**Program 2 – stringLinkedList**

**Assignment Objective:** To learn how to build a linked list ADT.

**Requirements:**

* Create a class called **node** with the following members:
  + Private members:
    - string text;
    - node \*next;
    - constructor … node(string text = “”, node \*pn = NULL)
* Create a class called stringLinkedList with the following members
  + Private members:
    - node \*first, \*last;
    - int listCount // to record the number of entries in the list
    - int getIndex(string text, node \*pn, int index) const; // recursion helper
    - void printIt(node \*pn, int index) const; // recursion helper
    - void clear(node \*pn); // recursion helper
  + Public members:
    - stringLinkedList() // causes the object to be initialized (first=last= NULL and count = 0).
    - ~stringLinkedList() // deletes the dynamically allocated nodes in the list.

// MODIFIERS

* + - bool insert(string text) // inserts the string at the beginning of the list. Returns whether the string was inserted.
    - bool add(string text) // appends the string to the end of the list. Returns whether the string was added.
    - bool insertAt(int index, string text) – inserts the string at the index “index”, presuming 0<= index <= length(); entries at that position need to shift right. Returns true if the entry was inserted; false otherwise.
    - bool deleteAt(int index, string &text) – deletes the entry at the given index. Returns true if the index was within 0 <= index < length(); returns false otherwise. If the delete takes place, it returns the string in the deleted node via the reference parameter “text”
    - bool readAt(int index, string &text) const – same as deleteAt(), but does not delete the entry.
    - void clear() // causes the list to be emptied; all dynamically allocated memory should be deleted. **This must be implemented with recursion, not a loop.**

// NON-MODIFIERS

* + - int getIndex(string text) const // returns the first position at which the value v was found; otherwise returns -1. **This must be implemented with recursion, not a loop.**
    - void printIt() const // causes the list to be printed; for each line, print the index and the string at that index. **This must be implemented with recursion, not a loop.**
    - int count() const – returns the number of entries in the list, listCount.
* First submission: The highlighted portions shall be submitted as described below. Note, to do any testing, there should be a “stub” for each of the functions that are not yet completed.
* Final submission: The totality of the assignment shall be submitted as described below.
* Demonstrate that the sBST data structure works:
* Demonstrate that the stringLinkedList data structure works with the provided p2m.cpp driver file, capturing the output in p2Output.txt. This is done by the following commands:

g++ p2.cpp p2m.cpp -o p2

./p2 p2Input.txt > p2Output.txt

* The p2Output.txt file should be identical to the output found in p2CorrectOutput.txt.
* **Deliverables:**
  + Into D2L, put a **zip file** containing p2.cpp, p2m.cpp, p2.h, and the p2Output.txt
  + In class: a printing of p2.h, p2.cpp, and p2Output.txt, in that order. Do not print the p2m.cpp file.