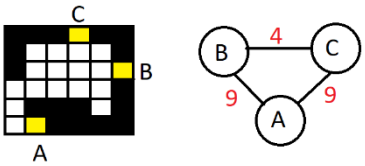
Milestone 2 - Stage 2

In this milestone, you will be asked to integrate the Graph class of *Stage 1* with the template code attached with this document.

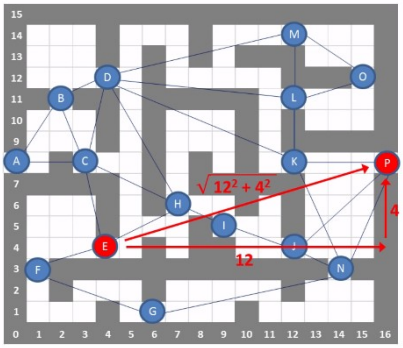
To use the template, you will need to install pygame library. You can use **pip install pygame** to install it from the internet or **pip install <pygame whl file path>**

To do:

1) Maze2Graph.py: Fill in the Transform2Graph method. In this part, your task is to transform the 2D array Maze by finding the weighted edges. For example, the following nodes have connections and can be transformed into a weighted graph as follow:



2) While transforming from maze to graph, add the estimated distance to the target as a value for the graph nodes. Use Euclidian distance for H(n) where n is the node index as follow:



3) Graph.py: Integrate your code in this class for the A Star and Dijkstra algorithms. Note that the shortest path shall contains a list of coordinates as follow [[r1, c1], [r2, c2], …], where each (ri, ci) are the indices of nodes in the maze as row and columns. The array must be sorted from starting node to the target node.

4) MazeBuilder.py: Finalize the Build method, located at line 93. In this part, you shall transform the given shortest path, which is a sorted list of coordinates, into the actual full coordinate sequence. So, for example:
Shortest Path: [[13, 12], [11, 12], [7, 12], [3, 12]]
Therefore, the actual sequence are the full [row, col] that exist between the starting node and the target node as follow:
Full Shortest Path: [[13, 12], [12, 12], [11, 12], [10, 12], [9, 12], [8, 12], [7, 12], [6, 12], [5, 12], [4, 12], [3, 12]]