Präsenzaufgaben 12 06.01.2022

Die Lösung der Aufgaben wird am Ende der Übung von Ihnen vorgestellt.

Pair und Tuple

Schreiben Sie zwei generische Klassen `Pair<T>` und `Tuple<T,U>`. Beide Klassen sollen zwei Werte aufnehmen können. In der Klasse `Pair` müssen beiden Werte den gleichen Typ haben. In der Klasse `Tuple` können beide Werte unterschiedliche Typen haben.

Beide Klassen sollen einen Konstruktor haben, der die beiden Werte als Übergabeparameter enthält. Außerdem sollen beide Klassen jeweils die getter-Methoden `get1` und `get2`, die setter-Methoden `set1` und `set2` und eine `toString`-Methode haben.

equals

Fügen Sie beiden Klassen eine `equals`-Methode hinzu. Überschreiben Sie dabei die `equals`-Methode aus `Object`:

```
public boolean equals(Object z)
```

Falls das übergebene Objekt nicht ein Pair bzw. Tuple ist, geben Sie false zurück. Es ist nicht möglich, aber auch nicht nötig, zu überprüfen, ob die generischen Parameter übereinstimmen.

Größte Zahl

Schreiben Sie eine Klasse `Calc` mit einer Funktion

```
public static Tuple<Double, Integer> getMax(double[] arr)
```

die die größte Zahl und den Index der größten Zahl aus dem Feld `arr` in einem Tupel zurückgibt.

Fibonacci-Zahlen

Fügen Sie der Klasse `Calc` eine Funktion zur Berechnung der Fibonacci-Zahlen hinzu. Nutzen Sie dabei die Klasse `Pair`. Schreiben Sie einen iterativen Algorithmus. Gehen Sie nach folgender Gleichung vor:

$(f_n , f_{n-1}) = (f_{n-1}+f_{n-2} , f_{n-1})$.

JUnit-Testklassen

Schreiben Sie eine JUnit-Testklasse `CalcTest` für Ihre Fibonacci-Funktion. Testen Sie mindestens

```
fibonacci(1)  --> 1
fibonacci(2)  --> 1
fibonacci(3)  --> 2
fibonacci(10) --> 55
fibonacci(-1) --> IllegalArgumentException
```

Zusatzaufgabe

Da (zumindest in Eclipse und IntelliJ) die Laufzeiten für die einzelnen Tests angezeigt werden, eignen sich JUnit-Tests auch als Performance-Tests. Schreiben Sie einen Testfall für `fibonacci(1000000000)`. Das Ergebnis stimmt zwar nicht, weil es einen Bereichsüberlauf gibt, aber die Laufzeit ist trotzdem aussagekräftig. Sie sollte je nach Computer bei etwa 10 Sekunden liegen. Probieren Sie, eine schnellere Fibonacci-Routine zu schreiben. Was bremst die Fibonacci-Funktion aus?