AI-LAB 2.

Dhanraj .K
1BM18CS027
5A

10/10/20

Q) A* algorithm.

To find minimum Distance from Source to destination provided

```
# Heuristic function as Euclidian distance
def euclidDistance(x, n, m):
    dist = math.sqrt((n-1 - x[0])** 2 + (m-1 - x[1])** 2)
    return dist.

# To find shortest Path.

def astar(nextP, n, m):
    minDistance = 999.
    next = []
    for x in nextPath:
        if (euclidDist(x, n, m) < minDistance):
            minDistance = euclidDist(x, n, m).
            next = x.
    return next.

# find path.

def findPath(n, m):
    Path.append([0, 0])
    current = [0, 0]
    while (current != [n-1, m-1]):
        nextPath = []
```

```
for x in neighbours:
    a=[]
    a.append(current[0] +x[0])
    a.append( current[1]+x[1])
    if a[0]> -1 and a[0]<n and a[1]> -1
       and a[1] <m:
        if (maze[a[0]] [a[1]]):
            if a not in path and a not in
                                   closed path:
                nextPath.append(a)

if (nextPath):
    current = findshortest Path (next Path,n,m)
    Path.append (current)

else:
    if path:
        closed Path .append (current)
        Path.pop()
        if Path:
            current = path[len(path) -1]
        else:
            Print ("PATH' Cant be found")
            exit (0)
    else:
        Print ("Path cant be found")
        exit(0)
```