To calculate A* Path, we require three values.

- gScore – A score from the current block to the new evaluated block
- hScore – A score from the new evaluated block to the end block. hScore or Heuristic Score is nothing but the euclidean distance the current block and the end block.
- fScore – gScore + hScore

```
openSet   = {start}
closedSet = an empty array

//For node n, gScore[n] is the cost of the cheapest path from start to n currently known.
gScore        = map with default value of Infinity
gScore[start] = 0

//For node n, fScore[n] := gScore[n] + heuristicScore(n). fScore[n] represents our current
//best guess as to how short a path from start to finish can be if it goes through n.
fScore        = map with default value of Infinity
fScore[start] = heuristicScore(start)

while openSet is not empty
    current = the node in openSet having the lowest fScore[] value
    if current = goal
        return reconstruct_path(cameFrom, current)

    openSet.Remove(current)
    for each neighbor of current
        // tentative_gScore is the distance from start to the neighbor through current
        tentative_gScore := gScore[current] + 1
        if tentative_gScore < gScore[neighbor]
            // This path to neighbor is better than any previous one. Record it!
            cameFrom[neighbor] := current
            gScore[neighbor] := tentative_gScore
            fScore[neighbor] := gScore[neighbor] + heuristicScore(neighbor)
            if neighbor not in openSet
                openSet.add(neighbor)

//Open set is empty but goal was never reached
return failure
```