Problem 2

1. My implementation achieves O(1) time for remove() because every time I remove, I am only removing the first node of my data structure. I have a pointer called First that is always pointing to the fist Node of this structure, so to remove a node, the code does not need to traverse through the entire list, it only has to take the value of the pointer and delete it. This remains constant, which is why it achieves O(1) time.

Code:

**int** remove(){

**int** value;

value = first.data;

first = first.next;

**return** value;

}

For the getValue(index i) function, it achieves O(1) time because I never need to traverse through the entire list, only to the point where the index tells you to. I set a variable named k that equals 0, and every time I want to get the value at a specific index ‘i’, the code will know where it is located because k will constantly be increasing until it is less than the value of ‘i’. This way, we never have to go through the entire list, but only the specific number that we want to see.

Code:

**while**(a.next!= **null** && k < i){

a = a.next;

k++;

}

Entire code:

**int** getValue(**int** i){

Node a = first;

**int** k = 0;

**if**(i ==1){

**return** first.data;

}

**if**(i > total){

**return** -1;

}

**while**(a.next!= **null** && k < i){

a = a.next;

k++;

}

**return** a.data;

}