Staatsexamen 46115 / 2021 / Frühjahr / Thema Nr. 2 / Teilaufgabe Nr. 2 / Aufgabe Nr. 2

Aufgabe 2 [Java-Klasse Stack]

Gegeben sei die folgende Java-Implementierung eines Stacks.

```java
class Stack {
  private Item head;

  public Stack() {
    head = null;
  }

  public void push(int val) {
    if (head == null) {
      head = new Item(val, null);
    } else {
      head = new Item(val, head);
    }
  }

  public int pop() {
    // ...
  }

  public int size() {
    // ...
  }

  public int min() {
    // ...
  }

  class Item {
    private int val;
    private Item next;

    public Item(int val, Item next) {
      this.val = val;
      this.next = next;
    }
  }
}
```

(a) Implementieren Sie die Methode pop in einer objektorientierten Programmiersprache Ihrer Wahl, die das erste Item des Stacks entfernt und seinen Wert zurückgibt. Ist kein Wert im Stack enthalten, so soll dies mit einer IndexOutOfBoundsException oder Ähnlichem gemeldet werden.

Beschreiben Sie nun jeweils die notwendigen Änderungen an den bisherigen Implementierungen, die für die Realisierung der folgenden Methoden notwendig sind.

```java
public int pop() {
  if (head != null) {
    int val = head.val;
    size--;
    head = head.next;
    return val;
  } else {
    throw new IndexOutOfBoundsException("The stack is empty");
  }
}
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/Stack.java`

(b) size gibt in Laufzeit O(1) die Anzahl der enthaltenen Items zurück.

```java
13    public void push(int val) {
14      if (head == null) {
15        head = new Item(val, null);
16      } else {
17        head = new Item(val, head);
18      }
19      if (min > val) {
20        min = val;
21      }
22      size++;
23    }
24
25    public int pop() {
26      if (head != null) {
27        int val = head.val;
28        size--;
29        head = head.next;
30        return val;
31      } else {
32        throw new IndexOutOfBoundsException("The stack is empty");
33      }
34    }
35
36    public int size() {
37      return size;
38    }
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/Stack.java`

(c) min gibt (zu jedem Zeitpunkt) in Laufzeit O(1) den Wert des kleinsten Elements im Stack zurück.

```java
13    public void push(int val) {
14      if (head == null) {
15        head = new Item(val, null);
16      } else {
17        head = new Item(val, head);
18      }
19      if (min > val) {
20        min = val;
21      }
22      size++;
23    }
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/Stack.java`

```java
40    public int min() {
41      return min;
42    }
43
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/Stack.java`

Sie dürfen jeweils alle anderen angegebenen Methoden der Klasse verwenden, auch wenn Sie diese nicht implementiert haben. Sie können anstelle von objektorientiertem Quellcode auch eine informelle Beschreibung Ihrer Änderungen angeben.

Additum: Kompletter Java-Code

```java
3  class Stack {
4    private Item head;
5
6    private int size;
7    private int min = Integer.MAX_VALUE;
8
9    public Stack() {
```

```
10        head = null;
11      }
12
13      public void push(int val) {
14        if (head == null) {
15          head = new Item(val, null);
16        } else {
17          head = new Item(val, head);
18        }
19        if (min > val) {
20          min = val;
21        }
22        size++;
23      }
24
25      public int pop() {
26        if (head != null) {
27          int val = head.val;
28          size--;
29          head = head.next;
30          return val;
31        } else {
32          throw new IndexOutOfBoundsException("The stack is empty");
33        }
34      }
35
36      public int size() {
37        return size;
38      }
39
40      public int min() {
41        return min;
42      }
43
44      class Item {
45        private int val;
46        private Item next;
47
48        public Item(int val, Item next) {
49          this.val = val;
50          this.next = next;
51        }
52      }
53    }
```

Code-Beispiel auf Github ansehen: src/main/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/Stack.java

```
3   import static org.junit.Assert.assertEquals;
4   import static org.junit.Assert.assertThrows;
5
6   import org.junit.Test;
7
8   public class StackTest {
9
10    private Stack makeStack() {
11      Stack stack = new Stack();
12      stack.push(1);
13      stack.push(2);
14      stack.push(3);
15      return stack;
16    }
17
18    @Test
19    public void methodPop() {
20      Stack stack = makeStack();
21      assertEquals(3, stack.pop());
22      assertEquals(stack.size(), 2);
23      assertEquals(2, stack.pop());
24      assertEquals(stack.size(), 1);
```

```
25        assertEquals(1, stack.pop());
26        assertEquals(stack.size(), 0);
27        assertThrows(IndexOutOfBoundsException.class, () -> {
28          stack.pop();
29        });
30      }
31
32      @Test
33      public void methodSize() {
34        Stack stack = makeStack();
35        assertEquals(stack.size(), 3);
36      }
37
38      @Test
39      public void methodMin() {
40        Stack stack = makeStack();
41        assertEquals(stack.min(), 1);
42      }
43    }
```

Code-Beispiel auf Github ansehen: src/test/java/org/bschlangaul/examen/examen_46115/jahr_2021/fruehjahr/StackTest.java

Github: Staatsexamen/46115/2021/03/Thema-2/Teilaufgabe-2/Aufgabe-2.tex