5/14 CS 16 Lecture 12

Infinite Inclusion

When you recursively include the same header file recursively.

Fix this by using #ifndef and #define at the top of the file and #endif at the end of the file.

See the code\_from\_class to see an example. (Remove the #ifndef, #define, and #endif to see the error).

Linked Lists – Not on the midterm

More on dynamic memory management (Using the heap)

Basically a review

Beginning of class, reviewed memory leaks and dangling pointers.

Linked List is a solution to the issue of arrays. Arrays are an issue because if you want to insert a value in the middle of an array you have to move all the elements after where you need to insert to the right.

Linked Lists are not contiguous in memory. Easy to add elements anywhere in the array.

Example of insertions into a linked list:

Sample File:

#include <iostream>

#include <string>

using namespace std;

struct LLNode{

 int data;

 LLNode\* nextNode;

}; *// a linked list*

int **main**()

{

 *// Create a linked list: (123,456,567,789) then change it to (115,123,234,456,567,737,789,999)*

 LLNode \*head = new LLNode; *// This keeps track of the start of the list*

 LLNode \*zero = new LLNode;

 LLNode\* one = new LLNode;

 LLNode \* two = new LLNode;

 LLNode \*three = new LLNode;

 LLNode\* tail = new LLNode;

 *// First set the data for all the elements*

 zero->data = 123;

 one->data = 456;

 (\*two).data = 567; *// Notice we can use the -> or (\*). to access*

 three->data = 789;

 *// Now lets connect the linked list*

 head->nextNode = zero;

 (\*zero).nextNode = one;

 one->nextNode = two;

 two->nextNode = three;

 three->nextNode = NULL; *// This one is the end of the list so we set it to NULL*

 tail-> nextNode = three;

 cout << "Printing the new linked list: \n\n";

 for(LLNode\* i = head->nextNode; i != NULL; i = i->nextNode){

 if(i == head->nextNode) cout << "Linked list: ";

 cout << i->data << "->";

 if(i->nextNode == NULL) cout << 'X';

 }

 cout << "\n\n";

 *//Now lets modify the linked list to be (115,123,234,456,567,737,789,999)*

 LLNode\* aNode0 = new LLNode; *// This will later store 115*

 LLNode\* aNode1 = new LLNode; *// This will later store 234*

 LLNode\* aNode2 = new LLNode; *// This will later store 737*

 LLNode\* aNode3 = new LLNode; *// This will later store 999*

 *// We could set the nextNode first, but we're doing it this way to better follow the logic*

 aNode0->data = 115; *// Now this stores 115*

 aNode1->data = 234; *// Now this stores 234*

 aNode2->data = 737; *// Now this stores 737*

 aNode3->data = 999; *// Now this stores 999*

 *// Now set the head to 115 instead*

 head->nextNode = aNode0;

 aNode0->nextNode = zero; *// Set the node we are inserting's next node to the node after head*

 cout << "\nPrint after inserting 115\n\n" ;

 for(LLNode\* i = head->nextNode; i != NULL; i = i->nextNode){

 if(head->nextNode == i) cout << "Linked list: ";

 cout << i->data << " ";

 if(i->nextNode == NULL) {

 cout << "\n\n";

 }

 }

 *// Now put 234 in between 123 (zero) and 456 (one)*

 aNode1->nextNode = zero->nextNode; *// Node after 234 is 456 now*

 zero->nextNode = aNode1; *// Node after 123 is 234 now*

 cout << "\nPrint after inserting 234\n\n" ;

 for(LLNode\* i = head->nextNode; i != NULL; i = i->nextNode){

 if(head->nextNode == i) cout << "Linked list: ";

 cout << i->data << " ";

 if(i->nextNode == NULL) {

 cout << "\n\n";

 }

 }

 *// Now put 737 in between 567 (two) and 789 (three)*

 aNode2->nextNode = two->nextNode; *// Node after 737 is 789 now*

 two->nextNode = aNode2; *// Node after 567 is 737 now*

 cout << "\nPrint after inserting 737\n\n" ;

 for(LLNode\* i = head->nextNode; i != NULL; i = i->nextNode){

 if(head->nextNode == i) cout << "Linked list: ";

 cout << i->data << " ";

 if(i->nextNode == NULL) {

 cout << "\n\n";

 }

 }

 *// Now put 999 after 789 (three)*

 three->nextNode = aNode3; *// Node after 789 is 999 now*

 aNode3->nextNode = NULL; *// 999's next node indicates the end of the linked list*

 cout << "\nPrint after inserting 999\n\n" ;

 for(LLNode\* i = head->nextNode; i != NULL; i = i->nextNode){

 if(head->nextNode == i) cout << "Linked list: ";

 cout << i->data << " ";

 if(i->nextNode == NULL) {

 cout << "\n\n";

 }

 }

 return 0;

}

Output:

Printing the new linked list:

Linked list: 123->456->567->789->X

Print after inserting 115

Linked list: 115 123 456 567 789

Print after inserting 234

Linked list: 115 123 234 456 567 789

Print after inserting 737

Linked list: 115 123 234 456 567 737 789

Print after inserting 999

Linked list: 115 123 234 456 567 737 789 999