Graph Implementation Program REPORT

Course: Data Structures / CSE 2025.1

Lecturer: Murat Can Ganiz

Semester: 2021 Spring

Date: 12/06/2021

Group Members:

Ahmet Faruk Güzel – 150119659

Hüseyin Kerem Mican – 150119629

Muhammet Eren Atala – 150119904

Titles:

Introduction

In this project, Graph Implementation Program with Dijkstra's Algorithm has been created successfully. The program provides a simulation of social network by using a graph.

Dijkstra's Algorithm is required by the Graph Implementation Program. When the software is launched, the user will be presented with a menu interface with all of the necessary settings. Read file, Show adjacency matrix/list, Find shortest path and Exit are the options of user interface. User may read the input file, show the adjacency matrix or list, find the shortest path of the diagram or exit the program by typing the relevant number in the menu. In the option 1, user enters the input file's name which is asked. After entering the name of the file, program reads it. By choosing the second option, adjacency matrix/list will be printed on the screen. Whether it prints matrix or list in the output depends on how the function was written. To find the shortest path of the diagram, user needs to choose 3rd option. This function will ask the source vertex and destination vertex to the user. After entering source and destination vertices the program calculates the shortest path between these nodes by using Dijkstra's Algorithm. It also includes those vertices on the path and prints the length of it. To exit the program user may choose the 4th option.

Overall, a complete Dijkstra's Algorithm was used to build a proper example of a Graph Implementation Program. The basics and logic of the program are presented in this title. The code and primary structure of the program will be described in further detail in the following subsections.

Detailed Explanation of Program

The functions will be explained and it will be shown by figures in following explanations. All functions work successfully and they do not come up with any errors.

createGraph: This function, takes the Datas Struct as argument. The graph struct has been created with malloc and it creates a vertex array by taking the number of vertices as array size and it makes them null. After this process, function takes source node and destination node to create adjacency list.

traverse: This function is a struct Node function which prints destination and weight of vertexes.

printGraph: This is the second option's void function which prints graph as adjacency list in this program. There is a loop which uses traverse function to print vertexes in reverse.

ReadFile: This void type function takes Datas and Graph struct as argument and uses them as a storage. With necessary array, integer and char variables, in for and while loops function stores sources, destinations, weights, vertexes, and number of vertexes. At the end of this function, it creates the graph to use for the rest of the program.

UI: This function is user interface and takes Data struct as argument to distribute informations to other functions above. There are 4 cases which are read file, print list, find the shortest path and exit.

main: In main function there is a variable named data, type struct, created and being sent to UI function.

```
Please select an operation:
 1. Read file
 2. Show adjacency list
 3. Find shortest path
 4. Exit
1
```

When the user runs the program, an interface will be encountered. There will be 4 options to be choose. By typing number of Read File option, the program asks the input file's name as input from user. And it reads the file after taking the name of the file from user.

```
Please select an operation:
 1. Read file
 2. Show adjacency list
 3. Find shortest path
 4. Exit
2
print list
A: B,2 D,7 F,12 G,2
B: A,2 C,1 D,4 E,3 G,5
C: B,1 E,4 G,4
D: A,7 B,4 E,1 H,5
E: B,3 C,4 D,1 H,7
F: A,12 H,3
G: A,2 B,5 C,4
H: D,5 E,7 F,3
```

To print adjacency list, user needs to call second option. It will show the adjacency list from taken input file.

Conclusion

In this detailed report, the main aspects of a successful example of the Graph Implementation Program are outlined under three topics. In the introductory section, the program's definition was given. The second title describes how the program and its functions operate. The work done in functions has been displayed as subheads and figures in this title. Every subhead contains the program's structure and logic. Figures have been provided as screenshots and descriptions have been added to make the explanation clearer. All figures include every possible action that may be performed in a program. Dijkstra's Algorithm-based programs have been evaluated and debugged using various ways. Read file, Show adjacency matrix/list options are working properly, Find shortest path function is not included in the project.