Objektorientierte Programmierung I
freiwillige Übungsaufgaben
24.05.2022

Legende:

🌶 = einfach
🌶🌶 = mittel
🌶🌶🌶 = schwer

🌶 Aufgabe 1

Welche Aussagen stimmen? Kreuze an.

Aussage	wahr	falsch
Beim Aufruf einer Methode werden die Rücksprungadresse, die Argumente und die lokalen Variablen auf dem Stack gespeichert.		
Das `this`-Keyword steht nur in statischen Methoden zur Verfügung.		
Statische Methoden können andere statische Methoden aufrufen.		
Eine statische Methode kann nur auf einem Objekt, also einer konkreten Klassen-Instanz aufgerufen werden.		
Rekursion ist ausschließlich bei statischen Methoden möglich.		
Eine endlose Rekursion tritt auf, wenn es keine Abbruchbedingung gibt.		

Objektorientierte Programmierung I
freiwillige Übungsaufgaben
24.05.2022

🌶 Aufgabe 2

Gegeben sei das folgende Programm:

```
import java.util.Random;

class Sum {

    public static void main(String[] args) {
        int n = 20;

        // 1. Ein Array aus n Zufallszahlen generieren
        Random random = new Random();
        int[] randomNumbers = new int[n];
        for (int i = 0; i < randomNumbers.length; ++i) {
            randomNumbers[i] = random.nextInt(101);
        }

        // 2. Die Summe aller Zahlen berechnen
        //    Es sollen dabei aber NUR die geraden Zahlen addiert werden!
        int sum = 0;
        for (int i = 0; i < randomNumbers.length; ++i) {
            if (randomNumbers[i] % 2 == 0) {
                sum += randomNumbers[i];
            }
        }

        // 3. Ausgabe
        System.out.println("Summe der geraden Zahlen im Array: " + sum);
    }

}
```

Das Programm besteht aus drei Teilen, die jeweils durch Kommentare eingeleitet werden. Das Programm soll mithilfe von statischen Methoden in Unterprogramme aufgeteilt und somit übersichtlicher gemacht werden.

Wird z. B. der erste Programmteil in eine statische Methode ausgelagert, könnte das Programm wie folgt aussehen:

```
import java.util.Random;

class Sum {

    public static int[] generateRandomNumbers(int count) {
        Random random = new Random();
        int[] array = new int[count];
        for (int i = 0; i < array.length; ++i) {
            array[i] = random.nextInt(101);
        }
        return array;
    }

    public static void main(String[] args) {
        int n = 20;

        int[] randomNumbers = generateRandomNumbers(n);

        // 2. Die Summe aller Zahlen berechnen
        //    Es sollen dabei aber NUR die geraden Zahlen addiert werden!
        int sum = 0;
        for (int i = 0; i < randomNumbers.length; ++i) {
            if (randomNumbers[i] % 2 == 0) {
                sum += randomNumbers[i];
            }
        }

        // 3. Ausgabe
        System.out.println("Summe der geraden Zahlen im Array: " + sum);
    }

}
```

Versuche zunächst, diesen Schritt komplett nachzuvollziehen, bevor du weiterliest!

Lagere danach den zweiten und dritten Schritt auch jeweils in statische Methoden aus. Überlege, welche Parameter und welchen Rückgabedatentyp die Methoden jeweils haben müssen. Versuche, aussagekräftige Namen für die Methoden zu finden.

Objektorientierte Programmierung I
freiwillige Übungsaufgaben
24.05.2022

Aufgabe 3

Gegeben sei das folgende Programm:

```
import java.util.Scanner;

class Contains {

    public static boolean contains(int[] array, int value) {
        // hier Code einfügen
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int[] numbers = { 10, 39, 11, 3, 4, 16 };

        System.out.print("Nach welcher Zahl suchst du? ");
        int input = scanner.nextInt();
        if (contains(numbers, input)) {
            System.out.println(input + " ist im Array vorhanden.");
        } else {
            System.out.println(input + " ist nicht im Array vorhanden.");
        }
        scanner.close();
    }

}
```

Die statische Methode „`contains`“ soll `true` zurückgeben, falls das übergebene Array den Wert `value` enthält, ansonsten soll `false` zurückgegeben werden. Füge an der markierten Stelle Code ein, sodass das Programm korrekt funktioniert.

Aufgabe 4

Schreibe eine statische Methode namens `areArraysEqual`, die als Parameter zwei `int`-Arrays entgegennimmt. Die Methode soll herausfinden, ob die beiden Arrays den gleichen Inhalt haben. Damit ist gemeint, dass dieselben Elemente in derselben Reihenfolge in beiden Arrays enthalten sind. Überlege, welchen Rückgabedatentyp die Methode haben muss.

Teste deine Methode mithilfe eines Hauptprogramms.

Objektorientierte Programmierung I
freiwillige Übungsaufgaben
24.05.2022

🌶 Aufgabe 5

Schreibe eine statische Methode namens „`containsAnyEvenNumber`“, die bei einem als Argument übergebenen `int`-Array prüft, ob sich darin **mindestens** eine gerade Zahl befindet. Überlege, wie die Parameterliste und der Rückgabetyp der Methode sein müssen.

🍌🍌 Aufgabe 6

Die folgende statische Methode soll eine Zufallszahl generieren:

```
public static int generateRandomNumber(Random random, int min, int max) {
    // hier Code einfügen
}
```

Die Funktion erwartet ein Objekt vom Typ `Random` sowie zwei `int`-Werte als Argumente. Die zurückgegebene Zufallszahl soll im Intervall $I = [min; max]$ liegen.

Hinweis für die nächste Aufgabe:

Mithilfe der Methode `Math.sqrt()` (von engl. „square root“ für „Quadratwurzel“) kannst du die Wurzel aus einer Zahl berechnen. Es ist dafür kein zusätzlicher `import` nötig.

🍌🍌 Aufgabe 7

Wir wollen ein Programm schreiben, mit dessen Hilfe quadratische Gleichungen der Form

$$0 = ax^2 + bx + c$$

gelöst werden können. Dazu nutzen wir die sog. „Mitternachtsformel“:

$$x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$$

Dabei steht D für die sog. „Diskriminante“ und ist definiert als $D = b^2 - 4ac$.

Schreibe eine Klasse „`Equation`“ anhand des folgenden UML-Klassendiagramms:

Equation
a: double b: double c: double
<<create>> Equation(a: double, b: double, c: double) double calculateDiscriminant() int getNumberOfSolutions() double getFirstSolution() double getSecondSolution()

Die Methode `calculateDiscriminant()` soll die Diskriminante D berechnen und zurückgeben.
Die Methode `getNumberOfSolutions()` soll die Anzahl der Lösungen der Gleichung zurückgeben.

> Es gilt:
> Es gibt **keine** Lösung, wenn gilt: $D < 0$
> Es gibt **genau eine** Lösung, wenn gilt: $D = 0$
> Es gibt **zwei** Lösungen, wenn gilt: $D > 0$

Mithilfe der Methoden `getFirstSolution()` und `getSecondSolution()` soll man dann die Lösungen der Gleichungen erhalten können. Wenn der entsprechende Aufruf ungültig ist, also wenn z. B. `getSecondSolution()` aufgerufen wird, obwohl die Gleichung weniger als zwei Lösungen hat, soll die Zahl 0 zurückgegeben werden.

Teste deine Klasse mithilfe des folgenden Hauptprogramms:

```
import java.util.Scanner;

class Main {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Rechner fuer quadratische Gleichungen");
        System.out.println("(0 = ax^2 + bx + c)\n");
        System.out.println("Bitte die Koeffizienten eingeben:");
        System.out.print("a = ");
        double a = scanner.nextDouble();
        System.out.print("b = ");
        double b = scanner.nextDouble();
        System.out.print("c = ");
        double c = scanner.nextDouble();

        Equation equation = new Equation(a, b, c);
        switch (equation.getNumberOfSolutions()) {
            case 0:
                System.out.println("Die Gleichung hat keine Loesung.");
                break;
            case 1:
                System.out.println("x = " + equation.getFirstSolution());
                break;
            case 2:
                System.out.println("x1 = " + equation.getFirstSolution()
                    + ", x2 = " + equation.getSecondSolution()
                );
                break;
        }

        scanner.close();
    }

}
```

Ein Durchlauf des Programms könnte zum Beispiel wie folgt aussehen:

```
Rechner fuer quadratische Gleichungen
(0 = ax^2 + bx + c)

Bitte die Koeffizienten eingeben:
a = 3
b = -6
c = -45
x1 = 10.0, x2 = 2.0
```

🌶🌶🌶 Aufgabe 7

Schreibe eine **rekursive**, statische Methode, die zwei Werte vom Typ `int` addiert. Du darfst dabei jedoch lediglich Werte um 1 erhöhen oder vermindern. Du darfst davon ausgehen, dass die beiden gegebenen Werte nicht negativ sind.

Zusatz: Überlege, wie du die Methode anpassen könntest, um auch negative Zahlen zu ermöglichen.

Hinweis für die nächste Aufgabe:

Die Klasse `String` verfügt über eine Methode `substring()`, die einen neuen `String` aus einem Teil des Originalstrings erzeugt. Beispiel:

```
String text = "Hallo";
System.out.println(text.substring(1, 4)); // Ausgabe: all
```

Informiere dich über die Parameter, die diese Methode entgegennimmt, in der Java-Dokumentation unter https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/lang/String.html#substring(int,int).

🌶🌶🌶 Aufgabe 8

Schreibe eine **rekursive**, statische Methode, die überprüft, ob es sich bei einem Wort um ein Palindrom handelt (siehe Übungsblatt 6).

🌶🌶🌶 Aufgabe 9

Schreibe eine **rekursive**, statische Methode, die das n-te Glied der Fibonacci-Folge ausgibt (siehe Übungsblatt 3).