IE 343 2022 Fall Project

Due to time restrictions and busy schedules, we couldn’t demonstrate a higher effort on the project. The code aims to provide the maximum value to the album maker. Achieving the maximum value of the album with making it no longer than 30 minutes and penalizing the value with the associated penalty cost is basically the objective of the project. You can also fine some comments attached just next to the code. The code gets and reads the necessary data firstly. It also splits down the value matrix of the songs into arrays and allows us to search those arrays one by one. Then returns the value in the end.

**package** Work;

**import** java.io.BufferedReader;

**import** java.io.FileReader;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**import** Work.HeuristicSolve;

**import** Work.Track;

**public** **class** Main {

**public** **static** **void** main(String[] args) **throws** IOException{

List<List<String>> list = *readValues*();

System.***out***.println(list.get(0));

List<Integer> valueList = **new** ArrayList<Integer>();

List<Integer> weightList = **new** ArrayList<Integer>();

**for**(**int** i=1;i<list.size();i++) {

valueList.add(Integer.*parseInt*(list.get(i).get(4)));

weightList.add(Integer.*parseInt*(list.get(i).get(5)));

}

List<List<String>> list1 = *readSequential*();

List<ArrayList<Double>> sequential\_data = **new** ArrayList<ArrayList<Double>>();

ArrayList<Track> AllTracks = **new** ArrayList<>();

**double**[] TrackValues = **new** **double**[51];

**double** TrackPairs [][] = **new** **double**[50][50];

**for** (**int** i=1;i<list1.size();i++){

ArrayList<Double> row = **new** ArrayList<>();

**for** (**int** j=1;j<list1.get(0).size();j++){

row.add(Double.*parseDouble*(list1.get(i).get(j)));

}

sequential\_data.add(row);

}

System.***out***.println(sequential\_data.get(0).get(1));

//////////////////////////////////////////////////////////////////////////////

**for** (**int** i =1 ; i < list1.size();i++){

**for** (**int** j =1 ; j < list1.get(0).size();j++ ){

System.***out***.println(list1.get(i).get(j));

TrackPairs[i-1][j-1]=Double.*parseDouble*(list1.get(i).get(j));

}

}

**for** (**int** i =1 ; i < list1.size();i++){

**for** (**int** j =1 ; j < list1.get(0).size();j++ ){

**if** (AllTracks.size()<50){

Track track = **new** Track(Integer.*parseInt*(list.get(j).get(0)),Integer.*parseInt*(list.get(j).get(5)),Integer.*parseInt*(list.get(j).get(4)),TrackPairs[j-1]);

AllTracks.add(track);

}**else** {

**break**;

}

}

}

*Solve*(AllTracks);

**int** x;

x = 3;

}

**public** **static** List<List<String>> readValues() **throws** IOException {

**try**

{

List< List<String> > data = **new** ArrayList<>();//list of lists to store data

String file = "Work/term\_project\_value\_data.csv";//file path

FileReader fr = **new** FileReader(file);

BufferedReader br = **new** BufferedReader(fr);

//Reading until we run out of lines

String line = br.readLine();

**while**(line != **null**)

{

List<String> lineData = Arrays.*asList*(line.split(","));//splitting lines

data.add(lineData);

line = br.readLine();

}

//printing the fetched data

**for**(List<String> list : data)

{

**for**(String str : list)

System.***out***.print(str + " ");

System.***out***.println();

}

br.close();

**return** data;

}

**catch**(Exception e)

{

System.***out***.print(e);

List< List<String> > data = **new** ArrayList<>();//list of lists to store data

**return** data;

}

}

**public** **static** List<List<String>> readSequential() **throws** IOException {

**try**

{

List< List<String> > data = **new** ArrayList<>();//list of lists to store data

String file = "Work/term\_project\_sequential\_data.csv";//file path

FileReader fr = **new** FileReader(file);

BufferedReader br = **new** BufferedReader(fr);

//Reading until we run out of lines

String line = br.readLine();

**while**(line != **null**)

{

List<String> lineData = Arrays.*asList*(line.split(","));//splitting lines

data.add(lineData);

line = br.readLine();

}

//printing the fetched data

**for**(List<String> list : data)

{

**for**(String str : list)

System.***out***.print(str + " ");

System.***out***.println();

}

br.close();

**return** data;

}

**catch**(Exception e)

{

System.***out***.print(e);

List< List<String> > data = **new** ArrayList<>();//list of lists to store data

**return** data;

}

}

**public** **static** **double** Solve(ArrayList<Track> AllTracks){

**int** RemainingDurationTime = 1800000;

**double** TotalAlbumValue= 0;

ArrayList<Track> AddedTracks = **new** ArrayList<>();

Track currentBest =AllTracks.get(0);

**int** currentBestTime=AllTracks.get(0).track\_duration;

**while**(RemainingDurationTime > 0 && RemainingDurationTime-currentBestTime > 0){

**for**(**int** i=0;i<AllTracks.size();i++){

**if** (AllTracks.get(i).track\_individual\_value>currentBest.track\_individual\_value ){

currentBest = AllTracks.get(i);

}

}

**if** (RemainingDurationTime-currentBestTime>0){

RemainingDurationTime=RemainingDurationTime-currentBestTime;

TotalAlbumValue =TotalAlbumValue+currentBest.track\_individual\_value;

AllTracks.remove(currentBest);

AddedTracks.add(currentBest);

currentBest=AllTracks.get(0);

}

**int** l =0;

}

TotalAlbumValue=TotalAlbumValue-RemainingDurationTime\*0.02;

**return** TotalAlbumValue;

}

}

Arda Fırat

Beyza Kuruoğlu