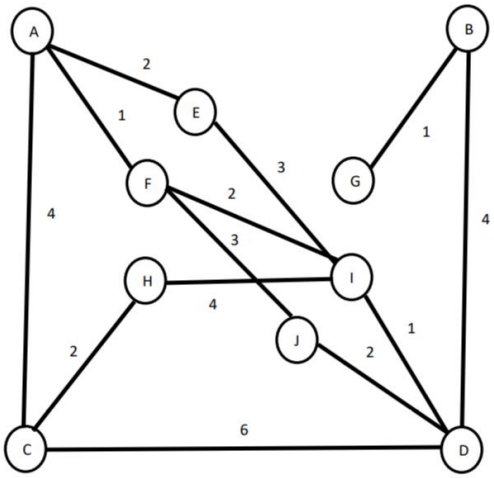
Dijkstra improvement – A*

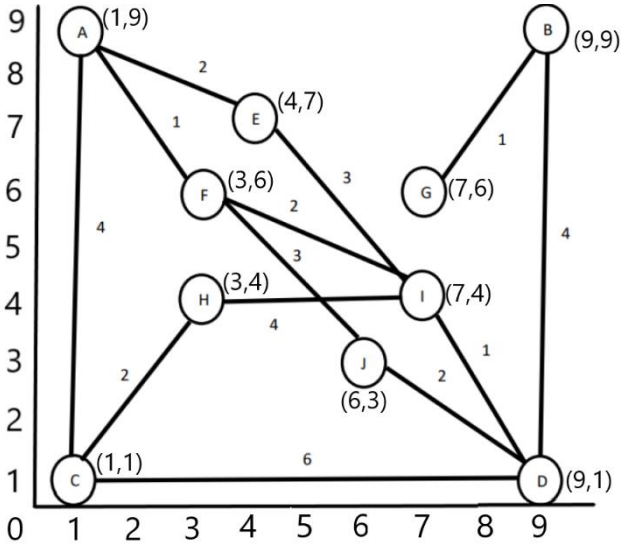


In the Dijkstra shortest path algorithm, it is guaranteed to find the shortest path by analyzing all possibilities. It will analyse all neighboring vertices and decide depending on the least weight of each edge.

Referring to the beside graph, Dijkstra will correctly state the shortest path from G to A is

G -> B -> D -> I -> F -> A

When putting maps into consideration, weights can be assigned representing traffic load, road works, mishaps such as flooding. A* is an improved variation of Dijkstra in order to make decision making quicker. It uses a best-first search approach thanks to a heuristic function. Keeping map navigation in mind, I chose to include the location of each vertex to give the system a sense of direction when choosing vertices for the desired path.



That being said, I implemented extra properties in each vertex to store the location on an X and Y axis.

As part of the decision making, I added a heuristic function calculating the **Euclidean distance** between the vertex in question and the goal vertex.

This prevented the algorithm to not search in directions away from the goal unless really required to do so. By simply adding the locations, my A* implementation would still output the same path as the best path **(G -> B -> D -> I -> F -> A).**

To see the heuristic function taking effect on the decision making, I decided to move vertex I to location (10,0). Whilst keeping the preferred weights, the location of the vertex was out of the way. This affected decision making and the A* output is now **G -> B -> D -> J -> F -> A.**

The A* not only manages to find optimal paths in a more logical way but performs ~30% faster than traditional Dijkstra.