Logic behind implement A* algorithm

First I consider grid cell to explain the path planning

Cell no	0	1	2	3	4
0					
1					
2					
3					
4					

Cell which color black is my obstacle cell. I can define obstacle in my code as 1.

1 means previously occupied cell. (0,0) is my starting position and (4,4) is my goal position.

In my program start position denote as init and goal position denote as goal.

I create a arraylist open which use for cell expansion. Initially open list hold (0,0) Now I expand the cell looking at it's successor which is (1,0).

1.Open=[0, 0] I associate a value for each movement. This value is called g. In this stage g value =0

2.Open= [1, 0] g value =1

3. Open=[2, 0] g value =2

4.open = [3 ,0] g value =3 [2,1] g value =3 Always expand the cell depend upon smallest g value but as it is equal it doesn't make any difference which one expand first. And so on.

```
int[][] Expand= new int[grid.length][grid[0].length];
```

Expand maintain the expansion information. As per the programming if any cell fail to expand it will hold the value -1. The obstacle cells are never be expanded so they assigned -1.

PathX and PathY use to find out the x co – ordinate by which the robot move and PathY find out the Y co ordinate of robot path.

A* uses a heuristics function. To setup this heuristic function I implemented h which have same size of the maze. The heuristic matrix calculate what is the number of steps it takes to get a goal if there is no obstacle. These numbers are not real of the actual distance to the goal, because they do not take into consideration the obstacles. If there is no obstacle the heuristic function would give me the distance to

the goal. So, the heuristic function has to be an optimistic guess of how far I am from the goal. The heuristic function has to be a function that helps us to find out where to search next in the case of ties.

So after using heuristic function f will be return the cost value of searching,

So,

$f=g+h(x,y)$

So actually the cell will be expanded depend upon lowest f value.