Stack

Dr. Matthias Hölzl und Allaithy Raed

14. Oktober 2019

Kata: Stack-Datenstruktur

- Erstellen Sie eine einfache `Stack`-Datenstruktur mittels TDD.
- Inkrementelles Vorgehen; kleine Schritte!
- **Bitte blättern Sie nach jedem Schritt erst weiter, wenn Sie das jeweilige Feature implementiert haben.**

Kata: Stack-Datenstruktur

- Entpacken Sie das Archiv mit der `StackKata` Template (`StarterKit.zip`) und öffnen Sie das Projekt.

- Solve simply!
- Testen Sie Funktionalität, nicht Implementierung!

Stack

- Implementieren Sie einen `Stack`-Datentyp mit folgender Signatur:
- `void push(int element)`
- `int pop()`
- `boolean isEmpty()`
- Wenn der Stack leer ist, soll `pop()` eine geeignete Exception werfen.

Erweiterung (1)

- Erweitern Sie den `Stack`-Datentyp um die Methode

  `int size()`

  die die Anzahl der auf dem Stack liegenden Elemente zurückgibt.

Erweiterung (2)

- Erweitern Sie den `Stack`-Datentyp um die Methode
  `int count(int element)`
  die zurückgibt, wie oft `element` auf dem Stack liegt.

Erweiterung (3)

- Erweitern Sie den `Stack`-Datentyp um die Methode

  `int popDefault(int default)`

  die sich wie `pop()` verhält, wenn der Stack nicht leer ist und `default` zurückgibt, wenn der Stack leer ist.

Erweiterung (4)

- Erweitern Sie den `Stack`-Datentyp um Methoden

  ```
  void setDefault(int default)
  void clearDefault()
  ```

- Nachdem `setDefault(default)` aufgerufen wurde, soll sich `pop()` wie `popDefault(default)` verhalten.
- Wenn `clearDefault()` aufgerufen wird, soll sich der Stack wieder wie ursprünglich verhalten.

Erweiterung (5)

- Ihr `Stack`-Datentyp soll jetzt auf einem Microcontroller verwendet werden, für den keine Collection-Klassen zur Verfügung stehen.
- Modifizieren Sie die Implementierung so, dass ein `int`-Array zum Speichern der auf dem Stack liegenden Elemente verwendet wird.
- Der Stack soll aber dennoch beliebig viele Elemente speichern können (nur begrenzt durch die Beschränkungen der JVM).
- Um keinen Speicher zu verschwenden soll ein Stack initial nur ein Array der Länge 1 verwenden und seine Daten in ein größeres Array kopieren, wenn das aktuell verwendete Array nicht mehr ausreicht.

Erweiterung (6)

- Erweitern Sie den `Stack`-Datentyp um eine Methode

  `void setCapacity(int n)`

  die die Größe des verwendeten Arrays folgendermaßen anpasst:
  - Wenn maximal `n` Elemente auf dem Stack liegen, so wird der Speicher durch ein Array der Länge `n` ersetzt
  - Wenn mehr als `n` Elemente auf dem Stack liegen, so wird der Speicher durch ein Array der Länge `size()` ersetzt.
- Was passiert bei Ihrer Implementierung, wenn `setCapacity(0)` aufgerufen wird?