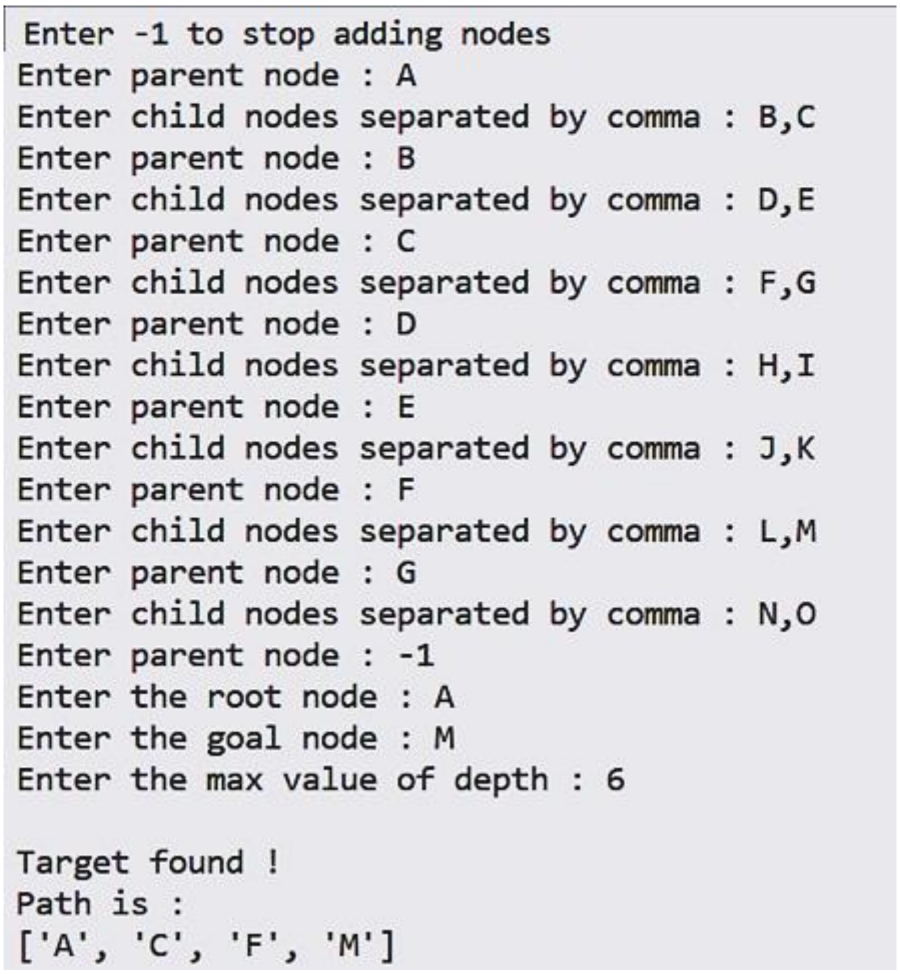
Program:

```
def expand(node):
  if node not in graph: return []
  return graph[node]
def iddfs(root, goal, maxDepth): depth = 0
  while depth <= maxDepth:
    result, path = dls(root, goal, depth, []) if
    result == goal:
      return path
    depth += 1
def dls(node, goal, depth, path):
  path.append(node) if node == goal:
    return node, path
  elif depth > 0:
    for child in expand(node):
      result, new_path = dls(child, goal, depth - 1, path.copy()) if
      result == goal:
        return result, new_path
  return None, path[:-1]
graph = {} print("Enter -1 to stop adding nodes")
while True:
  parent = input("Enter parent node : ") if parent == "-1":
  break c = input("Enter childern nodes separated by comma : ")
  children = c.split(",") graph[parent] = children
root = input("Enter the root node : ") goal =
input("Enter the goal node : ") result =
iddfs(root, goal, 5)
if result == None:
  print("\nTarget not found within the depth limit")
else:
  print("\nTarget found !\nPath is : ") print(result)
```

Output:



```
Enter -1 to stop adding nodes
Enter parent node : A
Enter child nodes separated by comma : B,C
Enter parent node : B
Enter child nodes separated by comma : D,E
Enter parent node : C
Enter child nodes separated by comma : F,G
Enter parent node : D
Enter child nodes separated by comma : H,I
Enter parent node : E
Enter child nodes separated by comma : J,K
Enter parent node : F
Enter child nodes separated by comma : L,M
Enter parent node : G
Enter child nodes separated by comma : N,O
Enter parent node : -1
Enter the root node : A
Enter the goal node : M
Enter the max value of depth : 6

Target found !
Path is :
['A', 'C', 'F', 'M']
```