch richtig
 falsch

Wenn x den Wert 4 hat, *ist der Wert dieses Ausdrucks und der Wert von x gleich 0.*

- (c) $4 - -- 2$ ist syntaktisch richtig
 falsch

Wenn x den Wert 4 hat,

- (d) $4 - -- x$ ist syntaktisch richtig
 falsch

Wenn x den Wert 5 hat, *ist der Wert dieses Ausdrucks 0 und $x = 4$.*

- (e) $80 \% 200$ ist syntaktisch richtig
 falsch

Der Wert von $80 \% 200$ beträgt 160
 80
 2
 120

- (f) 037 ist syntaktisch richtig
 falsch

Der Wert von 037 ist gleich 31 (*sic!*)

(g) `if (x == 0)`
 `System.out.println ("x ist Null");`
 `else`
 `System.out.print ("x ist");`
 `System.out.println (x);`

ist syntaktisch richtig
 falsch

Falls x den Wert 0 hat, lautet die Ausgabe: x ist Null (dritte Ausgabezeile wird ausgeführt.)

Falls x den Wert 4 hat, lautet die Ausgabe: x ist 4.

(h) `if (x == 0);`
 `System.out.println ("x ist Null");`

ist syntaktisch richtig
 falsch

Falls x den Wert 0 hat, lautet die Ausgabe: x ist Null.

Falls x den Wert 4 hat, lautet die Ausgabe: x ist Null (wg. ; hinter Test).

(i) `float minwert = x <= y ? x : y;`

ist syntaktisch richtig
 falsch

Wenn x and y Variablen vom Typ `double` sind, dann erhält `minwert` das Minimum von x und y .

Wenn x and y Variablen vom Typ `int` sind, dann erhält `minwert` das Minimum von x und y als `float`-Wert.

(j) $t1$ und $t2$ seien primitive Datentypen.

- $t1$ ist kompatibel zu $t1$ richtig
 falsch
- Wenn $t1$ zu $t2$ kompatibel ist, dann ist auch $t2$ kompatibel zu $t1$ richtig
 falsch
- `byte` ist kompatibel zu $t1$ richtig
 falsch

(k) `double quotient;`
`int x = 6;`
`int y = 10;`
`quotient = x/y;`

ist syntaktisch richtig
 falsch

Nach Ausführung der angegebenen Sequenz hat `quotient` den Wert

0.0

0.6

```
(l) switch (c) {
    case '(':
    case '[':
    case '{':
        System.out.println ("Eine öffnende Klammer");
    case ')':
    case ']':
    case '}':
        System.out.println ("Eine schließende Klammer");
        break;
    default:
        System.out.println ("Etwas anderes");
}
```

ist syntaktisch

richtig

falsch

Wenn `c` den Wert `'[` hat, lautet die Ausgabe

Eine öffnende Klammer.

Eine schließende Klammer.

```
(m) int max (int a, int b)
    return a < b? b : a;
```

ist syntaktisch

richtig

falsch (*die Klammern um Methodenrumpf fehlen*)

Wenn `a` und `b` die Werte 3 und 4 haben hat der Aufruf `max (a,b)` den Wert

```
(n) double max (double a, double b) {
    if (a < b) return b;
    else return a;
}
```

ist syntaktisch

richtig

falsch

darf in derselben Klasse mit der Methode unter Punkt m (mit Klammern um Methodenrumpf) stehen ja, *Überladen ist möglich*

nein,

```
(o) static final double PI=3.1415926535897932;
```

ist syntaktisch

richtig

falsch

Eine anschließende Anweisung

```
PI=3.14;
```

ist

erlaubt

nicht erlaubt