CS 2413 001 Summer 2000 Homework #4

Quiz to be held in class 1:20pm Friday 7 July 2000

You will need extra sheets of notebook paper to write your answers on.

1. **Define** each of the following: Abstract Data Type, Abstract Base Class and Data Structure.
2. What is the **difference** between a stack and a queue?
3. Give the time complexity (big-**O**) on a structure of *n* elements for the following operations, or mark **NA** if the operation is not applicable to the data structure.

	`StackArray`	`StackLinked`	`QueueArray`	`QueueLinked`
`size`				
`isEmpty`				
`peek`				
`push`				
`pop`				
`front`				
`enqueue`				
`dequeue`				

4. Suppose you have two stacks of characters, `stack1` and `stack2`. If `stack1` is empty and `stack2` currently has `’A’`, `’P’` and `’M’` on it (top to bottom), what will be the state of the stacks after the following sets of operations? **Show your work.**

   (a) `stack1.push(’B’); stack2.pop(); stack2.pop();`

   (b) `stack1.push(stack2.peek()); stack2.pop();`
   `stack1.push(’C’); stack.push(’D’);`

   (c) `stack1.push(stack2.peek()); stack1.push(stack2.peek());`
   `stack1.push(stack2.peek());`

   (d) `while (!stack2.isEmpty())` {
   `stack1.push(stack2.peek()); stack2.pop();` }

5. Suppose that, in the above question, the data structures are queues rather than stacks, that the second queue is in the above order front to rear rather than top to bottom, and that `peek`, `push` and `pop` are replaced by `front`, `enqueue` and `dequeue`, respectively. What will be the state of the queues after each of the above sets of operations? **Show your work.**
6. A *palindrome* is a string that reads the same backwards as forwards; for example, “eve,” “noon,” “live evil.” **Write a function** that uses a stack to determine whether a string is a palindrome.
7. **Write a function** that takes two incoming arguments, an output stream and a `String`, and outputs to the output stream the characters of the string in this order:

   - all of the upper case letters, in the order in which they appear in the string;
   - all of the lower case letters, in the order in which they appear in the string;
   - all of the digits, in the order in which they appear in the string.

   Your function should use three queues, one for each kind of character. You may use the C++ standard library functions `isupper`, `islower` and `isdigit`, declared in `ctype.h`.

References

S. Rhadakrishnan, L. Wise & C. N. Sekharan, *Object-Oriented Data Structures Featuring C++*, 1999.

M.R. Headington & D.D. Riley, *Data Abstraction and Structures Using C++*, D.C. Heath & Co., Massachusetts, 1994.