University of Toronto
csc148S - Introduction to Computer Science, Spring 2000

Midterm Test

Wednesday March 1, 2000

Duration: 50 minutes

Aids allowed: None

Family Name:	______________	Given names:	______________
Student #:	______________	Tutor:	______________

- There are 5 pages, including this one. The test is out of 20 marks and the value of each question is provided; please use this information to manage your time effectively.

Part A:	______	/ 10
Part B:	______	/5
Part C:	______	/ 5
Total	______	/ 20

Part A [10 marks in total]

We are interested in developing an ADT called `SkiLiftQueue` for modelling the queue at a ski lift. In this type of queue, *people arrive in the line one at a time but leave the line in groups.* A lift has a chair size which is the maximum number of people that can leave at once. In our model, the leaving groups are always as big as possible. Here is an example; Consider a lift with a chair size of 4. When a call is made to dequeue and there are 4 or more people in line, 4 people will leave the queue. When dequeue is called with 4 or fewer people in line, they will all leave the queue. If dequeue is called on an empty queue, the behaviour is unspecified.

1. The `LinkedQueue` class implements the standard `Queue` interface discussed in lecture and used in assignments 1 and 2. Complete the `LinkedSkiLiftQueue` class started below. Notice that the implementation makes use of a `LinkedQueue` object. You may not change any of the provided code.

```
class LinkedSkiLiftQueue implements SkiLiftQueue {
    int chairSize;
    LinkedQueue theQueue;

    LinkedSkiLiftQueue(int chairSize){
        theQueue = new LinkedQueue();
        this.chairSize = chairSize;
    }

    public void enqueue (Object o) {
        // fill in the body here




    }

    public Object[] dequeue() {
       Object[] result = new Object[chairSize];
       // fill in the remainder of the body here














    }
}
```

2. One of the methods needs a precondition. Write an appropriate precondition here and indicate which method it is for by circling one of the following:
   `LinkedSkiLiftQueue` `enqueue` `dequeue`

3. Quad-chairs are very popular. Write an overloaded constructor which does not have a chair-Size argument but uses the default value of four.

Part B [5 marks]

Consider the `MedSet.java` code for assignment three. You wrote a constructor with no parameters (`MedSet()`). It might have been useful to instantiate a `MedSet` object with an initial set of `Comparable` objects instead of instantiating an empty set and adding each item separately.
Write an overloaded constructor for `MedSet` objects that gives this behaviour. You can assume that you have working code for the other methods in the class.

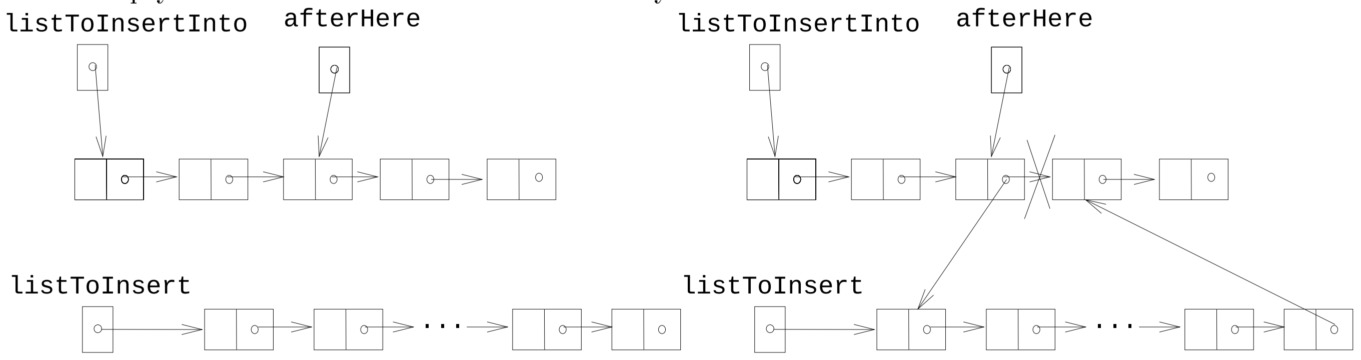
```
class MedSet implements SetWithMedian {

    // instance variables are not shown
    // HINT: You should not need to directly access any of these anyway

    public MedSet() {
        // details omitted
    }
    public Comparable median() {
        // details omitted
    }
    public void insert(Comparable c) {
        // details omitted
    }

    public MedSet(Comparable[] initialSet) {
        // you complete this one
```

Part C [5 marks]

Assume that we have a node class defined as follows:

```
public Class Node {
    public Object data;
    public Node next;
}
```

We have two linked lists of `Node` objects and would like to insert one into the other at the point specified. Write a code fragment to do this. The picture on the left shows the situation before your code is called and the one on the right gives the resulting structure. Notice that the number of nodes in the `listToInsert` is unknown. It could even be an empty list. `listToInsertInto` cannot be empty. It at least has one node referenced by `afterHere`.



```
Node listToInsert, afterHere, listToInsertInto;

// Code to initialize and set up the lists is not shown...

// Now write your code fragment:
```

End of Test