S-1

	O	C	G	F	E	R
Visited	0	0	0	0	0	0

queue:

print:

S-2

	O	C	G	F	E	R
Visited	1	0	0	0	0	0

queue: O

print:

S-3

	O	C	G	F	E	R
visited	1	0	0	0	0	0

queue

print O

S-4

	O	C	G	F	E	R
Visited	1	1	0	0	0	0

C

queue

print O

S-5

	O	C	G	F	E	R
Visited	1	1	1	0	0	0

queue G

print O C

S-6

	O	C	G	F	E	R
Visited	1	1	1	1	0	0

queue F

print O C G

S-7

	O	C	G	F	E	R
Visited	1	1	1	1	1	0

queue E

print O C G F

S-8

	O	C	G	F	E	R
Visited	1	1	1	1	1	1

queue R

print O C G F E

	O	C	G	F	E	R
Visited	1	1	1	1	1	1

queue

print O C G F E R

→ O, Visited - O, queue -

→ O, Visited - 1, queue - O

add O to queue and mark visited

→ O, Visited - 1, queue - , print - O

remove O to queue & print O

→ Visited O, C, K.

1 1 1

Queue CK

C and K is added to the queue, Mark C & K as visited

→ Visited : O, C, K

1 1 1

Queue : K

C is removed from the queue, O, C are printed

Visited O C K G

Queue: K G

Add G to the queue

Visited: O C K G

Queue: G

Remove K from the queue, print O C K

Visited O C K G

Queue:

Remove G from the queue, print O C K G

Visited : O C K G D

Queue: D

Add D to the queue and mark D as it is visited.

Visited: O C K G D

Queue:

Remove D from the queue, print O C K G D

Visited: O C K G D A I

Queue: A I

Add A, I to queue and mark as visited.

Visited: O C K G D A I
1 1 1 1 1 1 1

Queue: I

Remove A from the queue, print O C K G D A

Visited O C K G D A I B
1 1 1 1 1 1 1 1

Queue: I B

Add B to queue and Mark B as it is visited

Visited: O C K G D A I B R
1 1 1 1 1 1 1 1 1

Queue: I R

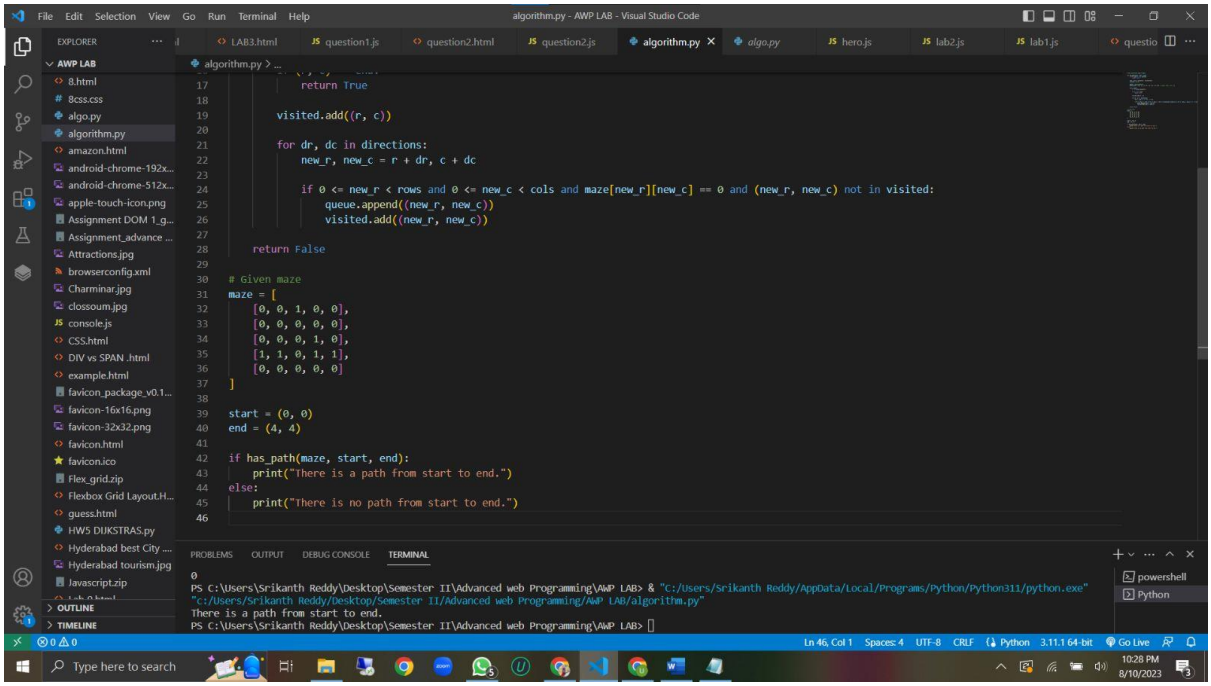
Add R to the queue and mark as visited.

Code

```
from collections import deque


def has_path(maze, start, end):
    if not maze or not maze[0]:
        return False

    rows, cols = len(maze), len(maze[0])
    visited = set()

    queue = deque([start])
    directions = [(0, 1), (1, 0), (0, -1), (-1, 0)]  # Right, Down, Left, Up

    while queue:
        r, c = queue.popleft()

        if (r, c) == end:
            return True

        visited.add((r, c))

        for dr, dc in directions:
            new_r, new_c = r + dr, c + dc

            if 0 <= new_r < rows and 0 <= new_c < cols and maze[new_r][new_c] == 0 and (new_r, new_c)
not in visited:
                queue.append((new_r, new_c))
                visited.add((new_r, new_c))

    return False
```

```
# Given maze
maze = [
    [0, 0, 1, 0, 0],
    [0, 0, 0, 0, 0],
    [0, 0, 0, 1, 0],
    [1, 1, 0, 1, 1],
    [0, 0, 0, 0, 0]
]

start = (0, 0)
end = (4, 4)

if has_path(maze, start, end):
    print("There is a path from start to end.")
else:
    print("There is no path from start to end.")
```

Output:


File Edit Selection View Go Run Terminal Help
algorithm.py - AWP LAB - Visual Studio Code
EXPLORER
AWP LAB
8.html
8css.css
algo.py
algorithm.py
amazon.html
android-chrome-192x...
android-chrome-512x...
apple-touch-icon.png
Assignment DOM 1_g...
Assignment_advance ...
Attractions.jpg
browserconfig.xml
Charminar.jpg
clossoum.jpg
console.js
CSS.html
DIV vs SPAN .html
example.html
favicon_package_v0.1...
favicon-16x16.png
favicon-32x32.png
favicon.html
favicon.ico
Flex_grid.zip
Flexbox Grid Layout.H...
guess.html
HW5 DIJKSTRAS.py
Hyderabad best City ....
Hyderabad tourism.jpg
Javascript.zip
OUTLINE
TIMELINE
LAB3.html
question1.js
question2.html
question2.js
algorithm.py
algo.py
hero.js
lab2.js
lab1.js
algorithm.py > ...
return True
visited.add((r, c))
for dr, dc in directions:
new_r, new_c = r + dr, c + dc
if 0 <= new_r < rows and 0 <= new_c < cols and maze[new_r][new_c] == 0 and (new_r, new_c) not in visited:
queue.append((new_r, new_c))
visited.add((new_r, new_c))
return False
# Given maze
maze = [
[0, 0, 1, 0, 0],
[0, 0, 0, 0, 0],
[0, 0, 0, 1, 0],
[1, 1, 0, 1, 1],
[0, 0, 0, 0, 0]
]
start = (0, 0)
end = (4, 4)
if has_path(maze, start, end):
print("There is a path from start to end.")
else:
print("There is no path from start to end.")
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
0
PS C:\Users\Srikanth Reddy\Desktop\Semester II\Advanced web Programming\AWP LAB> & "C:/Users/Srikanth Reddy/AppData/Local/Programs/Python/Python311/python.exe"
"c:/Users/Srikanth Reddy/Desktop/Semester II/Advanced web Programming/AWP LAB/algorithm.py"
There is a path from start to end.
PS C:\Users\Srikanth Reddy\Desktop\Semester II\Advanced web Programming\AWP LAB>
powershell
Python
Ln 46, Col 1
Spaces: 4
UTF-8
CRLF
Python
3.11.1 64-bit
Go Live
Type here to search
10:28 PM
8/10/2023