NAME : WIJAYAWARDHANA W.A.H.A.

REGISTRATION NO. : 2019/E/166

SEMESTER : SEMESTER 04

DATE ASSIGNED : 30 MARCH 2022

GRAPH – LAB 07

EC 4070

DATA STRUCTURES AND ALGORITHMS

01.

**Code:**

import java.util.ArrayList;

import java.util.Scanner;

public class JungleRun\_2019\_E\_166\_L7 {

int mapSize;

Scanner scanner = new Scanner(System.in);

ArrayList<String> pathElementList = new ArrayList<>();

String[][] mapElements = setMapElements(mapSize);

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

int lengthIndex= 0;

int minimumLength = 100;

/\*\*

\* For set mapSize variable and calling setMapElements.

\*/

public void setMapSize()

{

System.out.println("Enter the map size : ");

mapSize = scanner.nextInt();

mapElements = setMapElements(mapSize);

findPath(0,0," ",1);

}

/\*\*

\* setMapElements use for set elements into 2D array.

\* @param mapSize

\* @return

\*/

public String[][] setMapElements(int mapSize)

{

String[][] mapElementsN = new String[mapSize][mapSize];

for(int i =0; i < mapSize; i++)

{

for(int j =0; j<mapSize; j++)

{

mapElementsN[i][j] = scanner.next();

}

}

return mapElementsN;

}

/\*\*

\* findPath method use to find the shorted path of the jungle.

\*/

public void findPath(int rowIndex , int columnIndex , String tempPathArray ,int lengthPath)

{

if((rowIndex-1>=0)&&(columnIndex<mapSize)&&("E".equals(mapElements[rowIndex-1][columnIndex])))

{

tempPathArray = tempPathArray+ " " + "E";

pathElementList.add(tempPathArray);

if(minimumLength > pathElementList.size())

{

System.out.println("Path added.");

minimumLength = lengthPath;

}

length.add(lengthIndex , pathElementList.size());

lengthIndex++;

}

else if((rowIndex<mapSize)&&(columnIndex+1<mapSize)&&("E".equals(mapElements[rowIndex][columnIndex+1])))

{

tempPathArray = tempPathArray+ " " + "E";

if(minimumLength > pathElementList.size())

{

System.out.println("Path added.");

minimumLength = lengthPath;

}

length.add(lengthIndex , pathElementList.size());

lengthIndex++;

}

else if((rowIndex<mapSize)&&(columnIndex-1>=0)&&("E".equals(mapElements[rowIndex][columnIndex-1])))

{

tempPathArray = tempPathArray+ " " + "E";

if(minimumLength > pathElementList.size())

{

System.out.println("Path added.");

minimumLength = lengthPath;

}

length.add(lengthIndex , pathElementList.size());

lengthIndex++;

}

else if((rowIndex+1<mapSize)&&(columnIndex<mapSize)&&("E".equals(mapElements[rowIndex+1][columnIndex])))

{

tempPathArray = tempPathArray+ " " + "E";

if(minimumLength > pathElementList.size())

{

minimumLength = lengthPath;

}

length.add(lengthIndex , pathElementList.size());

lengthIndex++;

}

else if((rowIndex+1<mapSize)&&(columnIndex<mapSize)&&("P".equals(mapElements[rowIndex+1][columnIndex])))

{

tempPathArray = tempPathArray+ " " + "P";

lengthPath++;

findPath(rowIndex+1 , columnIndex,tempPathArray ,lengthPath);

}

else if((rowIndex<mapSize)&&(columnIndex+1<mapSize)&&("P".equals(mapElements[rowIndex][columnIndex+1])))

{

tempPathArray = tempPathArray+ " " + "P";

lengthPath++;

findPath(rowIndex , columnIndex+1,tempPathArray,lengthPath);

}

else if((rowIndex-1>=0)&&(columnIndex<mapSize)&&("P".equals(mapElements[rowIndex-1][columnIndex])))

{

tempPathArray = tempPathArray+ " " + "P";

lengthPath++;

findPath(rowIndex-1 , columnIndex,tempPathArray,lengthPath);

}

else if((rowIndex<mapSize)&&(columnIndex-1>=0)&&("P".equals(mapElements[rowIndex][columnIndex-1])))

{

tempPathArray = tempPathArray+ " " + "P";

lengthPath++;

findPath(rowIndex , columnIndex-1 ,tempPathArray,lengthPath);

}

else

{

return;

}

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

JungleRun\_2019\_E\_166\_L7 newObject = new JungleRun\_2019\_E\_166\_L7();

newObject.setMapSize();

System.out.println(newObject.minimumLength);

}

}

**Output:**

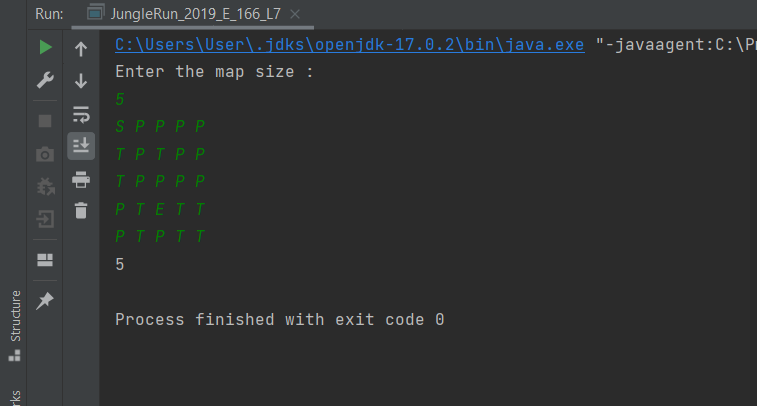


FIGURE 01 – OUTPUT

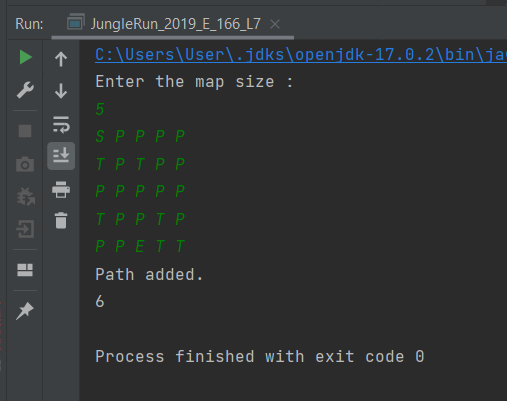


FIGURE 02 – OUTPUT