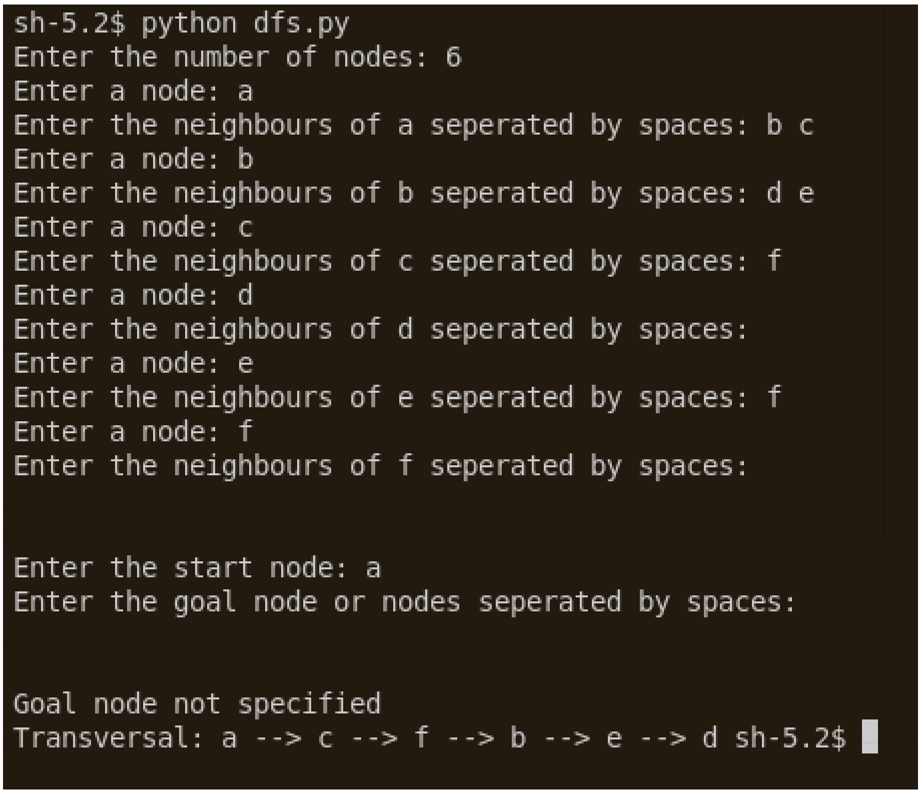
CODE:

```
def movegen(current, graph):
    return [[n, current] for n in graph[current]]


def goal_test(current, goal):
    return current in goal


def traversal(closed):
    print("Traversal ", end="")

    for i in range(len(closed)):
        print(closed[i][0], end=" ")
        if i != len(closed) - 1:
            print("-->", end=" ")


def dfs(graph):
    open = []
    closed = []

    start_node = input("\n\nEnter the start node: ")
    goal_node = input("Enter the goal node or nodes seperated by 
spaces: ").split()

    open.append([start_node,None])

    while open:
        current = open[0][0]
        closed.append(open[0])
        del open[0]
        if goal_test(current, goal_node):
            print("\n\nGoal found\n")
            traversal(closed)
            return
        else:
            child_list = movegen(current, graph)
            for child in child_list:
                if any(child[0] == o[0] for o in open) or 
any(child[0] == c[0] for c in closed):
                    continue
                open.insert(0, child)
    if not goal_node:
        print("\n\nGoal node not specified")
        traversal(closed)
    else:print("\n\nGoal not found")


def input_graph():
    graph = {}

    n = int(input("Enter the number of nodes: "))

    for i in range(n):
        node = input("Enter a node: ")
        neighbours = input(f"Enter the neighbours of {node}
seperated by spaces: ").split()
        graph[node] = neighbours

    return graph


if __name__ == "__main__":
    graph = input_graph()
    dfs(graph)
    print()
```

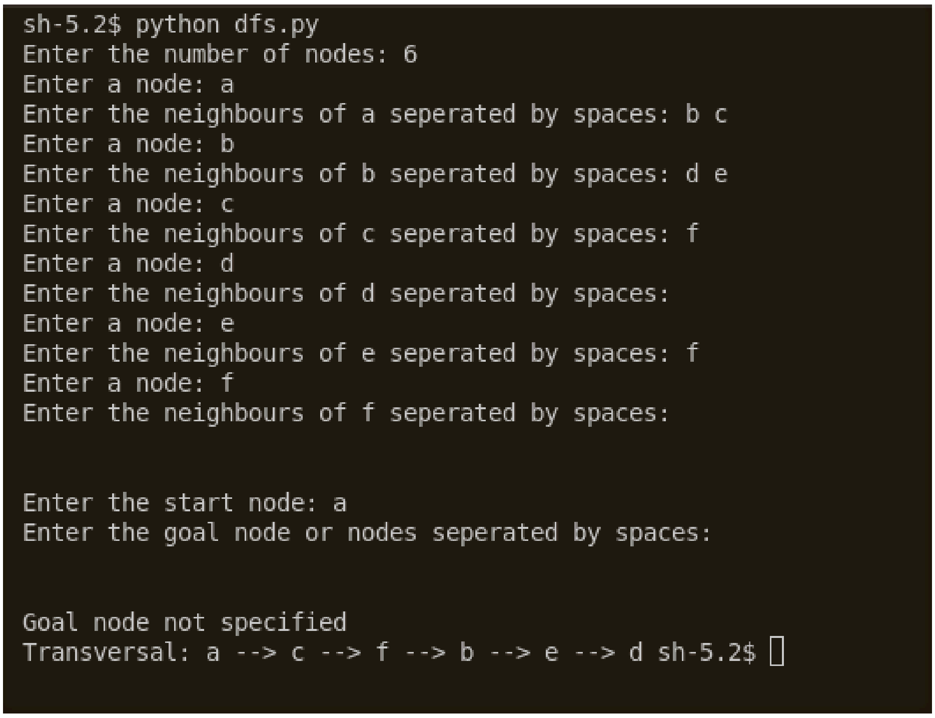
OUTPUT:

1. Initial state is the goal state



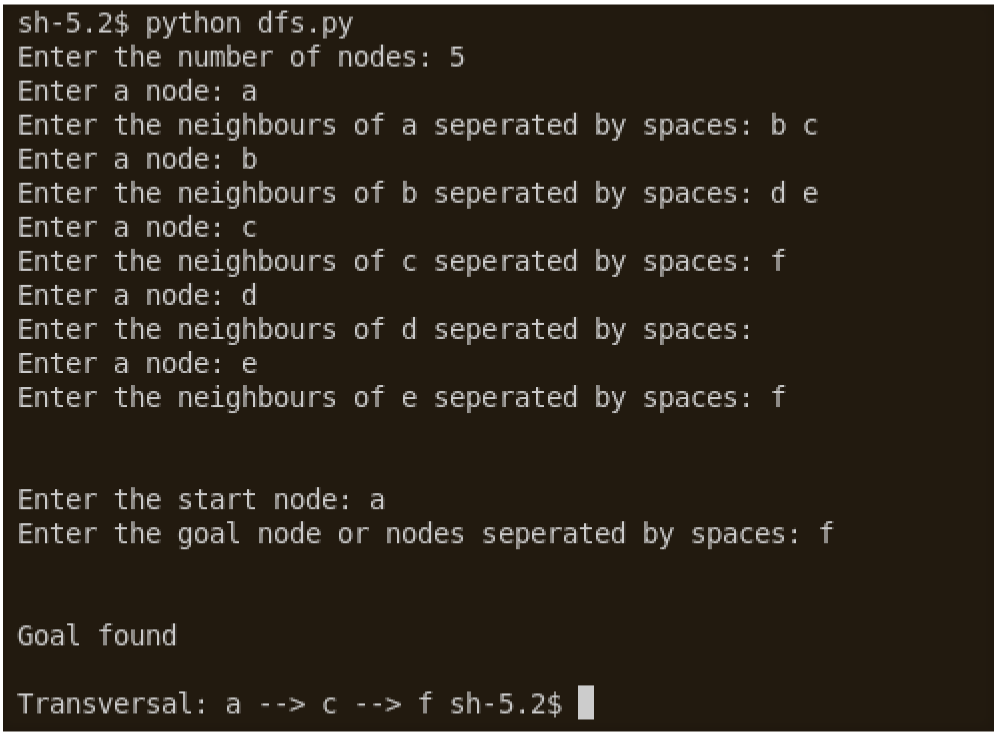
```
sh-5.2$ python dfs.py
Enter the number of nodes: 6
Enter a node: a
Enter the neighbours of a seperated by spaces: b c
Enter a node: b
Enter the neighbours of b seperated by spaces: d e
Enter a node: c
Enter the neighbours of c seperated by spaces: f
Enter a node: d
Enter the neighbours of d seperated by spaces:
Enter a node: e
Enter the neighbours of e seperated by spaces: f
Enter a node: f
Enter the neighbours of f seperated by spaces:


Enter the start node: a
Enter the goal node or nodes seperated by spaces:


Goal node not specified
Transversal: a --> c --> f --> b --> e --> d sh-5.2$
```

2. Goal state is not specified



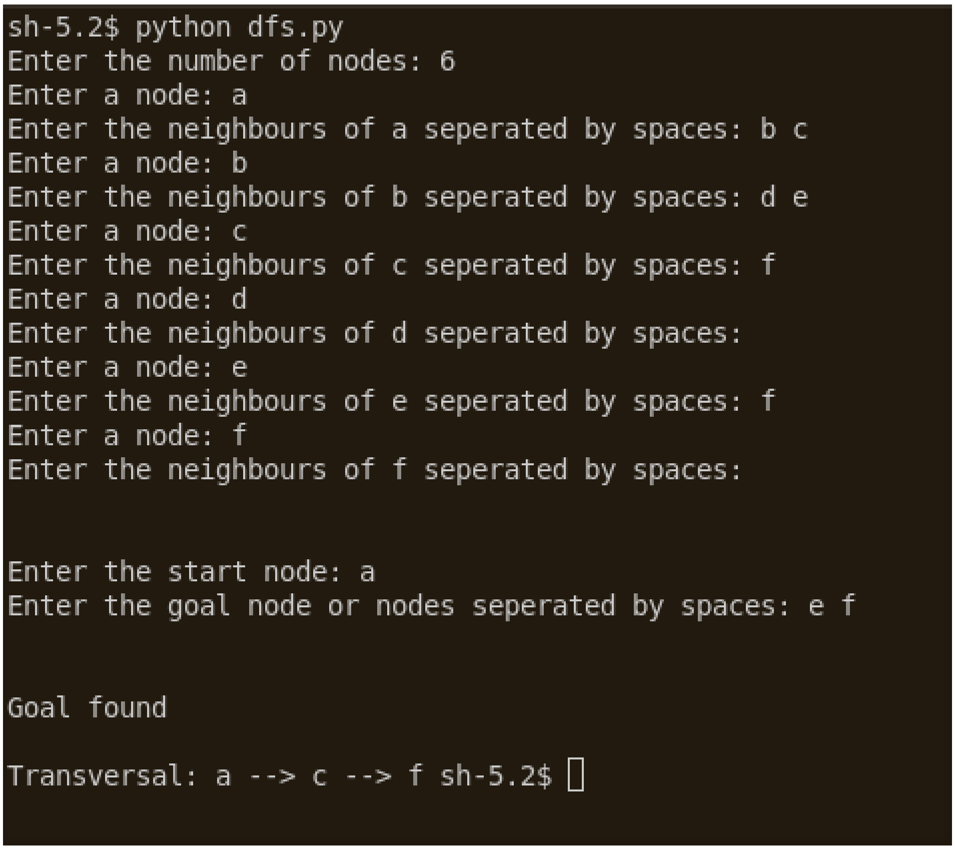
```
sh-5.2$ python dfs.py
Enter the number of nodes: 6
Enter a node: a
Enter the neighbours of a seperated by spaces: b c
Enter a node: b
Enter the neighbours of b seperated by spaces: d e
Enter a node: c
Enter the neighbours of c seperated by spaces: f
Enter a node: d
Enter the neighbours of d seperated by spaces:
Enter a node: e
Enter the neighbours of e seperated by spaces: f
Enter a node: f
Enter the neighbours of f seperated by spaces:


Enter the start node: a
Enter the goal node or nodes seperated by spaces:


Goal node not specified
Transversal: a --> c --> f --> b --> e --> d sh-5.2$
```

3. 1 goal state



```
sh-5.2$ python dfs.py
Enter the number of nodes: 5
Enter a node: a
Enter the neighbours of a seperated by spaces: b c
Enter a node: b
Enter the neighbours of b seperated by spaces: d e
Enter a node: c
Enter the neighbours of c seperated by spaces: f
Enter a node: d
Enter the neighbours of d seperated by spaces:
Enter a node: e
Enter the neighbours of e seperated by spaces: f


Enter the start node: a
Enter the goal node or nodes seperated by spaces: f


Goal found

Transversal: a --> c --> f sh-5.2$
```

4. More than 1 goal state



```
sh-5.2$ python dfs.py
Enter the number of nodes: 6
Enter a node: a
Enter the neighbours of a seperated by spaces: b c
Enter a node: b
Enter the neighbours of b seperated by spaces: d e
Enter a node: c
Enter the neighbours of c seperated by spaces: f
Enter a node: d
Enter the neighbours of d seperated by spaces:
Enter a node: e
Enter the neighbours of e seperated by spaces: f
Enter a node: f
Enter the neighbours of f seperated by spaces:


Enter the start node: a
Enter the goal node or nodes seperated by spaces: e f


Goal found

Transversal: a --> c --> f sh-5.2$
```