Name: Athcon Arya
USN: 1BM18CS020

AI - TEST-1

⇒ Maze structure

```
import math

maze []
#hatPath []
closedPath []
neighbours [[-1, 1], [0, 1], [1, 0], [1, -1], [0, -1], [-1, -1],
            [-1, 0], [-1, -1]]

def euclidDist (x, n, m):
    dist = math.sqrt ((n-1-x[0]) ** 2 + (m-1-x[1]) ** 2)
    return dist

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

    return next

def findPath (n, m):
    Path.append ([0,0])
```

```
current=[00]

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.appen(a)

if(nextpath):
 current = findShortestpath(nextpath,n,m
 path.appen(current)
 Path.pop()
 if path:
  current = path[len(path)-1]
ELSE:
 Print("No Path")
 Exit(0)
Else:
 Print("No Path")
 Exit(0)
```

```
def Start():
    n = int(input("\n number of rows: "))
    m = int(input("\n number of cols: "))

Print("\n Enter the Maze Structure: (0-blocked, 1-free):
                                                    \n")
for i in range(n):
    a = []
    a = list(map(int, input().split(" ")))
    Maze.append(a)

Print("\n\n *** Maze ***")
for i in range(n):
    for j in range(m):
        Print(Maze[i][j], end = " ")
    Print()

find path(n, M)

Print("\n * Path * ")
for i in range(n):
    for j in range(M):
    if (i, j) in path):
        Print("-", end = " ")
    else:
        Print(Maze[i][j], end = " ")
    Print()

Print()
Print(path)
if __name__ == "__main__":
        Start()
```