Name: Athcon Arya
USN: 1BM18CS020

AI - Test - 1

⇒ Maze Structure

```
import math

Maze
Phat Path
Closed Path
neighbours


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])
```

Ans.

```
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("Number of rows: "))
    m = int(input("Number of cols: "))

print("Enter the Maze structure: (0-blocked, 1-free)")

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()
```

Scanned with CamScanner