CIS 200 - Lab 4 ANs

Fall 2016

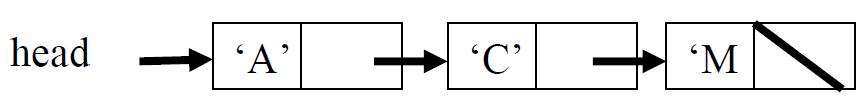
Consider that individual nodes in an unsorted linked list have the following definition:

|  |
| --- |
| struct ListRec {  char value;  ListRec\* next; }; |

# Task 1

Without implementing a separate class to maintain the list, simply create a simple list directly in the main function. You should create a pointer to point to the head of the list, and name that pointer head.

Use the previously stated definition of a node, and create the following list in main:



In the above visual depiction, the means that the next data member points to NULL (or nullptr.)

int main() {

ListRec \* head = new ListRec;

head->value = 'A';

head->next = new ListRec;

head->next->value = 'C';

head->next->next = new ListRec;

head->next->next->value = 'M';

head->next->next->next = nullptr;

print(head);

cout << endl;

return 0;

}

# Task 2

Write a void function in the same file as main to print out a linked list, given that the head is passed into the function as an argument. The prototype (and subsequently, the header) of the function should look like this:

|  |
| --- |
| void print(ListRec\* listHead) {  //print out the elements in the list } |

void print(ListRec\* listHead)

{

ListRec \* ptr = listHead;

while (ptr != nullptr) {

cout << ptr->value << " ";

ptr = ptr->next;

}

}



# Task 3

Write another function that takes two parameters:

* The head of a list to be copied
* The head of another list that will contain the copy of the first

The function perform a deep copy. Recall that with a deep copy, you must create a new node and copy over the value from the corresponding node in the list being copied to the list that will contain the copy.

The function prototype/header is as follows:

|  |
| --- |
| void deepCopy(ListRec\* oldListHead, ListRec\* & newListHead) {  //perform a deep copy from old list to new list } |

# Task 4

Write the main function to perform a test of each of the above tasks. It must create the list (task 1), call the print function (from task 2), and the deepCopy function (from task 3.)

From main, you should :

1. make a copy of the original list

2. change the data in first node of the original list

3. call the print function on **both** the original list you created, and the copied list to verify that the deep copy worked as expected.

/\*

Lab 4

Bruce Elenbogen

\*/

#include <iostream>

using namespace std;

//Task 1

struct ListRec

{

char value;

ListRec\* next;

};

//Task 2

void print(ListRec\* listHead)

{

ListRec \* ptr = listHead;

while (ptr != nullptr) {

cout << ptr->value << " ";

ptr = ptr->next;

}

}

void deepCopy(ListRec\* oldListHead, ListRec\* & newListHead)

{

ListRec \* oldPtr = oldListHead;

ListRec \* newPtr ;

newListHead = nullptr;

while(oldPtr != nullptr){

if (newListHead == nullptr) {

newListHead = new ListRec;

newListHead->value = oldPtr->value;

newListHead->next = nullptr;

newPtr = newListHead;

}

else {

newPtr->next = new ListRec;

newPtr->next->value = oldPtr->value;

newPtr->next->next = nullptr;

newPtr = newPtr->next;

}

oldPtr = oldPtr->next;

}

}

int main() {

ListRec \* head = new ListRec;

ListRec \*copyHead;

head->value = 'A';

head->next = new ListRec;

head->next->value = 'C';

head->next->next = new ListRec;

head->next->next->value = 'M';

head->next->next->next = nullptr;

print(head);

cout << endl;

deepCopy(head, copyHead);

print(copyHead);

cout << endl;

return 0;

}

