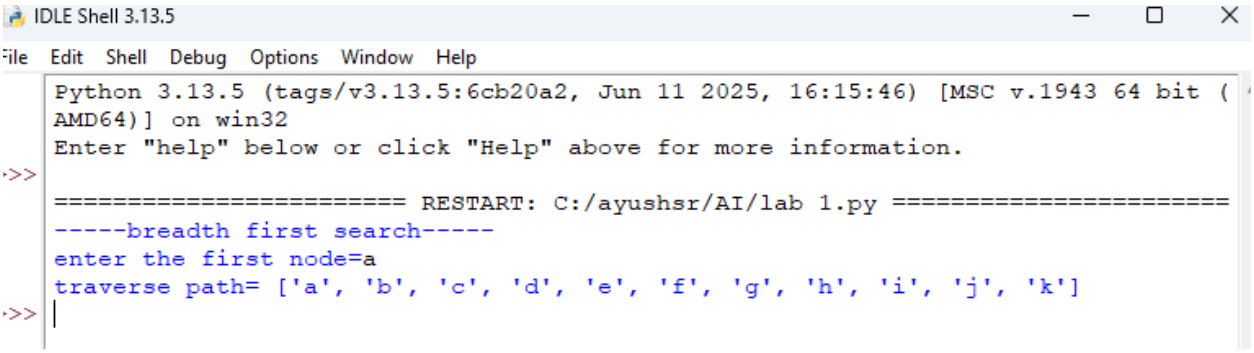
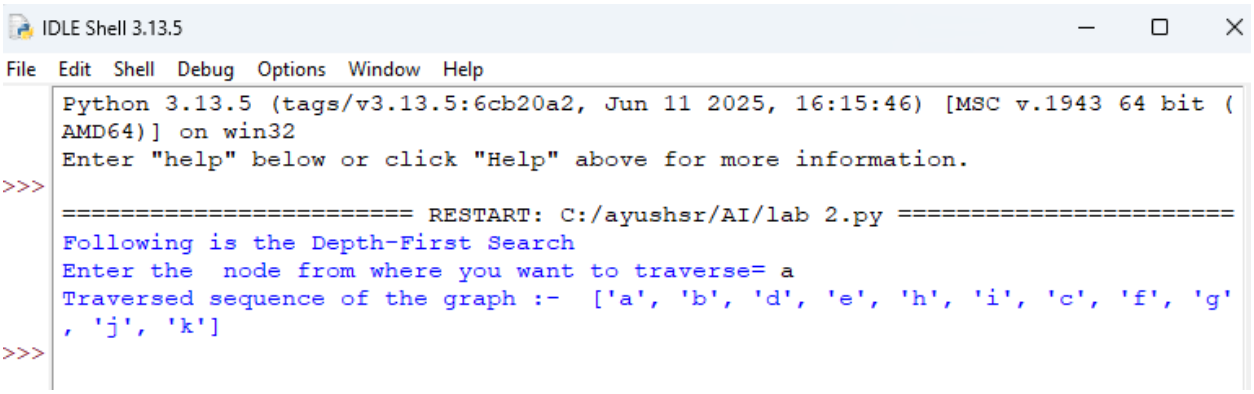


Bfs



Dfs

Bsf

```
import queue

adj_list={

   "a":["b","c","d"],

   "b":["e","f"],

   "c":["g"],

   "d":["h"],

   "e":[],

   "f":["i"],

   "g":["j"],

   "h":["k"],

   "i":[],
```

```
    "j":[],
    "k":[],
    }

output=[]
visited=[]
queue=[]

def bfsdemo(visited,graph,node):
    visited.append(node)
    queue.append(node)

    while queue:
        m=queue.pop(0)
        output.append(m)

        for neighbour in graph[m]:
            if neighbour not in visited:
                visited.append(neighbour)
                queue.append(neighbour)

    print("traverse path=",output)
print("-----breadth first search-----")
startnode=input("enter the first node=")
bfsdemo(visited,adj_list,startnode)
```

```
dfs
adj_list = {
   "a":["b","c"],
   "b":["d","e"],
   "c":["f","g"],
   "d":[],
   "e":["h","i"],
   "f":[],
   "g":["j","k"],
   "h":[],
   "i":[],
   "j":[],
   "k":[],
   }

closedlist={}
dfs_traversal_output=[]



for node in adj_list.keys():
   closedlist[node]="notvisited"



def dfs_util(u):
   closedlist[u]="visited"
   dfs_traversal_output.append(u)

   for v in adj_list[u]:
      if closedlist[v]!="visited":
```

```
        dfs_util(v)



print("Following is the Depth-First Search")
startnode=input("Enter the  node from where you want to traverse= ")
dfs_util(startnode)

print("Traversed sequence of the graph :- ",dfs_traversal_output)
```