Aufgabenbeschreibung Live-Aufgabe

Objekttechnologien - Sommersemeter 20118

1 Grundlegendes

Im Rahmen dieser Live-Aufgabe soll mit Hilfe des Decorator-Patterns eine Variante des Spiels „FizzBuzz“ nachprogrammiert werden.

2 Erstellen Sie folgende Klasse

2.1 DefaultCounter

Diese Klasse besitzt eine Methode:

2.1.1 *int nextValue()*

Diese Methode liefert bei jedem Aufruf einen um 1 erhöhten Wert zurück.

3 Decorator-Pattern

3.1 Counter

Erstellen Sie für die erzeugte Klasse und deren Methode ein Interface mit dem Namen *Counter*

3.2 Mod3Decorator

Erstellen Sie eine Klasse *Mod3Decorator*, die als Decorator für *Counter* dient.
Dieser Decorator zeigt folgendes Verhalten:

- Wenn der vom dekorierten Objekt zurück gelieferte Wert durch *3* teilbar ist, wird ein Text auf der Konsole ausgegeben.
- Andernfalls erfolgt keine Ausgabe.

Der Wert wird immer unverändert zurück gegeben.

3.3 Mod4Decorator

Erstellen Sie eine Klasse *Mod4Decorator*, die als Decorator für *Counter* dient.
Dieser Decorator zeigt folgendes Verhalten:

- Wenn der vom dekorierten Objekt zurück gelieferte Wert durch *4* teilbar ist, wird ein Text auf der Konsole ausgegeben.

- Andernfalls erfolgt keine Ausgabe.

Der Wert wird immer unverändert zurück gegeben.

3.4 Mod3And4Decorator

Erstellen Sie eine Klasse *Mod3And4Decorator*, die als Decorator für *Counter* dient.

Dieser Decorator ändert die von *int nextValue()* zurück gelieferten Werte nach folgendem Schema ab:

- Wenn der vom dekorierten Objekt zurück gelieferte Wert durch *3* und *4* teilbar ist, wird *-1* zurück gegeben.
- Andernfalls wird der Wert unverändert zurück gegeben.

3.5 Ausgabe

Fügen Sie die einzelnen Decorator in sinnvoller Weise so zusammen, rufen Sie die Methode *nextValue()* 24 mal auf und geben Sie den zurück gelieferten Wert auf der Konsole aus.

Dabei soll folgende Ausgabe entstehen:

```
1
2
3 ist durch 3 teilbar
3
4 ist durch 4 teilbar
4
5
6 ist durch 3 teilbar
6
7
8 ist durch 4 teilbar
8
9 ist durch 3 teilbar
9
10
11
12 ist durch 3 teilbar
12 ist durch 4 teilbar
-1
13
14
15 ist durch 3 teilbar
15
16 ist durch 4 teilbar
```

```
16
17
18 ist durch 3 teilbar
18
19
20 ist durch 4 teilbar
20
21 ist durch 3 teilbar
21
22
23
24 ist durch 3 teilbar
24 ist durch 4 teilbar
-1
```

4 Zur Klarstellung

4.1 Modulo

Um festzustellen, ob eine Zahl durch eine bestimmte Zahl teilbar ist, bietet sich der Modulo-Operator (%) an.
Beispiel:

```
if (value % 17 ==0 ){
  //Value ist durch 17 teilbar
}
```

4.2 Test / Main-Klasse

Erzeugen Sie ein Demo-Programm oder einen JUnit-Test, welcher die Funktionsfähigkeit und Korrektheit Ihrer Lösung demonstriert.