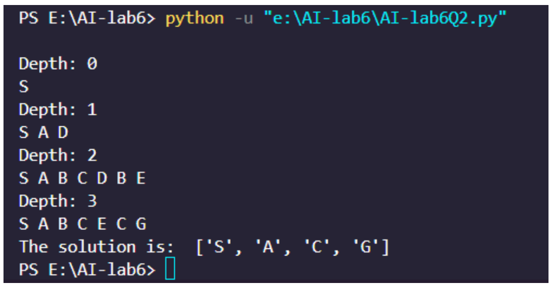
```python
def depth_limited_search(graph, current_node, goal, limit):
    print(current_node, end= ' ')
    if current_node == goal:
        return [current_node]
    if limit == 0:
        return None
    for neighbor in graph.get(current_node, []):
        result = depth_limited_search(graph, neighbor, goal, limit - 1)
        if result is not None:
            return [current_node] + result
    return None

def iterative_deepening_search(graph, start_node, goal):
    depth = 0
    while True:
        print(f"\nDepth: {depth}")
        result = depth_limited_search(graph, start_node, goal, depth)
        if result is not None:
            return result
        depth += 1

graph = { 'S': ['A', 'D'],
         'A': ['B', 'C'],
         'B': ['C', 'E'],
         'C': ['G'],
         'D': ['B', 'E'],
         'E': ['G'],
         'G': [] }


result = iterative_deepening_search(graph, 'S', 'G')
print("\nThe solution is: ", result)
```



```
PS E:\AI-lab6> python -u "e:\AI-lab6\AI-lab6Q2.py"

Depth: 0
S
Depth: 1
S A D
Depth: 2
S A B C D B E
Depth: 3
S A B C E C G
The solution is:  ['S', 'A', 'C', 'G']
PS E:\AI-lab6>
```