Lab Eight: The Deque Data Structure

October 30, 2018

Introduction

A double-ended queue, called a *deque* (pronounced "deck"), is a list data structure that allows insertions and deletions at both ends of the list. No insertions or deletions are allowed in the middle of the list. Here is a possible interface for a deque data structure:

```
public interface Deque<E> {

  public void addFirst(E element);
  public E removeFirst();
  public E getFirst();
  public boolean removeFirstOccurrence(Object obj);
  public void addLast(E element);
  public E removeLast();
  public E getLast();
  public boolean removeLastOccurrence(Object obj);
}
```

Doubly-Linked Lists

A doubly-linked list is a generalization of the linked list data structure that consists of a data field and *two* link fields: `previous` and `next`:

```
  private static class DNode<E> {

    private E data;
    private DNode<E> previous;
    private DNode<E> next;

    public DNode(E data, DNode<E> previous, DNode<E> next) {
      this.data = data;
      this.previous = previous;
      this.next = next;
    }

    public DNode(E data) {
      this(data, null, null);
    }
    public DNode() {
      this(null, null, null);
    }
  }
```

This allows you to create linked lists which can be traversed in either direction. One advantage over singly-linked lists is that you don't need to know where the previous node is to delete the current node — you have a previous pointer in the current node that points to it.

The LinkedDeque Class

For this lab, you are to implement the `Deque` interface using a doubly-linked list. The class that implements the interface is to be called `LinkedDeque`. The `Deque` interface and skeleton code for the `LinkedDeque` implementation appears in the provided java source files. Study these files carefully. Complete the code by implementing the following methods:

1. public void addFirst(E element);
2. public E removeFirst();
3. public E getFirst();
4. public boolean removeFirstOccurrence(Object obj);
5. public void addLast(E element);
6. public E removeLast();
7. public E getLast();
8. public boolean removeLastOccurrence(Object obj);

Submission

lease submit your lab via Moodle by 11:55pm Friday, November 2.