**Name:Zihan Xia**

**UCLA Id Number**: 205838465

A brief description of notable obstacles you overcame:

For pointer and linked list, it is quite confusing at first since I need to draw graphs in my mind to kind of figure out what I am supposed to do with each step and how should i manage the pointers such as next and head. I overcome it by drawing graphs. Also, we need to be quite mindful of memory leaking while dealing with dynamic data type.

#include <iostream>

#include <string>

#include <cassert>

#include "List.h"

#include "ListNode.h"

**int** main(){

**using** **namespace** std;

**using** **namespace** cs32;

List test;

//build the linked list

test.addToRear("A");

test.addToRear("B");

test.addToRear("C");

test.addToRear("D");

test.addToRear("E");

test.addToRear("F");

test.addToRear("G");

test.addToRear("H");

test.addToFront("I");

assert(test.size() == 9);

//test if the position of the node is right

assert(test.position("B") == 2);

assert(test.position("D") == 4);

assert(test.position("I") == 0);

assert(test.position("H") == 8);

assert(test.position("M") == -1);

//test min

assert(test.min() == "A");

assert(test.before("D", "E") == **true**);

assert(test.before("D", "A") == **false**);

assert(test.before("M", "C") == **false**);

assert(test.before("C", "M") == **false**);

string element;

assert(test.get(1, element) == **true**);

// element should be A after get function

assert(element == "A");

assert(test.get(2, element) == **true**);

assert(element =="B");

assert(test.get(12, element) == **false**);

assert(test.min() == "A");

test.removeAllBiggerThan("B"); //should remove strings larger than B, which leaves us 2 numbers after

assert(test.size() == 2);

assert(test.position("A") == 0);

assert(test.position("B") == 1);

// cout << "all tests passed!" << endl;

**return**( 0 );

}