Laboratory work #5

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1067

Screenshot from Timus:



Explanation of algorithm:

Create a tree. Create a class “Directory” to store the nodes for each directory and create a LinkedList to store subdirectories under this node. Traverse and save each node in turn.

Computational complexity of algorithm:

T(N) = O(N)

Source code:

import java.util.LinkedList;

import java.util.ListIterator;

import java.util.Scanner;

public class App {

public static class Directory {

public String name;

public LinkedList<Directory> child;

public Directory(String name) {

this.name = name;

}

@Override

public String toString() {

return name;

}

}

public static Directory putSorted(LinkedList<Directory> tree, String name) {

ListIterator<Directory> it = tree.listIterator();

Directory newDir = new Directory(name);

boolean inserted = false;

while (it.hasNext()) {

Directory dir = it.next();

int compare = dir.name.compareTo(name);

if (compare == 0) {

inserted = true;

newDir = dir;

break;

}

else if (compare > 0) {

inserted = true;

it.previous();

it.add(newDir);

break;

}

}

if (!inserted) {

tree.add(newDir);

}

return newDir;

}

public static void put(LinkedList<Directory> tree, String directory) {

int slashId = directory.indexOf('\\');

if (slashId == -1) {

putSorted(tree, directory);

}

else {

String currentDirectory = directory.substring(0, slashId);

String nextDirectory = directory.substring(slashId + 1);

Directory dir = putSorted(tree, currentDirectory);

if (dir.child == null)

dir.child = new LinkedList<Directory>();

put(dir.child, nextDirectory);

}

}

public static void printDirectory(LinkedList<Directory> tree, int level) {

if (tree != null) {

for (Directory dir : tree) {

for (int j = 0; j < level; j++) {

System.out.append(' ');

}

System.out.println(dir.name);

printDirectory(dir.child, level + 1);

}

}

}

public static void main(String[] args) {

LinkedList<Directory> tree = new LinkedList<Directory>();

Scanner scan = new Scanner(System.in);

int N = scan.nextInt();

scan.nextLine();

for (int i = 0; i < N; i++) {

put(tree, scan.nextLine());

}

printDirectory(tree, 0);

scan.close();

}

}