CSCI 2270 Lecture Notes

1/30/19

Say we have an array with 1000 of its elements populated (Length 1001). This array is sorted. We want to insert an element and keep the array sorted. Without a linked list, we must move every element larger than the value we are inserting to the right one place. This takes 1000 operations. Very inefficient!

* The Linked List Node

struct Node{

string key;

Node \*next; // self referential pointer that points to its own type

};

* + recall that can have a pointer to a struct type
    - Node \*p1;

Node \*p1 = new Node;

Node \*p2 = new Node;

p1 -> key = “llama”;

p2 -> key = “donkey”;

p1 -> next = p2;

p2 = new Node;

cout << p1 -> next -> key << endl;

* + this cout statement will print “donkey”

p1 -> next -> next = p2; //links up the next struct

* The Linked List Class
  + Outside of class, define a struct for a Node (singly linked)
    - two membes
      * 1. key
      * 2. self ref- pointer
  + Define SLL (singly linked list) class
    - private data members
      * head
      * tail
    - public functions
      * constructor – no parameters
        + set pointers to null
      * destructor
        + gets called when object goes out of scope
        + use it to deallocate all the linked nodes
      * Node \*search(string sKey)
        + locate a node with given value
        + return pointer
      * void appendNode(string newItem)
        + add new node to the end
      * void insert(string afterMe, string newItem)
      * void displayList()
        + print contents, starting at the head
        + traverse list
        + display every node key