CSE225-CSE2025 Data Structures PROJECT #2
Graph Implementation - Dijkstra's Algorithm
Report

Beyzanur Çabuk – 150119632
Erdi Türkay – 150119853

Functions

void initializeMinHeap(struct MinHeap *minHeap, int capacity): This function takes a MinHeap struct and initialize it with the given *capacity* by setting members.

void swapMinHeapNodes(struct Node *x, struct Node *y): This function is used when inserting to or extracting from the min heap. It basically swaps two min heap nodes.

int parentOfHeapNode(int i): This function calculates and returns the index of the parent of the node whose index is given as *i*.

int leftChildOfHeapNode(int i): This function calculates and returns the index of the left child of the node whose index is given as *i*.

int rightChildOfHeapNode(int i): This function calculates and returns the index of the right child of the node whose index is given as *i*.

struct Node getMinElementFromMinHeap(struct MinHeap *minHeap): This function returns the root node, i.e. the minimum element, of the given *minHeap*. This operation does not modify the heap, the root node is not extracted.

void insertElementToMinHeap(struct MinHeap *minHeap, struct Node element): This function inserts the given node *element* into the given *minHeap*. This function handles all needed operations to preserve min heap property.

void heapify(struct MinHeap *minHeap, int i): This is a recursive function. By doing necessary swaps, it transforms the subheap whose root’s index is given as i.

void extractMinElementFromMinHeap(struct MinHeap *minHeap): This function extracts the root node, i.e. the minimum element, from the given *minHeap*. To do this, it moves the last element to the root, decreases the size and calls heapify function.

void readInputFile(): This function is called when the user chooses ***1 - Read File*** option from the menu. It reads the file whose name is entered by the user and populates the adjaceny matrix for future usage.

void showAdjacencyMatrix(): This function is called when the user chooses **2 - Show Adjacency Matrix** option from the menu. It prints the adjacency matrix in aligned form.

int showMenuAndGetChoice(): This function shows the menu and gets the choice of the user until he or she enters a valid choice. It returns the user choice.

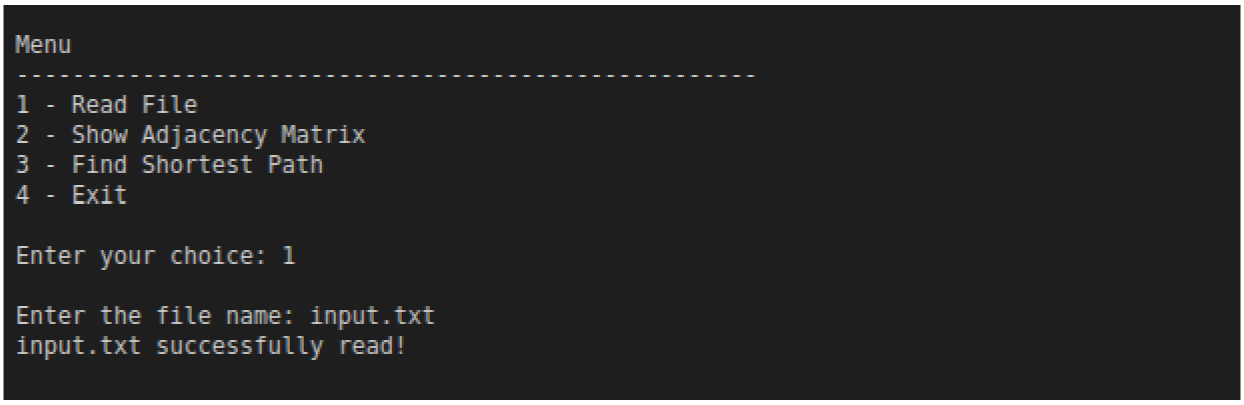
void printPath(int j): This is also a recursive function. It prints the shortest path from source to destination whose index given as *j*. This function uses the *parentArr* that contains the prodecessor of each city.

void shortestPath(): This function is called when the user chooses 3 – **Find Shortest Path** option from the menu. It firstly gets the source and destination cities from the user. Then it creates and initializes a min heap. Then it applies Dijkstra's algorithm using that min heap to find shortest pathes from the source city. Finally, it checks whether there is a path from source to destination or not. If there is, calling *printPath* function, shortest path from source to destination is printed. It also prints the total length of this shortest path.

void main(): This is the main function of the program. Until user exits, it gets the choice from the user by calling *showMenuAndGetChoice* and calls the appropriate function based on the user's choice.

Screenshots

1) Read File

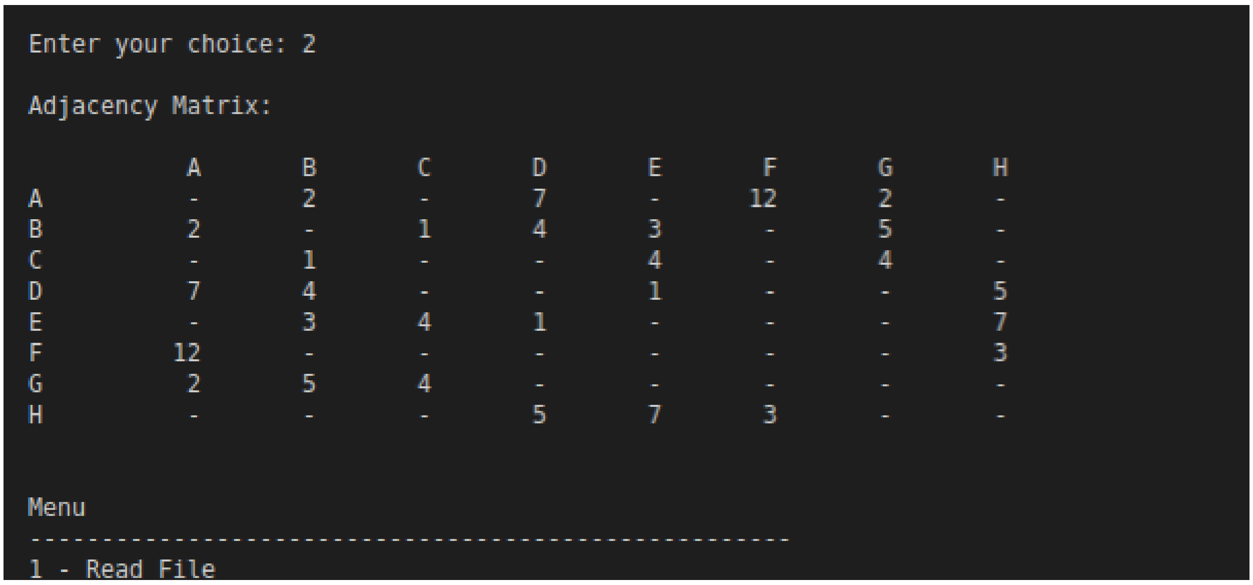


```
Menu
----------------------------------------------------
1 - Read File
2 - Show Adjacency Matrix
3 - Find Shortest Path
4 - Exit

Enter your choice: 1

Enter the file name: input.txt
input.txt successfully read!
```

2) Show Adjaceny Matrix



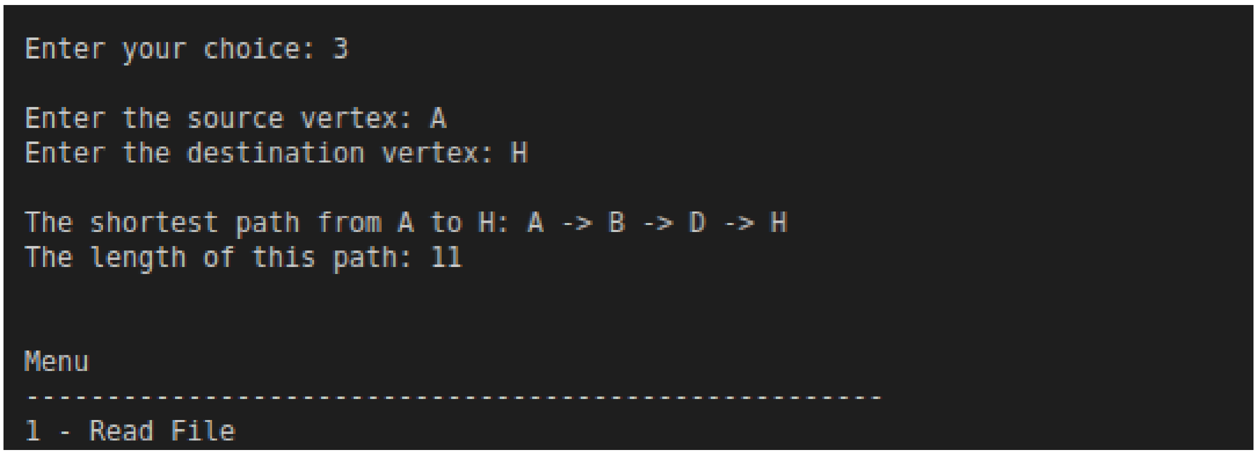
```
Enter your choice: 2

Adjacency Matrix:

           A       B       C       D       E       F       G       H
A          -       2       -       7       -      12       2       -
B          2       -       1       4       3       -       5       -
C          -       1       -       -       4       -       4       -
D          7       4       -       -       1       -       -       5
E          -       3       4       1       -       -       -       7
F         12       -       -       -       -       -       -       3
G          2       5       4       -       -       -       -       -
H          -       -       -       5       7       3       -       -


Menu
----------------------------------------------------
1 - Read File
```

3) Shortest Path



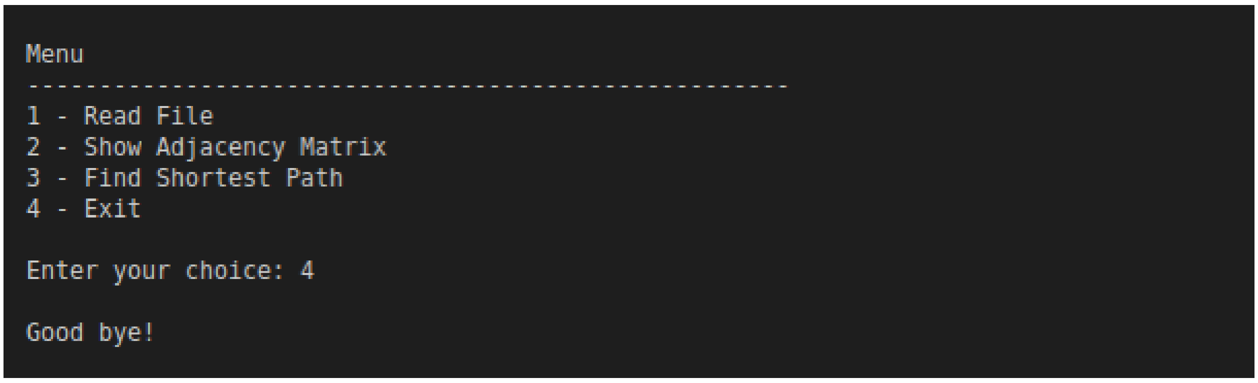
```
Enter your choice: 3

Enter the source vertex: A
Enter the destination vertex: H

The shortest path from A to H: A -> B -> D -> H
The length of this path: 11


Menu
-----------------------------------------------------
1 - Read File
```

4) Exit



```
Menu
-----------------------------------------------------
1 - Read File
2 - Show Adjacency Matrix
3 - Find Shortest Path
4 - Exit

Enter your choice: 4

Good bye!
```

There is no incomplete part of the project, all parts are completed by following the instructions in the project document.