CSY2006 Week 20

Lab Exercises:

1. Write your own version of a class template that will create a static stack of any data type. Demonstrate the class with a driver program.

2. Write your own version of a class template that will create a dynamic queue of any data type. Demonstrate the class with a driver program.

3. Extend the MathStack class discussed in the lecture (sample programs) to implement the following additional functions:
   mult – Pops the two values off the stack, multiplies them and pushes their product onto the stack.
   div – Pops the top two values off the stack, divides the second value by the first, and pushes the quotient onto the stack.
   addAll – Pops all values off the stack, adds them and pushes their sum onto the stack.
   multAll – Pops all values off the stack, multiplies them, and pushes their product onto the stack.
   Demonstrate the class with a driver program.

4. Design an inventory class that stores the following members:
   serialNum: An integer that holds a part's serial number
   manufactDate: A member that holds the date the part was manufactured
   lotNum: An integer that holds the part's lot number
   The class should have appropriate members for storing data into and retrieving data from these members. Next, design a queue class that can hold objects of the above type. Design a program that uses the queue class. The program should have a loop that asks the user if he or she wishes to add a part to the inventory, or take a part from the inventory. The loop should repeat until the user is finished.
   If the user wishes to add a part to the inventory, the program should ask for the serial number, data of manufacture and lot number. The data should be stored in an inventory object, and pushed onto the queue. If the user wishes to take a part from the inventory, the program should dequeue and display the contents. When the user finishes the program, it should display the contents of the member values of all the objects that remain on the queue.

5. Consider the class stackType in Sample Programs(Malik). Add and implement the following function:
   void reverseStack(stackType<Type> &otherStack);
   This operation copies the elements of a stack in reverse order onto another stack.

6. Write definitions of the functions to overload the assignment operator and copy constructor for the class queueType in Sample programs (Malik). Also, write a program to test these operations.