CS 323\_33 Programming Language: C++

Project #1 Linked List Middle Node

Andres Quintero

Due Date:

Soft copy: 2/4/2020

Hard copy: 2/6/2020

Main():

Step 0: Open input file and output files from command line arguments

Step 1: Initialize listHead as dummy node will values (-9999, NULL)

Step 2: Call constructLL() method

Step 3: Call printList() method

Step 4: Initialize middleNode with return value from findMiddleNode() method

Step 5: Call printNode() method

Step 6: Close all files

**Source Code: main.cpp**

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

class listNode {

public:

int data;

listNode\* next;

listNode(int parameterData, listNode\* parameterNext){

data = parameterData;

next = parameterNext;

};

};

// Prototypes

void printList(listNode\* node, fstream& outFile);

listNode\* findSpot(listNode\* listHead, listNode\* newNode);

void listInsert(listNode\* listHead, listNode\* newNode);

void constructLL(listNode\* listHead, fstream& inFile, fstream& outFile2);

void printNode(listNode\* node, fstream& outFile);

listNode\* findMiddleNode(listNode\* listHead, fstream& outFile);

int main(int argc, char\* argv[]){

fstream inFile(argv[1]); // Input File

fstream outFile1(argv[2]); // Sorted LinkList AND Middle Node

fstream outFile2(argv[3]); // Debugging

listNode\* listHead = new listNode(-9999, NULL);

constructLL(listHead, inFile, outFile2);

printList(listHead, outFile1);

// SEGMENT FAULT?

listNode\* middleNode = findMiddleNode(listHead, outFile2);

printNode(middleNode, outFile1);

//Closing all files

inFile.close();

outFile1.close();

outFile2.close();

}

void constructLL(listNode\* listHead, fstream& inFile,fstream& outFile2){

int data;

while(inFile >> data){ // same as eof

listNode\* newNode = new listNode(data, NULL);

listInsert(listHead, newNode);

printList(listHead, outFile2);

}

}

void listInsert(listNode\* listHead, listNode\* newNode){

listNode\* Spot = findSpot(listHead, newNode);

newNode->next = Spot->next;

Spot->next = newNode;

}

listNode\* findSpot(listNode\* listHead, listNode\* newNode){

listNode\* Spot = listHead;

while(Spot->next != NULL && Spot->next->data < newNode->data){

Spot = Spot->next;

}

return Spot;

}

void printList(listNode\* node, fstream& outFile){

outFile << "listHead ->";

while(node != NULL){

if(node->next == NULL){ // because NULL->data will cause a segment falt 11

outFile << "(" << node->data << "," << node << "," << node->next << "," << "NULL" << ")";

} else {

outFile << "(" << node->data << "," << node << "," << node->next << "," << node->next->data << ") ->";

}

node = node->next;

}

outFile << "-> NULL" << endl;

}

listNode\* findMiddleNode(listNode\* listHead, fstream& outFile){

listNode\* walk1 = listHead;

listNode\* walk2 = listHead;

while(walk2->next->next != NULL){

printNode(walk1, outFile);

if(walk2 != NULL && walk2->next != NULL){

walk1 = walk1->next;

walk2 = walk2->next->next;

}

}

return walk2;

}

void printNode(listNode\* node, fstream& outFile){

if(node->next == NULL){

outFile << "(" << node->data << "," << node << "," << node->next << "," << "NULL" << ")";// because NULL->data will cause a segment falt 11

} else {

outFile << "(" << node->data << "," << node << "," << node->next << "," << node->next->data << ") ->";

}

**Input File: LLmiddleNode\_Data.txt**

91 322 9 10

77 8 999 12

133 14

8

538 29 91

88

702 361

637

99

**Output File 1: Linked List Print *and* Middle Node**

listHead ->(-9999,0x7fe764c016f0,0x7fe764d00080,8) -> (8,0x7fe764d00080,0x7fe764d00030,8) ->(8,0x7fe764d00030,0x7fe764d00000,9) -> (9,0x7fe764d00000,0x7fe764d00010,10) ->(10,0x7fe764d00010,0x7fe764d00050,12) -> (12,0x7fe764d00050,0x7fe764d00070,14) ->(14,0x7fe764d00070,0x7fe764d000a0,29) -> (29,0x7fe764d000a0,0x7fe764d00020,77) ->(77,0x7fe764d00020,0x7fe764d000c0,88) -> (88,0x7fe764d000c0,0x7fe764d000b0,91) ->(91,0x7fe764d000b0,0x7fe764c01700,91) -> (91,0x7fe764c01700,0x7fe764d00100,99) ->(99,0x7fe764d00100,0x7fe764d00060,133) -> (133,0x7fe764d00060,0x7fe764c01710,322) ->(322,0x7fe764c01710,0x7fe764d000e0,361) -> (361,0x7fe764d000e0,0x7fe764d00090,538) ->(538,0x7fe764d00090,0x7fe764d000f0,637) -> (637,0x7fe764d000f0,0x7fe764d000d0,702) ->(702,0x7fe764d000d0,0x7fe764d00040,999) -> (999,0x7fe764d00040,0x0,NULL)-> NULL

(702,0x7fe764d000d0,0x7fe764d00040,999) ->

**Output File 2: Debugging Print – *Not submitted***