CMPE160 PROJECT2

ARRAYLIST AND LINKLIST IMPLEMENTATION

SPECIFICATION:

In this project you are going to implement two classes named MyArrayList<E> and MyLinkedList<E>. They will be generic classes that can store any given type of object. The details about generic classes are given in Appendix. Both classes will implement MyIterator<E> interface which has the following methods:

• boolean hasNext(): returns true if the iteration has more elements.
• E next(): returns the next element in the iteration.
• void remove(): removes from the list the last element returned by the iterator.

The methods you are going to implement for MyLinkedList class are as follows:

• MyLinkedList(): Default constructor that takes no parameters and creates an empty MyLinkedList.
• MyLinkedList(E[] elements): Another constructor that takes an array as parameters and creates a MyLinkedList with the contents of the array.
• void add(E element): Adds an element to the front of the MyLinkedList.
• void append(E element): Adds an element at the end of the MyLinkedList.
• E get(int i): Gets the item at position i, returns null if no such item exists.
• E remove(int i): Removes and returns the item at position i, returns null if no such item exists.
• void clear(): Removes all elements.
• boolean contains(E element): returns true if the list contains the item, false otherwise. Do not forget to use equals method here.
• boolean isEmpty(): returns true if the list is empty, false otherwise.
• MyIterator<E> iterator(): returns a MyIterator over the elements in this list.
• int size(): returns the number of elements in this list.
• E set(int position, E element): replaces the element at the given position with the given element, and returns the replaced element (null if there is no such element at the given position).

The methods you are going to implement for MyArrayList class are as follows:

• MyArrayList (): Default constructor that takes no parameters and creates an empty MyArrayList with initial capacity of 10.
• MyArrayList (int capacity): Default constructor that takes no parameters and creates an empty MyArrayList with given capacity.
• MyLinkedList(E[] elements): Another constructor that takes an array as parameters and creates a MyLinkedList with the contents of the array. The capacity of MyLinkedList should be twice the size of the array.
• void add(E element): Adds an element to the front of the MyArrayList.
• void append(E element): Adds an element at the end of the MyArrayList.
• E get(int i): Gets the item at position i, returns null if no such item exists.
• E remove(int i): Removes and returns the item at position i, returns null if no such item exists.
• void clear(): Removes all elements, and resets the capacity to 10.
• boolean contains(E element): returns true if the list contains the item, false otherwise. Do not forget to use equals method here.
• boolean isEmpty(): returns true if the list is empty, false otherwise.
• MyIterator<E> iterator(): returns a MyIterator over the elements in this list.
• int size(): returns the number of elements in this list.
• E set(int position, E element): replaces the element at the given position with the given element, and returns the replaced element (null if there is no such element at the given position).

Besides these public methods, you may add additional private methods if you need them, and also methods for the MyIterator interface. You also need to define a new class Node<E> that stores the data of an element of MyLinkedList together with pointers to the next and previous elements.

**What to deliver:** You will submit only MyArrayList.java, MyLinkedList.java, Node.java and your report in a zip archive and mail them to cmpe160.projects submission@gmail.com with subject “CMPE160 Project2” (without quotation marks) until 11/05/2011, 17.00. The mail address given is for project submissions, if you have questions use my regular e-mail address. Please write your name and number as comments to the top of each java file. Clear code with meaningful variable names and good commenting is requested. Your usage of javadoc will also be evaluated so try to use javadoc as much and as reasonable as possible. Your report should be short, discussing the relationships between your classes, and other stuff that cannot be understood from your code/comments.

**APPENDIX:** Generic classes like LinkedList, ArrayList provide a means for describing a data structure in a type-independent manner. We can then instantiate type specific objects of the generic class. For instance, you have implemented a Queue class that holds Vehicle objects. You can declare a generic Queue class that stores any kind object as follows:

```java
public class Queue<E> {

    protected ArrayList<E> list;

    public Queue() {
        list = new ArrayList<E>();
    }

    public void add(E element) {
        list.add(element);
    }

    public E getFront() {
        if (list.isEmpty()) return null;
        return list.remove(0);
    }
}
```
public boolean isEmpty() {
    return list.isEmpty();
}

This class can be used to store Vehicle objects by declaring it as:

Queue<Vehicle> vehicleQueue = new Queue<Vehicle>();

It can also be used to store Event objects by declaring it as:

Queue<Event> eventQueue = new Queue<Event>();