**CS 700-34 LinkedList Insertion and Reverse the List**

**Student:** Shuhua Song

**Due Date: Submission Data:**

Soft Copy: 02/10/2020 Soft Copy: 02/10/2020

Hard Copy: 02/13/2020 Hard Copy: 02/13/2020

**Algorithm Steps:**

1. **Main()**

Step 1: Open input file ‘inFile’ by using argv[0]

Open output file ‘outFile1’ and ‘outFile2’ by using argv[1] and argv[2]

Step 2: Create a listHead as the dummy listNode, with value (“dummy”, null), make the listHead point to the dummy node;

Step 3: Construct a linkedlist in constructLL(inFile, outFile2)

Step 4: Print the complete ordered final list in printList(listHead, outFile1)

Step 5: Reverse the ordered linkedlist in LLreverse(listHead,outFile2)

Step 6: Print the reversed linked list entirely.

Step 7: Close all files;

1. Function **constructLL(inFile, outFile)** Steps

Step 1: Read the data from inFile

Step 2: Create a new node by using the new data, listNode(data,null)

Step 3: Insert the new data in the correct position of linkedlist in listInsert(listHead, newNode.

Step 4: Print the current list for debug purpose

Step 5: repeat step 1-step4 until the end of inFile

1. Function **listInsert(newNode**) steps

Step 1: Find the proper position ‘spot’ in the linkedlist for this new node to insert

Step 2: newNode’s next = spot.next, spot.next = newNode

1. Function **findspot(newNode)** Steps

Step 1: set the spot = listHead

Step 2: if spot’s next != null and spot’s next’s data < newNode’s data

spot = spot’next

Step 3: Repeat step2 until condition failed

Step 4: return spot

1. Function findMiddleNode(outFile)

Step 1: set walk1 = listHead, walk2 = listHead

Step 2: printNode(walk1, outFile1)

Step 3: if walk2 != null and walk2’next != null

walk1 = walk1’s next , walk2 = walk2.next.next

step 4:return walk1

1. reverseLL(listHead,outFile2)

Step 0: set node last = listHead

Step 1: set spot = last.next

Step 2: moveSpotNodeToFront(listHead, Spot)

Step3: print list for debugging by using printList(listHead, outFile2)

Step 4:repeat step 1 to step 3 until last’s next is null

1. moveSpotNodeToFront(listHead, Spot)

Step 0: set listHead.next = spot.next;

Step 1: spot.next = tail.next;

Step 2: tail.next = spot;

**Code:**

**import** java.io.FileNotFoundException;  
**import** java.io.FileReader;  
**import** java.util.Scanner;  
**import** java.io.\*;  
  
**public class** LinkedList {  
  
 **private static** listNode *listHead*;  
 **private static** listNode *tail*;  
 **static class** listNode {  
 String **data**;  
 listNode **next**;  
 listNode(){  
 **data** = **"dummy"**;  
 **next** = **null**;  
 }  
 listNode(String data, listNode next){  
 **this**.**data** = data;  
 **this**.**next** = next; ;  
 }  
  
 **void** printNode(listNode node, BufferedWriter fw) **throws** IOException {  
 fw.write(**"("** + node.**data** + **" ,"** + node + **" ,"** + node.**next** + **" ,"** + node.**next**.**data** + **")"**);  
 }  
 }  
   
 *//Constructor* LinkedList(){  
 listNode dummy = **new** listNode();  
 *listHead* = dummy;  
 *tail* = **new** listNode();  
 }  
 *//Construct a LinkedList* **public void** constructLL(Scanner reader, BufferedWriter writer) **throws** IOException {  
 **while**(reader.hasNext()){ *//while there is another token to read* String str = reader.next();*//reads in the String tokens and skip white-space character to start reading next token* listNode newNode = **new** listNode(str, **null**);*//can't use nextLine(), because the '\n' character is part of a valid line token. it will return empty string* listInsert(newNode);  
 printList(*listHead*,writer); *////  
 //System.out.println(str);* }  
 }  
  
 **public void** listInsert(listNode newNode) {  
 **if**(*listHead*.**next** == **null**) {  
 *listHead*.**next** = newNode;  
 }**else**{  
 listNode spot = findSpot(newNode);  
 newNode.**next** = spot.**next**;  
 spot.**next** = newNode;  
 }  
 }  
  
 **public** listNode findSpot(listNode newNode) {  
 listNode curNode = *listHead*;  
 **while**(curNode!= **null** && curNode.**next** != **null**){  
 **if**(curNode.**next**.**data**.compareTo(newNode.**data**) > 0) **break**;  
 curNode = curNode.**next**;  
 }  
 **return** curNode;  
 }  
  
 **public** listNode findMiddleNode(BufferedWriter fw) **throws** IOException {  
 listNode walk1 = *listHead*;  
 listNode walk2 = *listHead*;  
 listNode node = **new** listNode();  
 **while**(walk2 != **null** && walk2.**next** != **null**){  
 node.printNode(walk1, fw);  
 walk1 = walk1.**next**;  
 walk2 = walk2.**next**.**next**;  
 }  
 **return** walk1;  
 }  
  
 **public void** reverseLL(BufferedWriter fw) **throws** IOException {  
 listNode last = *listHead*;  
 listNode spot = *listHead*;  
 **while**(last.**next** != **null**){  
 spot = last.**next**;  
 moveSpotNodeToFront(spot);  
 printList(*tail*, fw);  
 }  
 }  
 **public void** moveSpotNodeToFront(listNode spot){  
 *listHead*.**next** = spot.**next**;  
 spot.**next** = *tail*.**next**;  
 *tail*.**next** = spot;  
 }  
  
   
 **public void** printList(listNode listHead, BufferedWriter fw) **throws** IOException {  
 listNode cur = listHead;  
 fw.write(**"listHead"**);  
 **while**(cur != **null** && cur.**next** != **null**){  
  
 fw.write(**"->("** + cur.**data** + **" ,"** + cur + **" ,"** + cur.**next** + **" ,"** + cur.**next**.**data** + **")\n"**);  
 cur = cur.**next**;  
 }  
 fw.write(**"->("** + cur.**data** + **" ,"** + cur + **")->"**);  
 fw.write(**"NULL"** + **"\n"**);  
 }  
  
  
 **public static void** main(String args[]) **throws** IOException {  
 *//The Scanner class is used to read file in tokens from an input stream* LinkedList list = **new** LinkedList();  
 Scanner inFile = **new** Scanner(**new** FileReader(args[0]));  
 BufferedWriter outFile1 = **new** BufferedWriter(**new** FileWriter(**new** File(args[1]))) ;  
 BufferedWriter outFile2 = **new** BufferedWriter(**new** FileWriter(**new** File(args[2]))) ;  
  
 *//print the completed linkedlist* list.constructLL(inFile, outFile2); */////* outFile1.write(**"Completed LinkedList: "**);  
 outFile1.write(**"\n"**);  
 list.printList(*listHead*, outFile1);  
  
 outFile1.write(**"Reverse LinkedList: "**);  
  
 list.reverseLL(outFile2);  
 list.printList(*tail*,outFile1);  
  
 inFile.close();  
 outFile1.close();  
 outFile2.close();  
 }  
}

**Output:**

Completed LinkedList:

listHead->(dummy ,LinkedList$listNode@816f27d ,LinkedList$listNode@5fd0d5ae ,Akshar)

->(Akshar ,LinkedList$listNode@5fd0d5ae ,LinkedList$listNode@16b98e56 ,Andres)

->(Andres ,LinkedList$listNode@16b98e56 ,LinkedList$listNode@214c265e ,Brandon)

->(Brandon ,LinkedList$listNode@214c265e ,LinkedList$listNode@723279cf ,Eleftherios)

->(Eleftherios ,LinkedList$listNode@723279cf ,LinkedList$listNode@6d311334 ,Esteban)

->(Esteban ,LinkedList$listNode@6d311334 ,LinkedList$listNode@2d98a335 ,Evgeniia)

->(Evgeniia ,LinkedList$listNode@2d98a335 ,LinkedList$listNode@87aac27 ,Hishaam)

->(Hishaam ,LinkedList$listNode@87aac27 ,LinkedList$listNode@6acbcfc0 ,Jeffrey)

->(Jeffrey ,LinkedList$listNode@6acbcfc0 ,LinkedList$listNode@4dd8dc3 ,Jiade)

->(Jiade ,LinkedList$listNode@4dd8dc3 ,LinkedList$listNode@7ba4f24f ,Jianwei)

->(Jianwei ,LinkedList$listNode@7ba4f24f ,LinkedList$listNode@5b480cf9 ,Jiaxin)

->(Jiaxin ,LinkedList$listNode@5b480cf9 ,LinkedList$listNode@58372a00 ,Jiayu)

->(Jiayu ,LinkedList$listNode@58372a00 ,LinkedList$listNode@448139f0 ,Joel)

->(Joel ,LinkedList$listNode@448139f0 ,LinkedList$listNode@27d6c5e0 ,Justin)

->(Justin ,LinkedList$listNode@27d6c5e0 ,LinkedList$listNode@15aeb7ab ,Kelvin)

->(Kelvin ,LinkedList$listNode@15aeb7ab ,LinkedList$listNode@682a0b20 ,Kevin)

->(Kevin ,LinkedList$listNode@682a0b20 ,LinkedList$listNode@7cca494b ,Luis)

->(Luis ,LinkedList$listNode@7cca494b ,LinkedList$listNode@7ef20235 ,Marco)

->(Marco ,LinkedList$listNode@7ef20235 ,LinkedList$listNode@3d075dc0 ,Matthew)

->(Matthew ,LinkedList$listNode@3d075dc0 ,LinkedList$listNode@378bf509 ,Mohebullah)

->(Mohebullah ,LinkedList$listNode@378bf509 ,LinkedList$listNode@6d03e736 ,Phillip)

->(Phillip ,LinkedList$listNode@6d03e736 ,LinkedList$listNode@4f3f5b24 ,Robin)

->(Robin ,LinkedList$listNode@4f3f5b24 ,LinkedList$listNode@568db2f2 ,Russell)

->(Russell ,LinkedList$listNode@568db2f2 ,LinkedList$listNode@7699a589 ,Taeyong)

->(Taeyong ,LinkedList$listNode@7699a589 ,LinkedList$listNode@3b9a45b3 ,Yechiel)

->(Yechiel ,LinkedList$listNode@3b9a45b3 ,LinkedList$listNode@5f184fc6 ,Yifei)

->(Yifei ,LinkedList$listNode@5f184fc6 ,LinkedList$listNode@3feba861 ,Yinyu)

->(Yinyu ,LinkedList$listNode@3feba861 ,LinkedList$listNode@6f496d9f ,Youyia)

->(Youyia ,LinkedList$listNode@6f496d9f ,LinkedList$listNode@10f87f48 ,Yuan)

->(Yuan ,LinkedList$listNode@10f87f48 ,LinkedList$listNode@7b23ec81 ,Zhiheng)

->(Zhiheng ,LinkedList$listNode@7b23ec81)->NULL

Reverse LinkedList: listHead->(dummy ,LinkedList$listNode@b4c966a ,LinkedList$listNode@7b23ec81 ,Zhiheng)

->(Zhiheng ,LinkedList$listNode@7b23ec81 ,LinkedList$listNode@10f87f48 ,Yuan)

->(Yuan ,LinkedList$listNode@10f87f48 ,LinkedList$listNode@6f496d9f ,Youyia)

->(Youyia ,LinkedList$listNode@6f496d9f ,LinkedList$listNode@3feba861 ,Yinyu)

->(Yinyu ,LinkedList$listNode@3feba861 ,LinkedList$listNode@5f184fc6 ,Yifei)

->(Yifei ,LinkedList$listNode@5f184fc6 ,LinkedList$listNode@3b9a45b3 ,Yechiel)

->(Yechiel ,LinkedList$listNode@3b9a45b3 ,LinkedList$listNode@7699a589 ,Taeyong)

->(Taeyong ,LinkedList$listNode@7699a589 ,LinkedList$listNode@568db2f2 ,Russell)

->(Russell ,LinkedList$listNode@568db2f2 ,LinkedList$listNode@4f3f5b24 ,Robin)

->(Robin ,LinkedList$listNode@4f3f5b24 ,LinkedList$listNode@6d03e736 ,Phillip)

->(Phillip ,LinkedList$listNode@6d03e736 ,LinkedList$listNode@378bf509 ,Mohebullah)

->(Mohebullah ,LinkedList$listNode@378bf509 ,LinkedList$listNode@3d075dc0 ,Matthew)

->(Matthew ,LinkedList$listNode@3d075dc0 ,LinkedList$listNode@7ef20235 ,Marco)

->(Marco ,LinkedList$listNode@7ef20235 ,LinkedList$listNode@7cca494b ,Luis)

->(Luis ,LinkedList$listNode@7cca494b ,LinkedList$listNode@682a0b20 ,Kevin)

->(Kevin ,LinkedList$listNode@682a0b20 ,LinkedList$listNode@15aeb7ab ,Kelvin)

->(Kelvin ,LinkedList$listNode@15aeb7ab ,LinkedList$listNode@27d6c5e0 ,Justin)

->(Justin ,LinkedList$listNode@27d6c5e0 ,LinkedList$listNode@448139f0 ,Joel)

->(Joel ,LinkedList$listNode@448139f0 ,LinkedList$listNode@58372a00 ,Jiayu)

->(Jiayu ,LinkedList$listNode@58372a00 ,LinkedList$listNode@5b480cf9 ,Jiaxin)

->(Jiaxin ,LinkedList$listNode@5b480cf9 ,LinkedList$listNode@7ba4f24f ,Jianwei)

->(Jianwei ,LinkedList$listNode@7ba4f24f ,LinkedList$listNode@4dd8dc3 ,Jiade)

->(Jiade ,LinkedList$listNode@4dd8dc3 ,LinkedList$listNode@6acbcfc0 ,Jeffrey)

->(Jeffrey ,LinkedList$listNode@6acbcfc0 ,LinkedList$listNode@87aac27 ,Hishaam)

->(Hishaam ,LinkedList$listNode@87aac27 ,LinkedList$listNode@2d98a335 ,Evgeniia)

->(Evgeniia ,LinkedList$listNode@2d98a335 ,LinkedList$listNode@6d311334 ,Esteban)

->(Esteban ,LinkedList$listNode@6d311334 ,LinkedList$listNode@723279cf ,Eleftherios)

->(Eleftherios ,LinkedList$listNode@723279cf ,LinkedList$listNode@214c265e ,Brandon)

->(Brandon ,LinkedList$listNode@214c265e ,LinkedList$listNode@16b98e56 ,Andres)

->(Andres ,LinkedList$listNode@16b98e56 ,LinkedList$listNode@5fd0d5ae ,Akshar)

->(Akshar ,LinkedList$listNode@5fd0d5ae)->NULL