Code 1 dfs

```
def DFS(g, start, goal):
   visited = []
   stack = [start]
   while stack:
      node = stack.pop()
      if node==goal:
       visited.append(node)
       return visited
      if node not in visited:
         visited.append(node)
         new=g.get(node)
      if new:
       stack.extend(new)
   return visited


g = {1: [2, 3], 2: [4, 5], 3: [6, 7]}
path = DFS(g, 1, 6)
print(path)
```

Code 2: dfs tree

```
def dfs_tree(tree, start, end):
   visited = set()
   traversal_order = []
   stack = [(start, [start])]
   while stack:
      node, path = stack.pop()
      if node not in visited:
         visited.add(node)
         traversal_order.append(node)
         print(f"node {node} visited")


      if node == end:
         return visited, traversal_order


      for child in reversed(tree[node]):
         if child not in visited:
```

```
            stack.append((child, path + [child]))

    return visited, traversal_order

tree = {1: [2, 3], 2: [4, 5], 3: [6, 7], 4: [2], 5: [2], 6: [3], 7: [3]}
start = 1
end = 6

visited, traversal_order = dfs_tree(tree, start, end)
print("DFS Traversal Order:", traversal_order)
print("Visited Nodes:", visited)
```

Code 3: dls

```
print("Enter the depth that you want to traverse to")
d=int(input())
def DFS(g, start, goal,d):
    visited = []
    path = []
    stack = [start]
    if start==goal:
      print("star tand the goal is same")
    else:
        depth=0

        while(depth<=d):
          node = stack.pop()

          if node not in visited:
            visited.append(node)
            new=g.get(node)
            depth=depth+1
            print("depth=",depth)

          if node==goal:
            return visited
          if new:
```

```
            stack.extend(new)
    return visited


g = {1: [2, 3], 2: [4, 5], 3: [6, 7],7:[8,9]}
visited = DFS(g, 1, 4,d)
print(visited)
```