Übung 2: Stack, Queue

Abgabetermin: 20.03.2012

Name: _____ **Matrikelnummer:** _____

Gruppe: □ **G1** Di 10:15 □ **G2** 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	□			□	

Aufgabe 1: Stack für Zeichen (12 Punkte)

Implementieren Sie einen Kellerspeicher für Zeichen, einmal mit einem Array in der Klasse *ArrayStack* und einmal als verkettete Liste in der Klasse *LinkedListStack*. Beide Klassen sollen die durch die abstrakte Klasse *Stack* gegebene Schnittstelle implementieren: *push* kellert ein Zeichen ein, *pop* kellert ein Zeichen aus, *size* liefert die Anzahl der Zeichen und *iterator* liefert einen Iterator (mit Schnittstelle *Iterator*) mit dem der Kellerspeicher von oben nach unten durchlaufen werden kann.

```
package at.jku.ssw;
public abstract class Stack {
  public abstract void push(char value);
  public abstract char pop()
    throws NoSuchElementException;
  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*.

```
package at.jku.students;
public class ArrayStack extends Stack {
  char[] stack = new char[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:

- 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(char value);
  public abstract void add(
    int index, char value);
  public abstract char get(int index);
  public abstract char remove(int index);
  public abstract int indexOf(char 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 Zeichen (12 Punkte)

Implementieren Sie eine FIFO-Warteschlange (First-in-first-out) für Zeichen, einmal mit einem Array in der Klasse *ArrayQueue* (zyklischer Puffer, Array mit fixer Größe) und einmal als verkettete Liste in der Klasse *LinkedListQueue*. Beide Klassen sollen die durch die abstrakte Klasse *Queue* gegebene Schnittstelle implementieren: *put* fügt ein Zeichen ein, *get* entnimmt ein Zeichen, *size* liefert die Anzahl der Zeichen und *iterator* liefert einen Iterator mit dem man die Warteschlange in FIFO-Reihenfolge durchlaufen kann.

```
package at.jku.ssw;                                          // Iterator & List
public abstract class Queue {                                // siehe Aufgabe 1
  public abstract void put(char value) throws Exception;
  public abstract char get() throws Exception;
  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;
  char[] queue = new char[MAX_SIZE];
  int count = 0;
  ...
}
public class LinkedListQueue extends Queue {
          // aus Vorgabe
  List list = at.jku.ssw.LinkedList();
  ...
}
```

```
Queue q = new ArrayQueue();
q.put('P'); q.put('I'); q.put('2');
Out.print(l.size() + ": ");
Iterator it = l.iterator();
while (it.hasNext()) {
  Out.print(" " + it.next());
}
// Ausgabe 3: P I 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 *get*. Geben Sie die Sequenz der Zeichen an, die die Get-Operation zurückgibt.

Abzugeben ist: Java-Programm, Testfälle, gesuchte Zeichensequenz