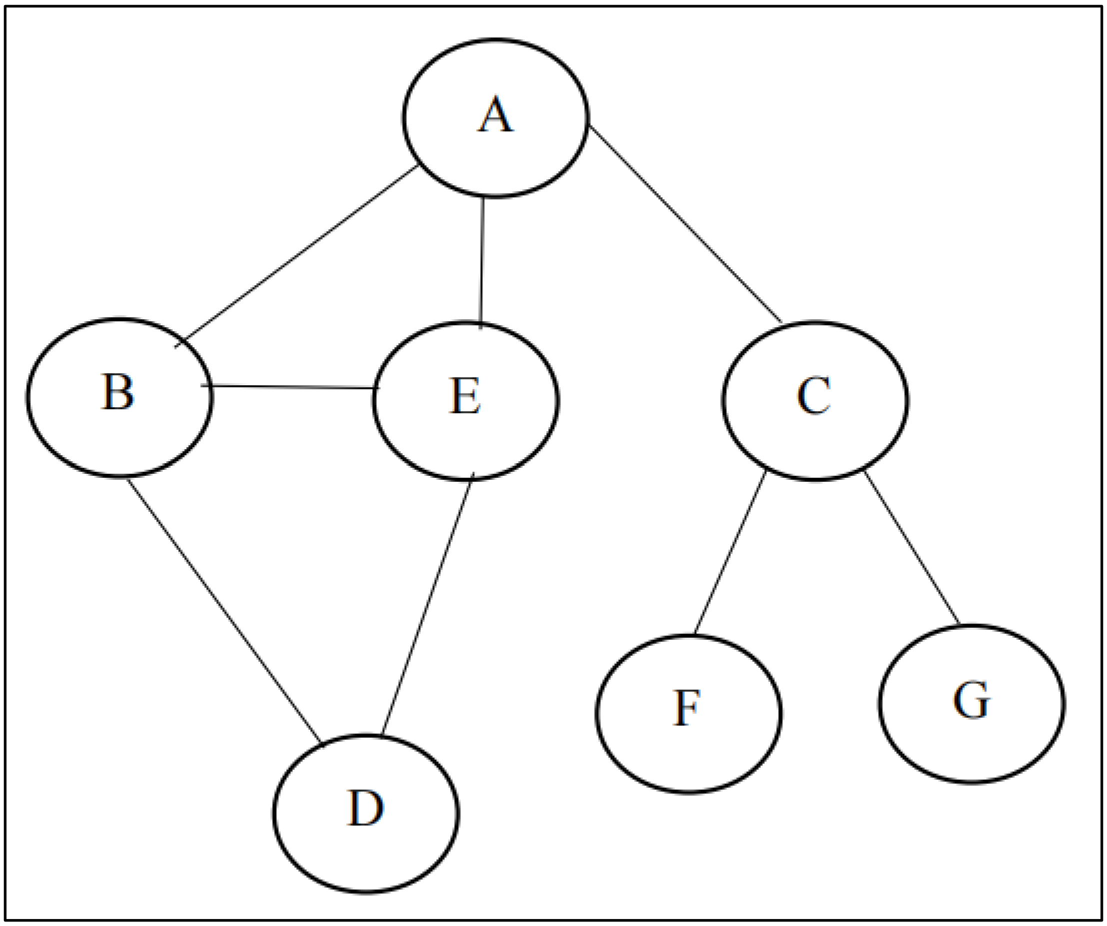
LAB # 12
IMPLEMENTING UNINFORMED SEARCH TECHNIQUIES

OBJECTIVE
Finding shortest path using BFS and DFS search technique in python

Lab Tasks:
1. Apply Breadth First Search on following graph considering the initial state is A and final state is G. Show results in form of open and closed list.



Source Code:

Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102

```
import queue
BFS = {'A':['B', 'C','E'],
        'B':['A', 'E', 'D'],
        'C':['F', 'G'],
        'D': ['B','E'],
        'E':['B','D'],
        'F': [],
        'G': []}
stack = queue.LifoQueue()
initial ='A'
goal ='G'
stack.put(initial)
closedlist = []
openlist= [goal]
print("Open List\t\tClose List")

while not stack.empty():
    node = stack.get()
    openlist.pop(0)
    if node==goal:
        print(openlist, end="\t\t")
        closedlist.append(node)
        print(closedlist)
        break
    items =graph[node]
    items.reverse()
    for item in items:
        if openlist.count(item) <=0 and closedlist.count(item) <=0:
            stack.put(item)
            openlist.insert(0, item)
    closedlist.append(node)
    print(openlist, end="\t\t")
    print(closedlist)
```

Output:

```
Open List                   Close List
['E', 'C', 'B']             ['A']
['D', 'C', 'B']             ['A', 'E']
['C', 'B']                  ['A', 'E', 'D']
['G', 'F', 'B']             ['A', 'E', 'D', 'C']
['F', 'B']                  ['A', 'E', 'D', 'C', 'G']
```


Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102

2. Using Question no. 1 apply BFS by taking initial and final state as user input. Show results in form of open and closed list.

Source Code:

```
import queue
BFS = {'A':['B', 'C','E'],
        'B':['A', 'E', 'D'],
        'C':['F', 'G'],
        'D': ['B','E'],
        'E':['B','D'],
        'F': [],
        'G': []}
stack = queue.LifoQueue()
initial ='A'
goal ='G'
stack.put(initial)
closedlist = []
initial=input("Enter your Initial State: ")
goal=input("Enter your Goal State: ")
openlist= [goal]
print("Open List\t\tClose List")
while not stack.empty():
    node = stack.get()
    openlist.pop(0)
    if node==goal:
        print(openlist, end="\t\t")
        closedlist.append(node)
        print(closedlist)
        break
    items =graph[node]
    items.reverse()
    for item in items:
        if openlist.count(item) <=0 and closedlist.count(item) <=0:
            stack.put(item)
            openlist.insert(0, item)
    closedlist.append(node)
    print(openlist, end="\t\t")
    print(closedlist)
```

Output:

Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102

```
Enter your Initial State: A
Enter your Goal State: G
Open List                Close List
['E', 'C', 'B']          ['A']
['D', 'C', 'B']          ['A', 'E']
['C', 'B']               ['A', 'E', 'D']
['G', 'F', 'B']          ['A', 'E', 'D', 'C']
['F', 'B']               ['A', 'E', 'D', 'C', 'G']
```

3. Apply Depth First Search on the graph given in question 1. Considering the initial state is A and final state is G. Show results in form of open and closed list.

Source Code:

```
import queue
DFS = {'A':['B', 'C','E'],
       'B':['A', 'E', 'D'],
       'C':['F', 'G'],
       'D': ['B','E'],
       'E':['B','D'],
       'F': [],
       'G': []}
stack = queue.LifoQueue()
initial ='A'
goal ='G'
stack.put(initial)
closedlist = []
openlist= [goal]
print("Open List\t\tClose List")

while not stack.empty():
   node = stack.get()
   openlist.pop(0)
   if node==goal:
      print(openlist, end="\t\t")
      closedlist.append(node)
      print(closedlist)
      break
   items =graph[node]
   items.reverse()
```


Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102

```
    for item in items:
    if openlist.count(item) <=0 and closedlist.count(item) <=0:
        stack.put(item)
        openlist.insert(0, item)
closedlist.append(node)
print(openlist, end="\t\t")
print(closedlist)
```

Output:

```
Open List                    Close List
['B', 'C', 'E']              ['A']
['D', 'C', 'E']              ['A', 'B']
['C', 'E']                   ['A', 'B', 'D']
['F', 'G', 'E']              ['A', 'B', 'D', 'C']
['G', 'E']                   ['A', 'B', 'D', 'C', 'F']
['E']           ['A', 'B', 'D', 'C', 'F', 'G']
```

4. Using Question no. 1 apply DFS by taking initial and final state as user input. Show results in form of open and closed list.

Source Code:

```
import queue
DFS = {'A':['B', 'C','E'],
      'B':['A', 'E', 'D'],
      'C':['F', 'G'],
      'D': ['B','E'],
      'E':['B','D'],
      'F': [],
      'G': []}
stack = queue.LifoQueue()
initial ='A'
goal ='G'
stack.put(initial)
closedlist = []
initial=input("Enter your Initial State: ")
goal=input("Enter your Goal State: ")
openlist= [goal]
print("Open List\t\tClose List")
```


Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102

```
while not stack.empty():
    node = stack.get()
    openlist.pop(0)
    if node==goal:
        print(openlist, end="\t\t")
        closedlist.append(node)
        print(closedlist)
        break
    items =graph[node]
    items.reverse()
    for item in items:
        if openlist.count(item) <=0 and closedlist.count(item) <=0:
            stack.put(item)
            openlist.insert(0, item)
    closedlist.append(node)
    print(openlist, end="\t\t")
    print(closedlist)
```

Output:

```
Enter your Initial State: A
Enter your Goal State: G
Open List                Close List
['B', 'C', 'E']          ['A']
['D', 'C', 'E']          ['A', 'B']
['C', 'E']               ['A', 'B', 'D']
['F', 'G', 'E']          ['A', 'B', 'D', 'C']
['G', 'E']               ['A', 'B', 'D', 'C', 'F']
['E']           ['A', 'B', 'D', 'C', 'F', 'G']
```


Name: Sheeza Imtiaz
Roll no: 2020F-BSE-102