LAB #6 – Linked Structures

Remember, you will not receive lab credit if you do not get checked off before leaving each lab. Once you have a zero on a lab, then it cannot be changed because we have no way to know if you were there or not! If you need to get last week’s lab graded, you need to show your program to the TAs within 10 minutes of getting to this lab.

You will implement a simplified stack and queue. Neither is as complex as the list data structure we’ve looked at in class. Unlike a list, you only insert at one place, take off from one place (albeit maybe a different place), get the next value to come off, and test if it’s empty. Nothing is ever done to any internal node. We will make it even simpler; you will add items and combine removing and getting the value. 2 functions. That’s it!

1. Stack-like behavior. A stack is a first in last out (FILO) structure. So the top of the stack will be the last item entered. You are not required to implement any other features of the formal Stack data structure. Be careful when you research this. You could make your program more difficult.

You will create a singly linked Stacknode. You will need to add an element and to take off an element. You should have these functions for your Stack:

```c
void add(char Value)
char remove()
```

The remove() function will return the data in the top node. It will then remove the node and adjust pointers.

This is all that is required.

2. Queue-like behavior. A queue is a first in first out (FIFO) structure. You will need a pointer to the front and to the back of the queue. You are not required to implement any other features of the formal Queue data structure.

You will create a doubly linked Queuenode. You will need a front pointer and a rear pointer. You will only add elements at the back node and only take off from the front node. You should have these functions for your Queue:

```c
void add(char Value)
char remove()
```

The remove() function will return the data in the front node. It will then remove the node and adjust the front pointer.

You can implement the stack, queue, stackNode, and queueNode as either a struct or as a class. Remember that the stack only needs a pointer to top while the queue needs pointers to front and back.

NOTE: De-allocate memory as appropriate.