**IE-343 PROJECT REPORT**

To complete the heuristic method of the problem I chose to use Dijkstra Algorithm which code’s is written in the recitation hour. To maximize the number of the tracks which will be put in the album. Since it is the heuristic method optimality of the solution is not guaranteed. Dijkstra Algorithm is an algorithm for finding the shortest paths between nodes in a graph. Therefore it can be modified for this algorithm to include the shortest durations in the album to maximize the number of the tracks that will be in the album.

Dijkstra Algorithm Step by Step Explanation

1. Open a java class which is called “HeuristicAlgorithm”.

2. Open another java class which is called “TrackData”.

3. To include the track’s property: Add to TrackData class “public Track[] track;”

4. To include the maximum duration time in miliseconds: Add to TrackData class “public int maxDurationTime = 30\* 10^3;”. It is stated that the album should last 30 minutes in miliseconds it should be 30 \* 10^3.

5.To include the total number of the tracks: Add to TrackData class “public int totalNumberOfTracks = 50“. It is stated in the question there is 50 songs which is recorded by the Ibrahim Maalouf and his band.

6. In the class of HeuristicMethod track sequential value should be shown as two dimensional matrix therefore it should be illustrating as “private int[] [] track\_sequential\_value“.

7. In the class of HeuristicMethod, there are number of tracks which should be shown as an arraylist : “private Arraylist<Integer> track; “.

8. In the class of HeuristicMethod, there is a one dimensional matrix which includes the track’s individual values: “private int[] track\_individual\_value;”

9. In the class of HeuristicMethod, to illustrate the previous values a one dimensional matrix should be constructed which is shown as “private int[] previous;”.

10. To show how and which type of data that will be used this should be constructed in the HeuristicAlgorithm class.

“public HeuristicMethod(int[][] track\_sequential\_value) {

this.track\_sequential\_value = track\_sequential\_value; // Two dimensional matrix

this.track = new ArrayList<Integer> (); // An Integer ArrayList

/\* To get the integer values of the track’s individual values from the file as one dimensional matrix\*/

this.track\_individual\_value = new int[this.track\_sequential\_value[0].length];

this.previous = new int[this.track\_sequential\_value[0].length];

// For every track’s individual value to determine the maximum value

for(int i = 0; i<track\_individual\_value[0].length;i++) {

track\_individual\_value[i] = Integer.MIN\_VALUE;

}

} “

11. In the class of HeuristicMethod, to illustrate the data of the track: “private TrackData trackData;”.

12. To overload the HeuristicMethod : Add as a first row “HeuristicMethod(TrackData trackdata) { “, as a second row “this.trackData = trackData;”.

13. Add to main “TrackData trackData = (TrackData) readSequential();”.

14. To call the heuristic algorithm from main add to main: “ HeuristicMethod heuristicMethod = new HeuristicMethod(trackData);” and “HeuristicMethod.algorithm();”.

15. To call the algorithm that shows the which tracks in the album from the main add to main: “heuristicMethod.printTracks(0, trackData.totalNumberOfTracks);”.

16. Add a method to the HeuristicAlgorithm which are called “algorithm” and “printTracks”.

17. In the method is called “algorithm” to find the maximum value to put in the first index and add a method to HeuristicAlgorithm “findMaxValue();”.

18. And put the first index to the first maximum value.

19. In the method is called “algorithm” to find the second maximum value, add a method to HeuristicAlgorithm “findSecondMaxValue();”.

20. And put the last index to the second maximum value.

21. In the method is called “findMaxValue”, the general approach of the method : To find the maximum value based on the idea if the value of the track is greater than the maximum previous value then the new maximum value now is that value. Check this method for every track in the track list.

Thus construct an algorithm:

“private int findMaxValue() {

int maxValue = -1;

int maxPrevious = Integer.MIN\_VALUE;

for( int previous : this track) { // To check the every track in the track arraylist

// To compare the individual values of the tracks

if(this.track\_individual\_value[previous] >= maxPrevious) {

maxValue = previous;

maxPrevious = this.track\_individual\_value[previous];

}

return maxValue; // Get the maximum value “

22. In the algorithm which is called “findSecondMaxValue” the general approach of the algorithm is after choosing maximum value by the comparison the track’s individual values the value which is smaller than the maximum value is the second maximum value.

“private int findSecondMaxValue() {

int secondMaxValue;

// To check the all tracks in the tracklist

for(int i = 0; i<this.trackData.totalNumberOfTracks;i++) {

for(int j = i+1; j< this.trackData.totalNumberOfTracks;j++) {

secondMaxValue =this.track\_individual\_value[i];

this.track\_individual\_value[i] = this.track\_individual\_value[j];

this.track\_individual\_value[j] = secondMaxValue;

}

}

return this.track\_individual\_value[this.trackData.totalNumberOfTracks-2]; “

23. The method which is called “solve” for stands for to sort the tracks as their descending value except the last index. The algorithm starts with checking the sequential values of track’s to make an order descending and removing the that value if there is another value which is greater than the maximum value. Also checking that there is enough space to put the values and already contains a value. This method should also be added to the main as “heuristicMethod.solve(0,trackData.totalNumberOfTracks );”

To show the algorithm which is in the HeuristicAlgorithm class:

“ // To add the values to an arraylist which has maximum values by comparing them.

public int solve(int first, int last){

for (int i=0;i>this.track\_sequential\_value[0].length;i++){

this.track.add(i);

}

this.track\_individual\_value[first] = 0;

/\* If this track is already on the list remove it and determine the values of them by comparing. \*/

while (!this.track.isEmpty()){

int u = findMaxValue();

this.track.remove(Integer.valueOf(u));

if (this.track\_individual\_value[u] == Integer.MIN\_VALUE){

break;

}

// To check that the every track’s seguential value in the album and make an order.

for (int v = 0; v<this.track\_sequential\_value[0].length; v++){

if (this.track.contains(Integer.valueOf(v))){

if (this.track\_sequential\_value[u][v] >= 0){

continue;

}

else{

int alt = this.track\_individual\_value[u] + this.track\_sequential\_value[u][v];

/\* To make an order by checking individual values of the tracks’

if (alt > this.track\_individual\_value[v]){

this.track\_individual\_value[v] = alt;

this.previous[v] = u;

}

}

}

}

}

return this.track\_individual\_value[last];

} “

24. To get the the list of the tracks which is in the album add the method to HeuristicAlgoritm class :

“

public void printTracks(int first, int last){

if (this.track\_individual\_value[last] == Integer.MIN\_VALUE){

System.out.println("No path exists!");

}

else{ // If this is the first index put it as previous index

ArrayList<Integer> path = new ArrayList<Integer>();

int u = last;

while (u != first){

path.add(u);

u = this.previous[u];

}

path.add(this.previous[u]);

for (int i = path.size()-1;i>0;i--){ //Printing the path by descending index

System.out.print(path.get(i)+" -->");

}

System.out.println(path.get(0));

}

} “