ulm university universität ulm

Einführung in die Informatik

Institut für Eingebettete Systeme/Echtzeitsysteme | Wintersemester 2020/21
Leo Auterhoff, Julian Bestler, Valentina Richthammer, Michael Glaß

Übungsblatt 6: Methoden

Abgabetermin: 20.12.2020, 23:59 Uhr

Geben Sie **Programmieraufgaben als Java Code (`*.java` Dateien)** ab. Alle anderen Aufgaben, die Text oder Grafiken erfordern, geben Sie **als PDF Dateien** ab. PDFs können Sie beispielsweise mit dem kostenlosen Programm *LibreOffice* erstellen. Alternativ können Sie etwas mehr Zeit investieren und LaTeX lernen, was Sie im späteren Studium immer wieder brauchen werden.

Wenn Sie **mehrere Dateien** abgeben wollen, dann fassen Sie diese zu **einem ZIP File** zusammen.

Präsenzaufgaben werden direkt im Tutorium bearbeitet, werden nicht bepunktet und müssen nicht abgegeben werden.

Präsenzaufgabe

Welche Ausgaben produziert dieser Code und warum?

```java
public class Code {

    static int a = 5;
    static int b = 10;
    static boolean c = true;

    public static void main(String[] args) {
        methode1(b, a);
        methode2(a, c);
    }

    static void methode1(int a, int b) {
        boolean c = false;
        b = b * a;
        System.out.println(a + " " + b + " " + c);
    }

    static void methode2(int b, boolean d) {
        b = a - b;
        System.out.println(a + " " + b + " " + c + " " + d);
    }
}
```

Aufgabe 1: Variablen Sichtbarkeit (6)

Gegeben sei folgendes Java-Programm.

```java
public class Sichtbar {

    static int x = 0;
    static int y = 0;
    static boolean z = false;

    public static void main(String... args) {
        // Zeitpunkt 1
        methode1(y, x);
        System.out.println("x: " + x);
        System.out.println("y: " + y);
        System.out.println("z: " + z);
        // Zeitpunkt 2
        methode2(x, y);
        System.out.println("x: " + x);
        System.out.println("y: " + y);
        System.out.println("z: " + z);
        // Zeitpunkt 3
        methode3(x, y, z);
        System.out.println("x: " + x);
        System.out.println("y: " + y);
        System.out.println("z: " + z);
    }

    public static void methode1(int x, int y) {
        x = y + 2;
        y = x + 3;
    }

    public static void methode2(int a, int y) {
        a = y + 2;
        x = a + 3;
    }

    public static void methode3(int a, int b, boolean c) {
        a = y;
        b = a + 5;
        x = a + b;
        y = x * 5;
        if (y < x)
            c = true;
        else
            z = true;
    }
}
```

Welche Ausgaben erzeugt das Programm und warum?

Aufgabe 2: Funktionsüberladung (2+2+2)

a) **Erweitern Sie folgendes Programm** so, dass die Aufrufe in der main-Methode korrekt funktionieren und die angegebenen erwarteten Ausgaben erzeugen. Verändern Sie dabei nicht die bereits vorgegebenen Methoden. **Erklären Sie**, wie viele Funktionen notwendig sind und warum. Stellen Sie außerdem sicher, dass bei jedem konkatenierten Wort, welches *Katze* enthält, automatisch die Methode cat() aufgerufen wird. Sie können hierzu die String-Methode *contains*[1] nutzen.

```
1  public class Konkatenieren {
2
3      static String str1 = "  _._     _,-'\"\"\"`-._      \n (,-.`._,'(       |\\`-/|  \n";
4      static String str2 = "     `-.-' \\ )-`( , o o)  \n Katze      `-     \\`_`\"'-   ";
5
6      private static void cat() {
7          System.out.println(str1 + str2);
8      }
9
10     public static void main(String[] args) {
11
12         char a = 'K';
13         char b = 'H';
14
15         String sa = "atze";
16         String sb = "und";
17
18         String e1 = concat(a, sa);
19         String e2 = concat(b, sb);
20         String e3 = concat(sa, sb);
21         String e4 = concat(concat(a, sa), " " + sb + " " + concat(b, sb));
22
23         System.out.println(e1); // Ausgabe: "Katze"
24         System.out.println(e2); // Ausgabe: "Hund"
25         System.out.println(e3); // Ausgabe: "atzeund"
26         System.out.println(e4); // Ausgabe: "Katze und Hund"
27     }
28 }
```

b) Durch welche Anpassungen kann eine Funktion in Java überladen werden? Anders gesagt: was wird verwendet, um bei einer überladenen Funktion festzustellen, welche Funktion tatsächlich ausgeführt werden soll?

c) Können die folgenden beiden Methoden gleichzeitig in einer Java Klasse existieren? **Erklären** Sie, warum es funktioniert oder nicht funktioniert.

```
1  public class Overload {
2
3      public static boolean foo(String a) {
4          return false;
5      }
6
7      public static int foo(String b) {
8          return 69;
9      }
10     // ...
11 }
```

[1] https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#contains(java.lang.CharSequence)

Aufgabe 3: Call-by (3+3)

Betrachten Sie das folgende Programm. Überlegen Sie sich, wie sich das Programm bei den verschiedenen Parameterübergabemechanismen für den Aufruf der Methoden `m(String x, int y, boolean c)` und `n(int i)` verhält, **unabhängig** von dem standardmäßigen JAVA-Parameterübergabemechanismus.

Erklären Sie für **beide Parameterübergabemechanismen** (*call by reference* und *call by value*) welche Ausgabe das Programm erzeugt und wie diese zustande kommt!

Hinweis: Nehmen Sie für die **return**-Anweisung jeweils den gleichen Mechanismus an, wie für den Parameter-Übergabemechanismus, den Sie gerade betrachten.
Gehen Sie beim Referenz-Aufruf davon aus, dass beim Aufruf `n(m(a,b,c))` nicht eine Referenz auf die Methode `m(String x, int y, boolean c)` übergeben wird, sondern `m(a,b,c)` vor dem Aufruf von `n(int i)` ausgeführt wird.

```
1  public class Program {
2
3      static String a = "Blumen";
4      static int b = 20;
5      static boolean c = false;
6
7      static int m(String x, int y, boolean c) {
8          x = "Ich habe " + y + " " + x;
9          c = true;
10         y = b + 10;
11         return b;
12     }
13
14     static void n(int i) {
15         if (c)
16             i = i + i;
17         else
18             i = i - 2;
19     }
20
21     public static void main(String[] args) {
22         n(m(a, b, c));
23
24         System.out.println(a);
25         System.out.println(b);
26         System.out.println(c);
27     }
28 }
```