The University of Chicago
Department of Computer Science

CMSC 15200 – Introduction to Computer Science 2
Summer Quarter 2006
Lab #3 (08/09/2006)

Name:

Student ID:

Lab Instructor: Borja Sotomayor

Do not write in this area

1	2	**TOTAL**

Maximum possible points: 35

Exercise 1 <<25 points>>

You will implement a queue data structure, as described in class. The structure and function declarations are the following (queue.h in the homework files):

```
struct ListNode {
       int data;
       ListNode *next;
};

struct Queue {
       ListNode *head;
       ListNode *tail;
};

// Creates a new queue
void createQueue(Queue &q);

// Enqueues a new element
void enqueue(Queue &q, int data);

// Dequeues a new element
// Assumes queue is non-empty
int dequeue(Queue &q);

// Returns the value of the first element in the queue,
// without dequeuing
int peek(Queue &q);

// Returns true if the queue is empty
bool isEmpty(Queue &q);

// Prints out the contents of the queue
void printQueue(Queue &q);

// Destroys queue
void destroyQueue(Queue &q);
```

Don't reinvent the wheel! You should be able to implement this queue data structure reusing practically all the code from the double-ended list seen in class.

To test your list implementation, a main_queue.cpp is provided in the lab files. Running this program with a correct queue implementation should yield the following:

```
1 2 3 4 5 6 7 8 9
First element is 1
1 2 3 4 5 6 7 8 9
Dequeued element 1
2 3 4 5 6 7 8 9
Dequeued element 2
3 4 5 6 7 8 9
```

```
Dequeued element 3
4 5 6 7 8 9
Dequeued element 4
5 6 7 8 9
Dequeued element 5
6 7 8 9
Dequeued element 6
7 8 9
Dequeued element 7
8 9
Dequeued element 8
9
Dequeued element 9
Queue is empty!
1 2 3
Queue is empty!
```

Exercise 2 <<10 points>>

Add the following function to the linked list implementation seen in class (available on the course website, in the "Files" section):

```
void clone(List &src, List &dst);
```

This function creates a *deep* copy (or "clone") of list *src* and stores it in *dst*. This does *not* mean that you simply have to make the head of *dst* point to the first element of *src*. You have to ***copy*** the contents of *src* and place them in list *dst*. This means that, for example, if we were to modify the contents of *dst* (after doing a clone operation), this will not affect the contents of *dst.* You can assume that *dst* has been properly initialized (i.e. the head is NULL or points to a ListNode). However, if *dst* is not empty, you will have to destroy its contents before performing the clone.