PathFinder Manual

Programming in C++, MFF UK

Norbert Horváth

Table of contents

1. Introduction

The PathFinder application is a path-finding algorithm simulator, which supports two types of graphs. The first one being a grid representation of an unweighted graph, the second is an oriented weighted graph, which the user can select, on which one they want to simulate an algorithm. Currently for the unweighted graph there are 5 possible algorithms the user can simulate, for the weighted oriented graph there are currently 4 algorithms. The algorithms for the unweighted graph are the following: Breadth First Search (BFS), Depth First Search (DFS), Dijkstra's algorithm, A Star algorithm and the Greedy Best First Search. For the weighted oriented graph, the software implements the following algorithms: Breadth First Search (BFS), Depth First Search (DFS), Dijkstra's algorithm and the Bellman-Ford algorithm. The application also provides the user with two maze generation techniques, first one being just a simple algorithm, which uses random function to determine with a 33% that a cell would be empty or an obstacle. Please note that these maze generation algorithms are only for the grid, meaning it only is implemented for the unweighted graph. The second one is a more sophisticated backtracker algorithm, which ensures that from every empty cell there is a path to every other empty cell, meaning that there are no isolated cells which are not reachable within an algorithm. This algorithm also ensures that there is a path form the starting node to the ending node. This property cannot be told about the first maze generation algorithm, because it the position of the starting and the ending nodes doesn't affect the algorithm.

2. Getting started

For the installation, please consult the README.md file. Please ensure that you have the following dependencies installed:

- cmake (at least version 3.16)
- c++17 or later versions (c++20, c++23)

After ensuring that you have everything installed correctly, based on the README file you can install the application.

3. Using the application

Upon starting the application, the user is presented with the following user interface

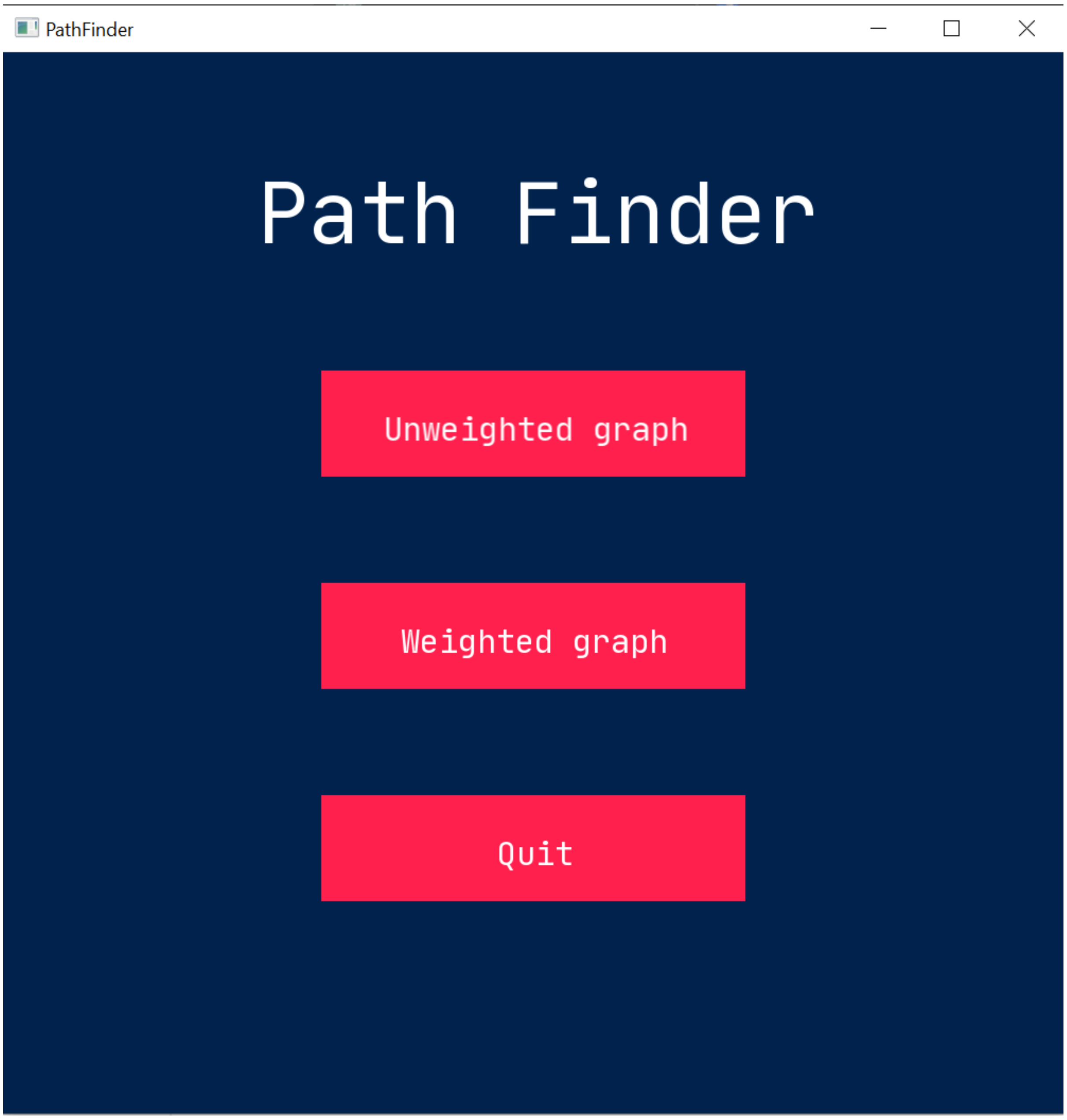


figure 1

Here you will be presented with three buttons. Unweighted graph, Weighted graph and Quit. If you select the “Quit” button, the application will close. If you select the unweighted graph you will be presented with the following user interface:

PathFinder Documentation

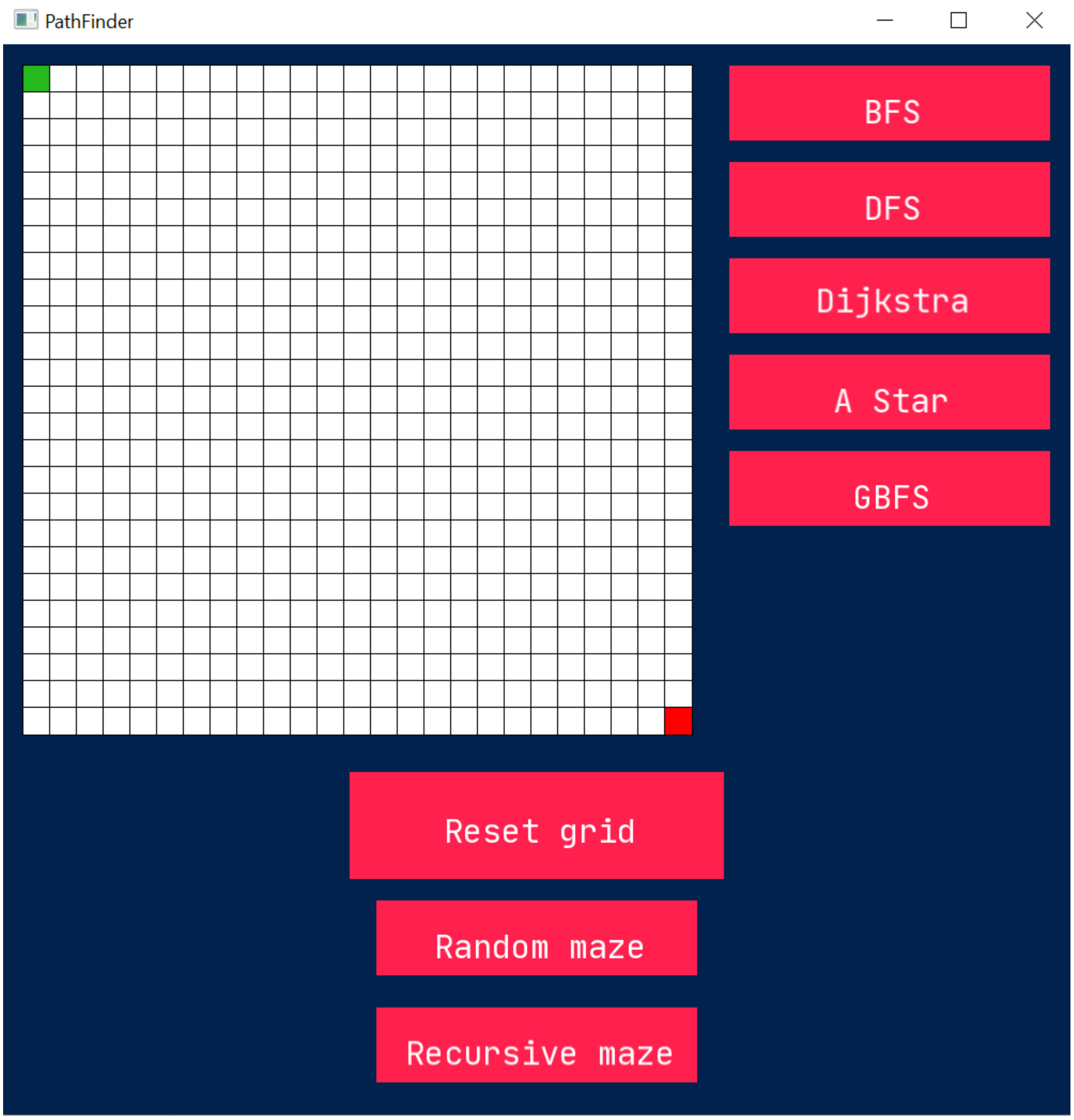


figure 2.1

Here you can see the grid. The green cell represents the starting node of the simulation, the red cell represents the ending node of the simulation. You can change the starting and the ending node of the simulation using your keyboard. If you press down the 'S' key and simultaneously click the left mouse button, you can change the cell. Similarly, you can change the ending node by pressing down the 'E' key and the left mouse button. The 'Reset grid' resets the grid, it deletes all the obstacles drawn to the window but the starting and the ending nodes will stay. It will also automatically stop any algorithm which is running. The 'Random maze' button will generate a random maze, which as previously mentioned uses a random calculation for every cell of the grid. The 'Recursive maze' button also generates a maze, but it has a little more sophisticated algorithm mechanism. The buttons on the top right corner are all path finding algorithms, which you can simulate on the grid.

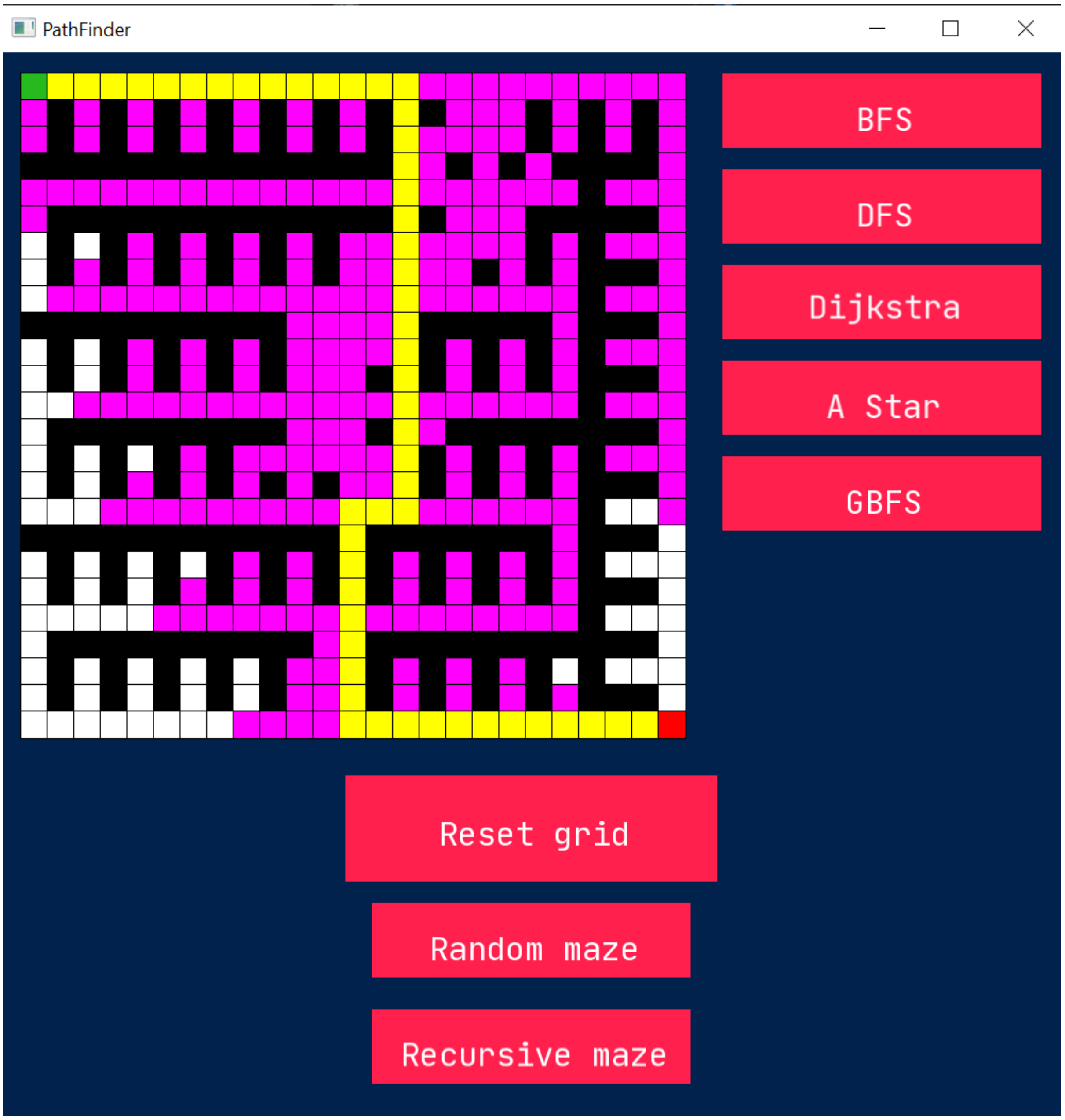


figure 2.2

Figure 2.2 shows the window after an algorithm finished. The purple squares represent the cells which were searched through, the yellow squares represent the path from the starting node to the ending node. The black squares represent the obstacles in the grid.

If you click on the menu the weighted graphs option, you will be presented with the following page:

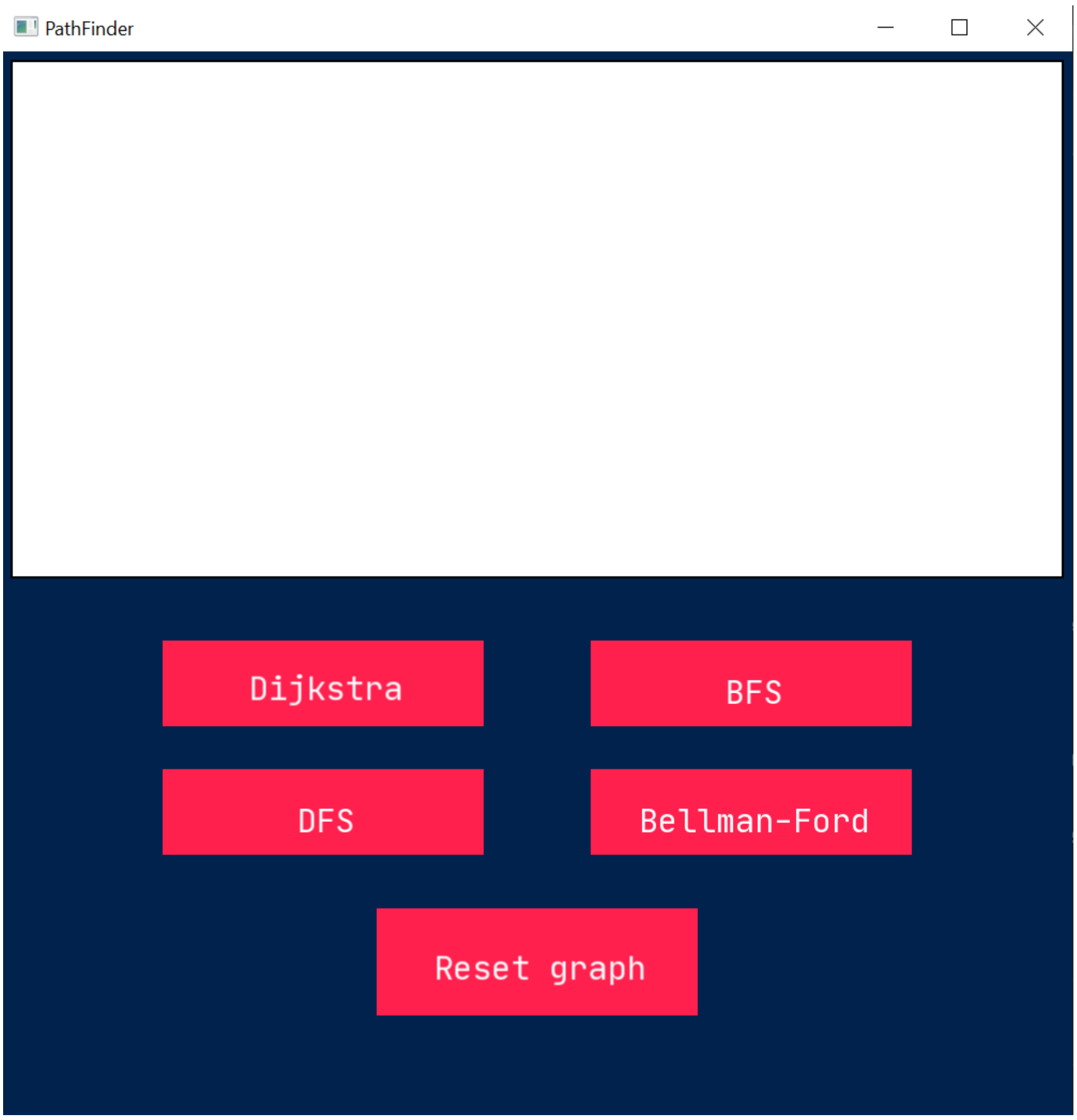


figure 3.1

On the upper side of the window you can see a white rectangle, this is the place you can, using the left mouse button draw vertices. These vertices will be automatically labelled using numbers from zero. You can add edges between vertices if you click on one of the vertices, which will turn cyan, and then you will now, that a vertex is selected. This will look like this:

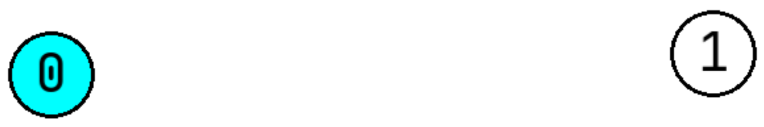
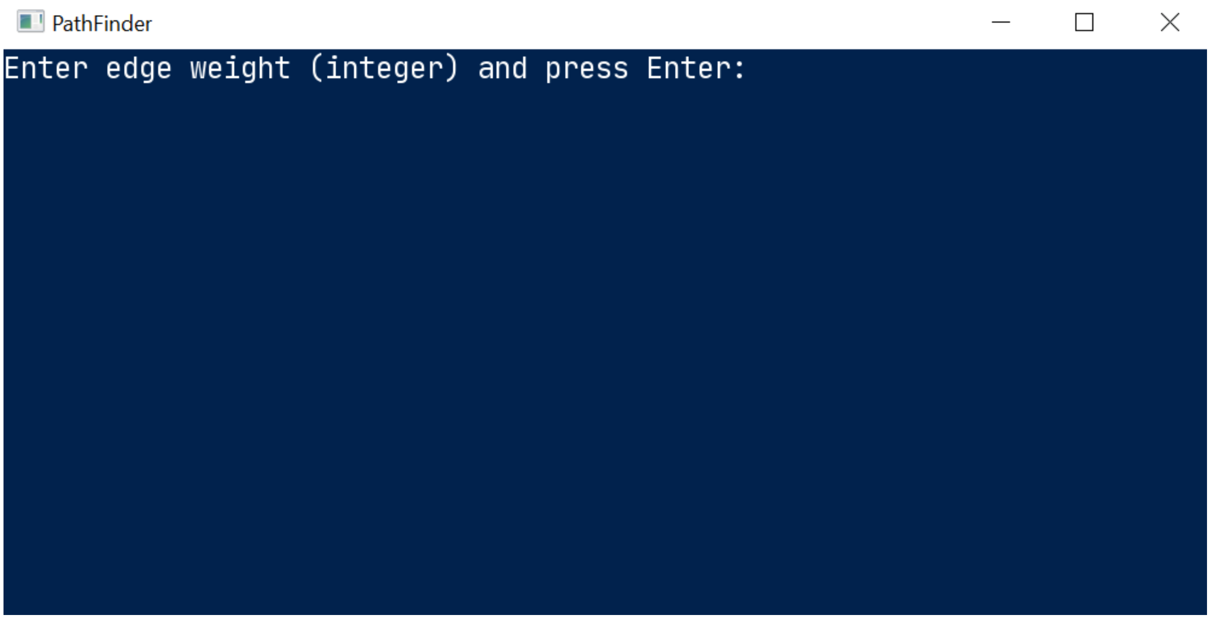


figure 3.2

If you accidentally clicked the node, and you don't want an edge, you can use the right mouse click, to reset the selection. After you click the vertex you want to connect the vertex to (in our case to the node with number 1), a following window will appear:



```
Enter edge weight (integer) and press Enter:
```

figure 3.3

Where you can than define the weight of the edge. After adding a weight to our edge, the graph will look like this:

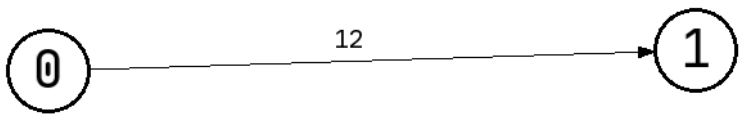


figure 3.4

The ’Reset’ button is used for resetting the grid, meaning that it will completely empty the rectangle of every vertex and edge. If you click any of the other buttons, they will simulate an algorithm, which if finished, will look like this:

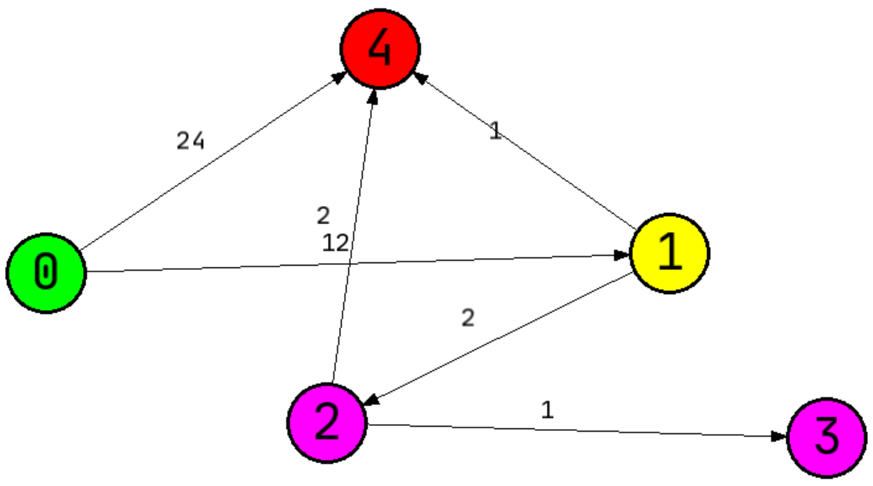


figure 3.5

Here again, the green node represents the starting node, the red node represents the ending node, and the yellow node represents the node in the path of the found path. The magenta ones are the helper nodes, which were searched trough when the algorithm was running. Please note that, the starting end ending nodes are here fixed, the starting node will always be the first node and the ending node will always be the last node, with the highest index. If the algorithm doesn’t find a path, no node will be coloured green, red, or yellow.