# **Format for uploading details in GitHub and Slack in word file format**

**Student Name: Sindhu N**

**Class and Sec: VI B**

**USN:** **4AL17CS094**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Online Test Details** | | | | |
| **Subject** |  | | | |
| **Semester** | **VI – B** | | **Duration** | **Minutes** |
|  | |  | | |

**Encl: snapshot of the test result**

|  |  |  |  |
| --- | --- | --- | --- |
| **Pre-Placement Training Summary** | | | |
| **Pre placement training** | **----** | | |
| **Faculty** | **----** | **Duration** | **-------** |

**Encl: snapshots of the daily class activities**

|  |  |
| --- | --- |
| **Coding Challenges** | |
| **Problem Statement: Prog1(java)** | |
| **Status: Completed** | |
| **Uploaded the report both in GitHub & Slack** | Yes |

**Encl: snapshots of your response to challenge.**

Write a program to find two lines with max characters in descending order.

package com.longest.lines;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.util.Comparator;

import java.util.Set;

import java.util.TreeSet;

public class Main {

public static void main(String[] args) {

BufferedReader br = null;

String filePath = args[0];

int topList = 0;

Set<Entries> liSet = new TreeSet<Entries>(new MyComp());

try {

br = new BufferedReader(new FileReader(new File(filePath)));

String line = br.readLine();

topList = Integer.parseInt(line.trim());

while((line = br.readLine()) != null){

line = line.trim();

if(!"".equals(line)){

liSet.add(new Entries(line.length(), line));

}

}

int count = 0;

for(Entries ent:liSet){

System.out.println(ent.line);

if(++count == topList){

break;

}

}

} catch (FileNotFoundException e) {

// TODO Auto-generated catch block

e.printStackTrace();

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

public static class Entries{

Integer length;

String line;

public Entries(Integer l,String line){

length = l;

this.line = line;

}

}

public static class MyComp implements Comparator<Entries>{

@Override

public int compare(Entries e1, Entries e2) {

if(e2.length > e1.length){

return 1;

} else {

return -1;

}

}

}

}

