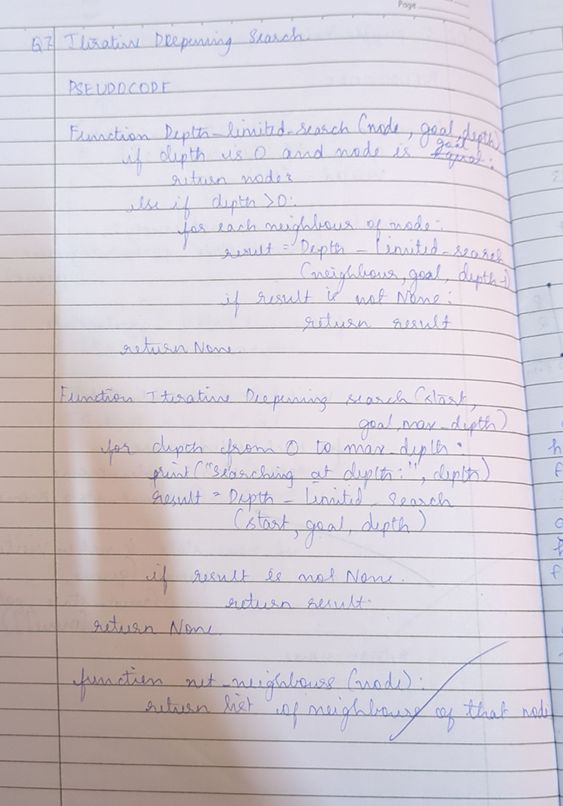
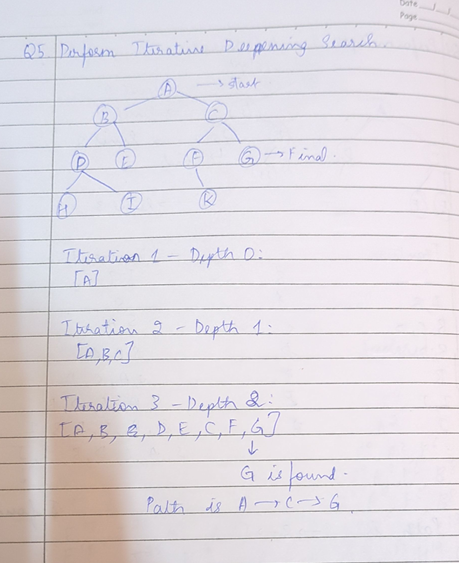
**Iterative Deepening Search**

**Algorithm:**

****

**State Space Tree:**

****

**Code:**

def depth\_limited\_search(node, goal, depth):

if depth == 0 and node == goal:

return node # Goal found

if depth > 0:

for neighbor in get\_neighbors(node):

result = depth\_limited\_search(neighbor, goal, depth - 1)

if result is not None:

return result

return None

def iterative\_deepening\_search(start, goal, max\_depth):

for depth in range(max\_depth):

print(f"Searching at depth: {depth}")

result = depth\_limited\_search(start, goal, depth)

if result is not None:

return result

return None

def get\_neighbors(node):

neighbors = {

'A': ['B', 'C'],

'B': ['D', 'E'],

'C': ['F', 'G'],

'D': [],

'E': [],

'F': [],

'G': []

}

return neighbors.get(node, [])

start\_node = 'A' # Start node

goal\_node = 'G' # Goal node

max\_depth = 3 # Maximum depth to search

result = iterative\_deepening\_search(start\_node, goal\_node, max\_depth)

if result:

print(f"Goal {goal\_node} found!")

else:

print(f"Goal {goal\_node} not found within depth limit.")

**Output:**

