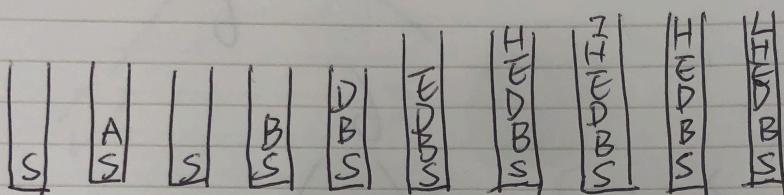
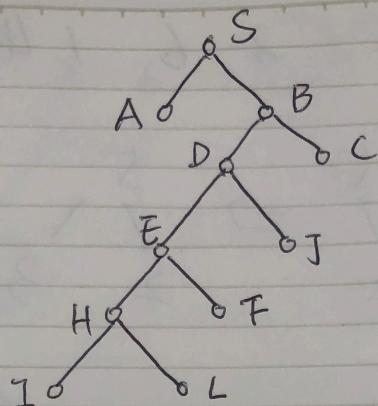
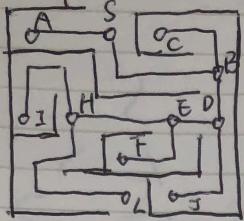
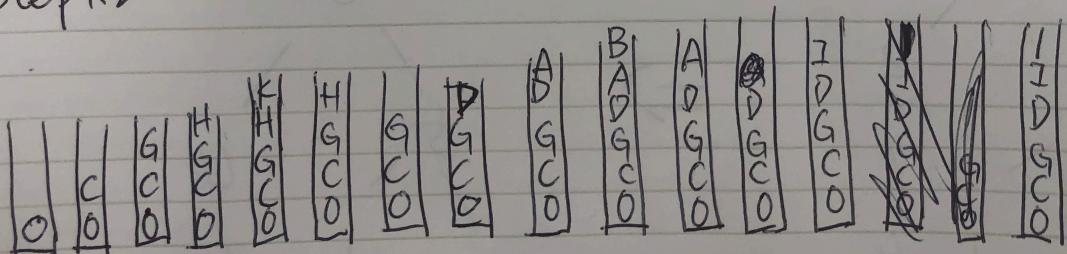


Step 1.1



Step 1.2



class Solution:

```
def hasPath(self, maze, start, destination):
    self.Row, self.Col = len(maze), len(maze[0])
    self.destination = destination
    self.visited = [[False for _ in range(self.Col)] for _ in range(self.Row)]
    return self.dfs_backtrace(maze, start[0], start[1])
```

```

def dfs_backtrace(self, maze, r, c):
    if [r,c] == self.destination:
        return True
    for dr,dc in ((0,1), (1,0), (0,-1), (-1,0)):
        nr = r + dr
        nc = c + dc
        while 0 <= nr < self.Row and 0 <= nc < self.Col and maze[nr][nc] == 0:
            nr += dr
            nc += dc
        nr -= dr
        nc -= dc
        if self.visited[nr][nc] == False:
            self.visited[nr][nc] = True
            if self.dfs_backtrace(maze, nr, nc) == True:
                return True
            self.visited[nr][nc] = False
    return False

def main():
    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 = [0,4]
    destination = [4,4]
    print(Solution().hasPath(maze, start, destination))

if __name__ == "__main__":
    main()

```

Output:

```

● (vizenv) ➔ python course /Users/shanpeng/software/anaconda/anaconda3/envs/vizenv/bin/python "/Use
rs/shanpeng/Documents/SFBU/python course/test1.py"
True
○ (vizenv) ➔ python course □

```