Prof. Dr. Volker Linnemann
Dr. Sven Groppe
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 8.2.2008

Vorlesung „Programmieren“

Wintersemester 2007/2008

13. Übungsblatt

1. Doppelt verkettete Listen (Programmieraufgabe) (22 Punkte)

Bei doppelt verketteten Listen enthält ein Eintrag nicht nur einen Verweis auf seinen nachfolgenden Eintrag, sondern auch einen Verweis auf seinen vorherigen Eintrag.

Realisieren Sie eine Klasse für doppelt verkettete Listen, welche folgende Schnittstelle implementiert:

```
/**
 * DoublyLinkedListInterface ist eine Schnittstelle
 * für verkettete Listen
**/
public interface DoublyLinkedListInterface<E> {

    /**
     * get gibt das Element an Position index zurück.
     * @param index Die Position des Elementes, welches zurückgegeben werden soll
     * @return Das Element an Position index.
    **/
    public E get(int index) throws IndexOutOfBoundsException;

    /**
     * search gibt die Position des ersten Vorkommens von dem Parameterobjekt in
     * der Liste zurück
     * @param value Das Parameterobjekt, nach dem gesucht wird
     * @return Die Position des ersten Vorkommens von dem Parameterobjekt value in
     *         der Liste
    **/
    public int search(E value);

    /**
     * delete löscht den Eintrag an der übergebenen Position und gibt den (Inhalt
     * des) Eintrag(es) zurück
     * @param index Position des Eintrages, das gelöscht werden soll
     * @return Inhalt des gelöschten Eintrages
    **/
    public E delete(int index) throws IndexOutOfBoundsException;

    /**
     * insertAfter fügt das Parameterobjekt value in die Liste nach dem Eintrag an
     * der Stelle index ein.
     * @param value  einzufügendes Objekt
     * @param index  Indexposition nach dem value eingefügt werden soll
     */
    public void insertAfter(int index, E value) throws IndexOutOfBoundsException;

    /**
     * insertHead fügt das Parameterobjekt value in die Liste als Kopfeintrag ein.
     * @param value  einzufügendes Objekt
     */
    public void insertHead(E value);
```

```
    /**
     * insertTail fügt das Parameterobjekt value in die Liste als Schwanzeintrag
     * ein.
     * @param value  einzufügendes Objekt
     */
    public void insertTail(E value);

    /**
     * size gibt die Anzahl der Einträge in die Liste zurück.
     * @return momentane Anzahl der Einträge der Liste
     */
    public int size();

    /**
     * isEmpty gibt zurück, ob die Liste kein Element enthält oder mehrere
     * @return true <=> kein Element in der Liste, false <=> ein oder mehr Elemente
     *         in der Liste
     */
    public boolean isEmpty();

    /**
     * showList liefert den Inhalt aller Elemente der Liste
     * hintereinander als Zeichenkette
    */
    public String showList();
}
```

Implementieren Sie zusätzlich eine Testmethode, die die folgende Situation einer Warteschlange an einer Kasse simuliert: Am Anfang sei die Kasse unbesetzt. Ein Kunde A, dessen Einkaufswagen Waren im Wert von 33 Euro beinhaltet, und danach ein Kunde B, dessen Einkaufswagen Waren im Wert von 21 Euro beinhaltet, gehen zur Kasse, woraufhin die Kasse besetzt wird und der Kunde A bedient wird. Danach kommen ein Kunde C (Waren im Wert von 4 Euro), ein Kunde D (Waren im Wert von 87 Euro) und ein Kunde E (Waren im Wert von 65 Euro) zur Kasse. Nachdem Kunde B bedient worden ist, entschließt sich Kunde D (für Warteschlangen eigentlich untypisch) die Schlange zu verlassen, um eine vergessene Ware im Wert von 8 Euro in den Einkaufswagen zu legen und sich an der Schlange hinten wieder anzustellen. Die Mitarbeiterin an der Kasse bedient nun alle Kunden und zählt ihre Einnahmen, die sie der Geschäftsführung mitteilt.

Bemerkungen:

- Dieses ist eine Programmieraufgabe. Bitte geben Sie die Lösung *sowohl* auf **Papier** *als auch* den Java-Quelltext in elektronischer Form per **Email** an Ihren Tutor ab.

Bemerkungen:

- Jede Seite soll oben rechts den Namen der Abgebenden und die Übungsgruppennummer (wichtig!) enthalten.
- Lösungen für die Übungsaufgaben sind (in der Regel) zu zweit abzugeben.
- Kommentieren Sie Ihre Lösungen! Besteht eine Lösung aus mehreren Zetteln, so sind diese zusammen zu heften. Bitte keine Hüllen, Mappen, o.ä..
- Bitte schicken Sie *Programmieraufgaben zusätzlich zur Abgabe auf Papier in elektronischer Form per Email* an Ihren jeweiligen Tutor.
- Kommentieren Sie ihren Quelltext bei Programmieraufgaben. Dabei sollen keine Trivialitäten kommentiert werden, also bitte keine Kommentare wie

  ~~`x=5; // Wir weisen nun der Variablen x den Wert 5 zu`~~

  sondern sinnvolle Kommentare, die Ideen des Quelltextabschnittes beschreiben oder auf Unteraufgaben (z. B. a), b), …) hinweisen.
- **Hinreichende Bedingung für die Zulassung zur Klausur:** 50% der erreichbaren Punkte bei jedem Übungszettel (bis auf zwei) und einmaliges Vorrechnen in der Übung
- **Zertifikatskriterium:** Das Bestehen der Klausur am Ende des Semesters

Abgabetermin: Freitag, 15.2.2008, nach der Vorlesung