

## Übung 2: Stack, Queue

Abgabetermin: 09.04.2013

Name: \_\_\_\_\_

Matrikelnummer: \_\_\_\_\_

Gruppe:  G1 Di 10:15  G2 Di 11:00  G3 Di 12:45

Aufgabe	Punkte	gelöst	abzugeben schriftlich	abzugeben elektronisch	Korr.	Punkte
Aufgabe 1	12	<input type="checkbox"/>	Java-Programm Testfälle und Ergebnisse	Java-Programm	<input type="checkbox"/>	
Aufgabe 2	12	<input type="checkbox"/>			<input type="checkbox"/>	

### Aufgabe 1: Stack für Objekte (12 Punkte)

Implementieren Sie einen Kellerspeicher für Objekte, einmal mit einem Array in der Klasse *ArrayStack* und einmal als verkettete Liste in der Klasse *LinkedListStack*. Die Schnittstelle beider Klassen ist durch die abstrakte Klasse *Stack* gegeben (für Methodenbeschreibungen siehe Java-Dokumentation in der Vorgabedatei).

```
package at.jku.ssw;

public abstract class Stack {
    public abstract void push(Object value);
    public abstract Object pop();
    public abstract int size();
    public abstract Iterator iterator();
}

public abstract class Iterator {
    public abstract boolean hasNext();
    public abstract char next();
}
```

Implementieren Sie die Klassen *ArrayStack* und *LinkedListStack* im Paket *at.jku.students*. Verwenden Sie dazu die Klassen *Stack*, *Iterator* und *LinkedList* aus der Vorgabedatei.

```
package at.jku.students;

public class ArrayStack extends Stack {
    Object[] stack = new Object[1]; int count = 0;
    ...
}

public class LinkedListStack extends Stack {
    // aus Vorgabe
    List list = at.jku.ssw.LinkedList();
    ...
}

Stack s = new ArrayStack();
s.push('2'); s.push('K');
s.pop();
s.push('I'); s.push('P');
Iterator it = s.iterator();
while (it.hasNext()) {
    Out.print(" " + it.next());
} // Ausgabe: P I 2
```

### Implementierungshinweise:

- Definieren Sie für alle Klassen, Methoden und Felder die geeignete Sichtbarkeit (*private*, *protected*, *package*, *public*).
- In der Klasse *ArrayStack* initialisieren Sie das Array mit Länge 1 und verdoppeln Sie die Länge, wenn das Array voll ist.
- In der Klasse *LinkedListStack* verwenden Sie die Klasse *at.jku.ssw.LinkedList* mit der Schnittstelle *at.jku.ssw.List* aus der Vorgabedatei. Implementieren Sie den Kellerspeicher, indem Sie die Stack-Operationen auf die passende Listen-Operationen abbilden.

```
package at.jku.ssw;

public abstract class List {
    public abstract void add(Object value);
    public abstract void add(
        int index, char value);
    public abstract Object get(int index);
    public abstract Object remove(int index);
    public abstract int indexOf(Object value);
    public abstract int size();
    public abstract Iterator iterator();
}
```

Zeigen Sie die Funktion der beiden Klassen mit folgender Zeichenfolge:

N S H \* A \* Q D R \* \* \* E U \* \* \* O T \* I \* \* \*

Ein Buchstabe bedeutet *push* und ein Sternchen *pop*. Geben Sie die Sequenz der Zeichen an, die die Pop-Operation zurückgibt.

Abzugeben ist: Java-Programm, Testfälle, gesuchte Zeichensequenz

### Aufgabe 2: Queue für Objekte (12 Punkte)

Implementieren Sie eine FIFO-Warteschlange (First-in-first-out) für Objekte, einmal mit einem Array in der Klasse *ArrayQueue* (zyklischer Puffer, Array mit fixer Größe) und einmal als verkettete Liste in der Klasse *LinkedListQueue*. Die Schnittstelle beider Klassen ist durch die abstrakte Klasse *Queue* gegeben (für Methodenbeschreibungen siehe Java-Dokumentation in der Vorgabedatei).

```
package at.jku.ssw;                                     // Iterator & List
public abstract class Queue {                           // siehe Aufgabe 1
    public abstract void put(Object value);
    public abstract Object get();
    public abstract int size();
    public abstract Iterator iterator();
}
```

Implementieren Sie die Klassen *ArrayQueue* und *LinkedListQueue* im Paket *at.jku.students*. Für die *LinkedListQueue* verwenden Sie wie in Aufgabe 1 die Klasse *LinkedList*.

```
package at.jku.students;                               Queue q = new ArrayQueue();
public class ArrayQueue extends Queue {                q.put('P'); q.put('I'); q.put('2');
    static final int MAX_SIZE = 100;                  Out.print(l.size() + ": ");
    Object[] queue = new Object[MAX_SIZE];            Iterator it = l.iterator();
    int count = 0;                                    while (it.hasNext()) {
    ...                                                Out.print(" " + it.next());
}                                                        }
public class LinkedListQueue extends Queue {           // Ausgabe 3: P I 2
    // aus Vorgabe
    List list = at.jku.ssw.LinkedList();
    ...
}
```

Zeigen Sie die Funktion der beiden Klassen mit folgender Zeichenfolge:

E A S \* Y \* Q U E \* \* \* S T \* \* \* I O \* N \* \* \*

Ein Buchstabe bedeutet *put* und ein Sternchen *get*. Geben Sie die Sequenz der Zeichen an, die die Get-Operation zurückgibt.

Abzugeben ist: Java-Programm, Testfälle, gesuchte Zeichensequenz