CISC 220 – 010 Homework #1 Due Friday, September 5, 2000

The following exercises will give you practice with 1) making and using template functions, 2) designing and implementing some simple classes, including operator overloading, and making multi-file applications – with header files and multiple source files. You will need your notes from the first two lectures.

Part A) Create a header file (called `utilities.h`) and put the prototypes for the two template functions `swap` and `bubble_sort` from class. Then create a source file, called `utilities.cc`, containing the code for these two functions. Make sure that the bubble sort function uses the swap function, instead of interchanging the elements as in the class example. You can test for *syntax* errors with this compile command:

```
CC –c utilities.cc
```

Fix any errors before proceeding to the next part.

Part B) Test the code in Part A) by writing a small main *driver* – a *driver* is simply a little function designed to take the given code on a test drive ☺ - just to see if it works. Your driver should instantiate a few small arrays, sort each one, and output them; it's a simple matter to verify that the codes work. Fix any errors before proceeding to the next part. In this case, the relevant compile command would be

```
CC sort-driver.cc  utilities.o
```

When this is a working program, make a script file (`HW-1-B.scr`) containing the three files, a compile, and a run of the program.

Part C) Design and implement a class, called Person, with the following attributes (private)

```
name      a string of 30 chars
idNumber  an unsigned long int
gender    a single char
status    a single char
```

And with the following minimal set of member functions (sometimes called methods, or the object's interface)

```
Constructor(s)
I/O – use overloaded >> and <<
A greater than > operator (testing on idNumber)
```

Make two files – one is the header file, called `person.h`, containing the class definition, and the other is the implementation file, called `person.cc`, containing the code for the member functions. Test for syntax errors with the compile command

```
CC –c person.cc
```

Then test the class by making a small driver. Once this works, make a script file (`HW-1-C.scr`) containing the three files and a run to hand in.

Part D) Write a small application in which you do the following:

Make an array of 50 of type Person
Read data into this array from file person.data (until eof)
Invoke the bubble sort
Print the resulting array, sorted in order, by idNumber.

Compile this application using the command

```
CC homework1.cc person.o utilities.o
```

Once this program works, make a script file (`HW-1-D.scr`) containing the source code for homework1.cc, a compile, and a run.

What to hand in: Hand in the three script files generated above, stapled together as one set. (Don't forget to put your name on the front page!)