

VNUHCM – University of Science

Faculty of Information Technology

[CSC1004]

DATA STRUCTURES & ALGORITHMS

GROUP HOMEWORK 2

GRAPH PROGRAMMING

January 1, 2021

INTRODUCTION

CLASS: 19CLC6 – GROUP 10

MEMBERS: 19127422 – NGUYỄN ĐỨC HUY

19127563 – NGUYỄN HOÀNG THÔNG

19127588 – NGUYỄN BẢO TRÂM

WORK IN PROGRESS



Table of Contents

INTRODUCTION.....	2
WORK IN PROGRESS	2
PROGAMMING.....	3
A. What should your app have?.....	3
B. How should users use your app?	4
C. Could your app have extra features?	5
REFERENCES	6

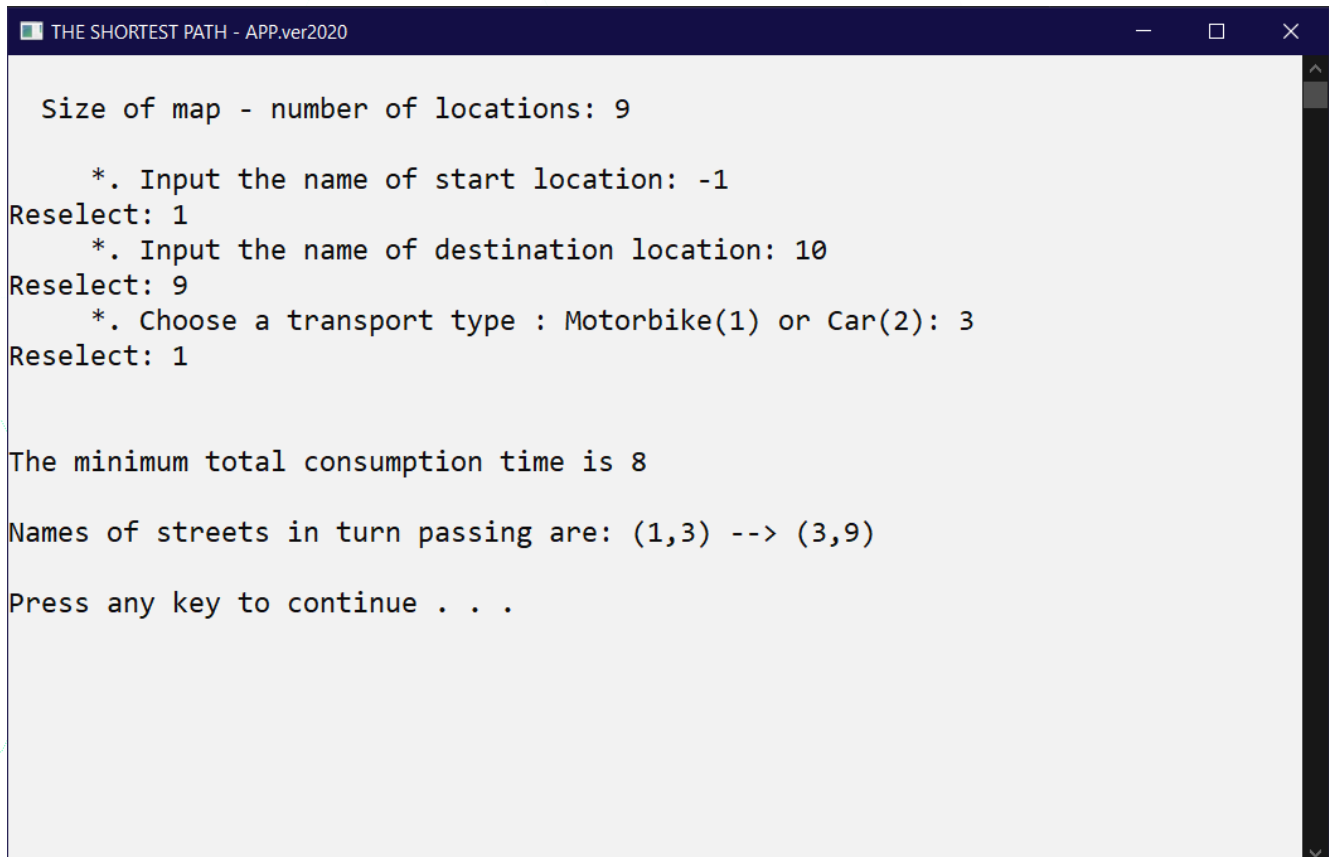
PROGRAMMING

In this homework, we are required to make a simple app for path – finding. Our app has a basic feature: Find the shortest path from A to B. We assume the shortest path from location A to location B is an ordered list of streets with minimum total consumption time.

A. What should our app have?

Our app uses functions to generate a random database in each program run, the default limit for the number of locations in the map is 20. Each time you create a database, the program will write to 3 files: street.txt, transport_type.txt, query.txt.

Additionally, during the database creation process, we will have the stage of checking whether the input value is valid or not before proceeding further. After creating data, the program reads data from the file query.txt to find the shortest path according to user requirements.



```
THE SHORTEST PATH - APP.ver2020

Size of map - number of locations: 9

    *. Input the name of start location: -1
Reselect: 1
    *. Input the name of destination location: 10
Reselect: 9
    *. Choose a transport type : Motorbike(1) or Car(2): 3
Reselect: 1

The minimum total consumption time is 8

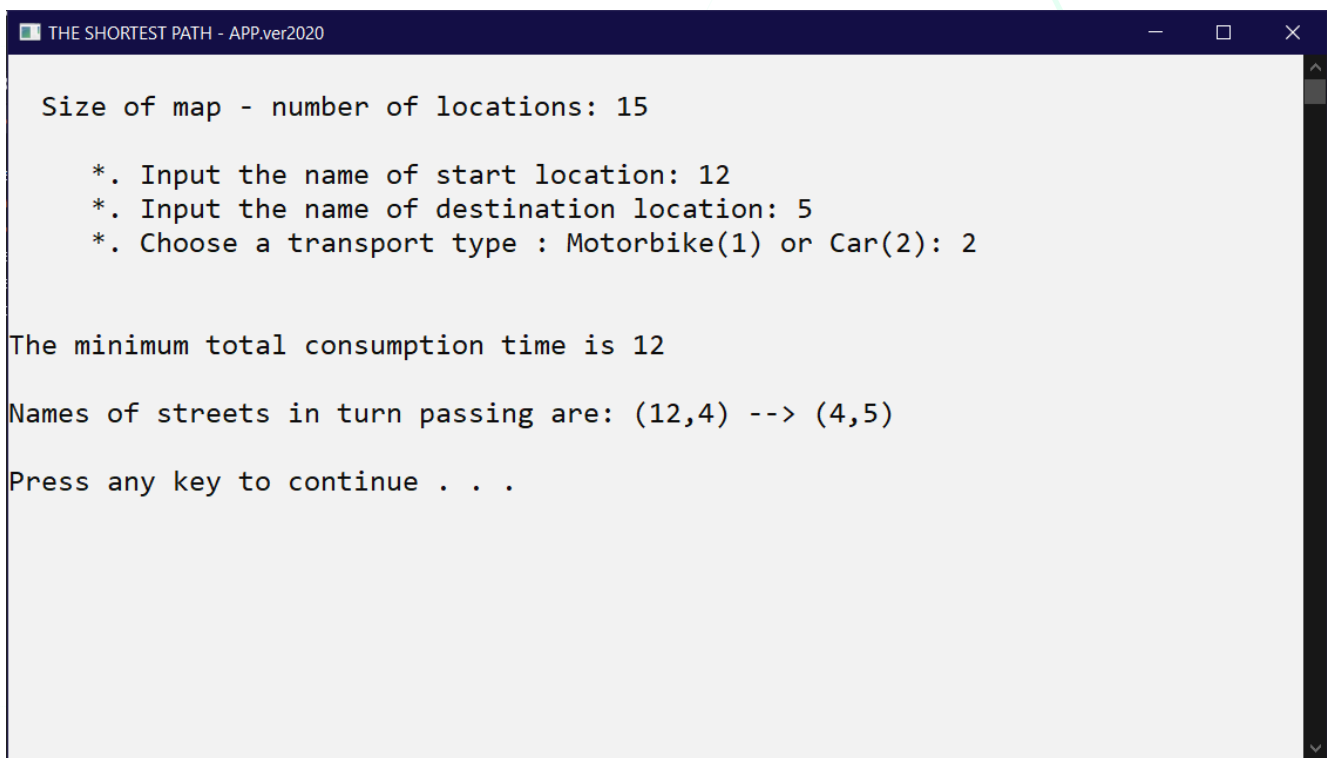
Names of streets in turn passing are: (1,3) --> (3,9)

Press any key to continue . . .
```

B. How should users use our app?

The following steps describe how users use your app:

1. Input the name of start location A.
2. Input the name of destination location B.
3. Choose a transport type: Motorbike or Car.
4. Return the shortest path from A to B (in file)



```
THE SHORTEST PATH - APP.ver2020

Size of map - number of locations: 15

*. Input the name of start location: 12
*. Input the name of destination location: 5
*. Choose a transport type : Motorbike(1) or Car(2): 2

The minimum total consumption time is 12
Names of streets in turn passing are: (12,4) --> (4,5)
Press any key to continue . . .
```

C. Our app has extra features

The basic concept of our app uses Dijkstra's to find the shortest path from locate A to B.

```
int findMin(vector<int>dist, vector<bool>visited) {  
    int min = INT_MAX, index_min = 0;  
  
    for (int i = 0; i < dist.size(); i++) {  
        if (!visited[i] && dist[i] <= min) {  
            min = dist[i];  
            index_min = i;  
        }  
    }  
  
    return index_min;  
}
```

Data type: Using a dynamic array and vector to store variables.

```
void Dijkstra(int** arr, int num, int start, int end) {  
    vector<int>dist(num, INT_MAX);  
    vector<bool>visited(num, false);  
    vector<int>parent(num, 0);  
  
    dist[start] = 0;  
    parent[start] = 0;  
  
    for (int i = 0; i < num - 1; i++) {  
        int u = findMin(dist, visited);  
        visited[u] = true;  
  
        for (int j = 0; j < num; j++) {  
            if (!visited[j] && arr[u][j] > 0 && dist[u] != INT_MAX  
                && (dist[u] + arr[u][j]) < dist[j])  
            {  
                dist[j] = dist[u] + arr[u][j];  
                parent[j] = u;  
            }  
        }  
    }  
}
```

Display: we used the library "Windows.h" to set the color of console and used additional implemented functions

```
// Move the cursor to the position(x; y)
void _Common::GoTo(SHORT posX, SHORT posY)
{
    HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);
    COORD position;
    position.X = posX;
    position.Y = posY;

    SetConsoleCursorPosition(handle, position);
}

void _Common::Goodbye()
{
    system("CLS");
    stringstream color;
    for (int i = 9; i >= 0; i--)
    {
        _Common::GoTo(40, 10);
        color << "color f" << i;
        system((color.str()).c_str()); // background color & text color
        cout << "<< THANK YOU FOR VISTING! GOOD BYE! >>" << endl << endl;
        color.str(std::string());
    }
}
```

```
class _Common
{
public:
    static void GoTo(SHORT, SHORT);
    static void Goodbye();
};
```

REFERENCES

- [1] <https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/>
- [2] [Dijkstra's algorithm – Wikipedia](#)
- [3] [Windows.h Và Hàm Định Dạng Màn Hình Console \(P1\).](#)
- [4] <https://codelearn.io/sharing/windowsh-ham-dinh-dang-noi-dung-console>