CISC 220 – 010 Homework #5 Due Monday, October 9, 2000

For this week's homework, we will be working with a new linear ADT - the queue. Remember that a queue is a restricted form of a list in which we can remove or retrieve from one end, and insert at the other end – no access to interior elements. There are several ways to implement a queue ADT, and we'll do a few of them here. When it comes time to test your implementation, use the following driver program.

```
// driver for queue class
#include "queue.h"

int main() {
   Queue<int> myCustomers;

   // put a few things in the queue
   for(int k = 0; k < 10; k ++ )
       myCustomers.append( k );

   // now service each customer
   int value;
   while( !myCustomers.empty() ) {
      myCustomers.retrieve( value );
      cout << "Now serving customer number " << value << endl;
      myCustomers.serve();
      if( value%2 )
          myCustomers.append( value + 1 );
   }

   // queue is empty…quittin' time
   cout << "done." << endl;

}
```

Part A) Implement the queue ADT directly using a linked structure – that is, make a node class (as in the linked list), and a Queue class. In the Queue class, have a head and a tail pointer to make it easy to append, and service elements. There should be the following six member functions: a constructor and destructor, append, serve, retrieve, and empty.

Put the class (remember it has to be a template class) in file called queue1.h – along with the function definitions. Once this implementation works, make a script file (HW-5-A.scr) in which you cat queue1.h, compile and run.

Part B) Implement the queue ADT directly using an array structure – that is, store the elements of the queue in an array the size of which is a constant. There should be the following seven member functions: a constructor and destructor, append, serve, retrieve, empty, and full.

Put the class (remember it has to be a template class) in file called queue2.h – along with the function definitions. Once this implementation works, make a script file (`HW-5-B.scr`) in which you cat queue2.h, compile and run.

Part C) Implement the queue ADT as a restricted List using the inheritance mechanism. Your class declaration starts off as follows.

```
template <class T>
class Queue : private List<T> {
    public:
         // put queue member functions here

};
```

Here, we inherit the implementation, but hide the interface of the `List` class, replacing it with a new `Queue` interface. The file `queue.h` will #include `List.h`, and also have the declaration and definitions for the `queue` class. The Queue functions should use the List functions where needed. Once this code works, make a script file (`HW-5-C.scr`) in which you cat `queue3.h`, and compile and run the driver.

Extra Credit: Implement the queue class in a circular array as described in section 3.3 of the textbook.

What to hand in: Hand in the three script files generated above (and the extra credit if you did it), stapled together as one set. (Don't forget to print your name on the front page!)