Übung 2: Stack, Queue

Abgabetermin: 18.03.2014

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	☐	Java-Programm Testfälle und Ergebnisse	Java-Programm	☐	
Aufgabe 2	12	☐	Java-Programm Testfälle und Ergebnisse	Java-Programm	☐	

Hinweis zur Abgabe: **Bitte das package at.jku.students verwenden und als elektronische Abgabe eine ZIP-Datei vom *src*-Verzeichnis erzeugen.**

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 Object 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 = new at.jku.ssw.LinkedList();
  ...
}
```

```
Stack s = new ArrayStack();
s.push("2");
s.push("Kurs");
s.pop();
s.push("Informatik");
s.push("Praktische");
Iterator it = s.iterator();
while (it.hasNext()) {
  Out.print(" " + it.next());
}  // Ausgabe: Praktische Informatik 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 2 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 insert(int index, Object value);
  public abstract void insertLast(Object value);
  public abstract Object get(int index);
```

```
  public abstract int indexOf(Object value);
  public abstract int lastIndexOf(Object value);
  public abstract Object remove(int index);
  public abstract void removeLast();
  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 take();
  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;
public class ArrayQueue extends Queue {
  static final int MAX_SIZE = 100;
  Object[] queue = new Object[MAX_SIZE];
  int count = 0;
  ...
}
public class LinkedListQueue extends Queue {
          // aus Vorgabe
  List list = new at.jku.ssw.LinkedList();
  ...
}
```

```
Queue q = new ArrayQueue();
q.put("Lehrveranstaltung");
q.put("Praktische");
q.put("Informatik");
q.put("2");
q.take();
Out.print(q.size() + ": ");
Iterator it = q.iterator();
while (it.hasNext()) {
  Out.print(" " + it.next());
}
// Ausgabe 3: Praktische Informatik 2
```

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 *take*. Geben Sie die Sequenz der Zeichen an, die die Take-Operation zurückgibt.

Abzugeben ist: Java-Programm, Testfälle, gesuchte Zeichensequenz