```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT

C:\Users\default.LAPTOP-AR8V7UIC\Desktop\Chandan\Programming\Lab_Work_3rd_Sem\CE2019_Lab4_30_31\GraphDSA>g++ -std=c++11 main.cpp

C:\Users\default.LAPTOP-AR8V7UIC\Desktop\Chandan\Programming\Lab_Work_3rd_Sem\CE2019_Lab4_30_31\GraphDSA>a.exe

Select Choice to perform Operation on Graph!
0. Get Operation List!
1. Check Graph Status?
2. Check is Graph Directed?
3. Add Vertex!
4. Add Edge!
5. Remove Edge!
6. Remove Vertex!
7. Show Number of Vertices!
8. Show Number of Edge!
9. Show Degree of Vertex!
10. Show Indegree of Vertex!
11. Show Outdegree of Vertex!
12. Show Neighbour of Vertex
13. Check if Vertex1 is Neighbour to Vertex2!
14. Display Adjacency Matrix!
15. Generate Graph Corresponding to Adjacency Matrix!
16. Generate Random Graph!
17. Exist

Enter Choice: 1
Graph Is Empty!

Enter Choice: 7
Num of Vertices = 4
```



```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT

Enter Choice: 14
Graph is Empty!

Enter Choice: 4
Enter Vertex1: 1
Enter Vertex2: 2
Adding Edge Between A and B...
Edge Added Between A and B!

Enter Choice: 4
Enter Vertex1: 1
Enter Vertex2: 4
Adding Edge Between A and D...
Edge Added Between A and D!

Enter Choice: 4
Enter Vertex1: 2
Enter Vertex2: 3
Adding Edge Between B and C...
Edge Added Between B and C!

Enter Choice: 4
Enter Vertex1: 2
Enter Vertex2: 5
Adding Edge Between B and E...
Vertex does not exist!
Enter Choice: 14
Displaying Adjacency Matrix...
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT

Adjacency Matrix:
   A B C D
   _ _ _ _
A | 0 1 0 1
B | 0 0 1 0
C | 0 0 0 0
D | 0 0 0 0

Enter Choice: 15
Generating Graph Corresponding To Adjacency Matrix...
Number of Vertices = 4
Number of Edge = 3
Generated Graph is:
A-> { B D }
B-> { C }
C-> { Isolated Vertex }
D-> { Isolated Vertex }

Enter Choice: 7
Num of Vertices = 4

Enter Choice: 3
Adding Vertex...
Vertex Added!

Enter Choice: 4
Enter Vertex1: 2
Enter Vertex2: 5
Adding Edge Between B and E...
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT

```
Adding Edge Between B and E...
Edge Added Between B and E!

Enter Choice: 4
Enter Vertex1: 3
Enter Vertex2: 1
Adding Edge Between C and A...
Edge Added Between C and A!

Enter Choice: 4
Enter Vertex1: 4
Enter Vertex2: 3
Adding Edge Between D and C...
Edge Added Between D and C!

Enter Choice: 4
Enter Vertex1: 4
Enter Vertex2: 6
Adding Edge Between D and F...
Vertex does not exist!
Enter Choice: 14
Displaying Adjacency Matrix...

Adjacency Matrix:
   A B C D E
   _ _ _ _ _
A | 0 1 0 1 0
B | 0 0 1 0 1
C | 1 0 0 0 0
D | 0 0 1 0 0
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT

```
Enter Choice: 15
Generating Graph Corresponding To Adjacency Matrix...
Number of Vertices = 5
Number of Edge = 6
Generated Graph is:
A-> { B D }
B-> { C E }
C-> { A }
D-> { C }
E-> { Isolated Vertex }

Enter Choice: 7
Num of Vertices = 5

Enter Choice: 3
Adding Vertex...
Vertex Added!

Enter Choice: 4
Enter Vertex1: 4
Enter Vertex2: 6
Adding Edge Between D and F...
Edge Added Between D and F!

Enter Choice: 5
Enter Vertex1: 5
Enter Vertex2: 3
Edge Does Not Exist!
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT

```
Enter Choice: 4
Enter Vertex1: 5
Enter Vertex2: 3
Adding Edge Between E and C...
Edge Added Between E and C!

Enter Choice: 4
Enter Vertex1: 5
Enter Vertex2: 6
Adding Edge Between E and F...
Edge Added Between E and F!

Enter Choice: 4
Enter Vertex1: 6
Enter Vertex2: 3
Adding Edge Between F and C...
Edge Added Between F and C!

Enter Choice: 7
Num of Vertices = 6

Enter Choice: 14
Displaying Adjacency Matrix...

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 1 0 1 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT cmd

```
Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 1 0 1 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0

Enter Choice: 15
Generating Graph Corresponding To Adjacency Matrix...
Number of Vertices = 6
Number of Edge = 10
Generated Graph is:
A-> { B D }
B-> { C E }
C-> { A }
D-> { C F }
E-> { C F }
F-> { C }

Enter Choice: 0
0. Get Operation List!
1. Check Graph Status?
2. Check is Graph Directed?
3. Add Vertex!
4. Add Edge!
5. Remove Edge!
6. Remove Vertex!
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT cmd

```
7. Show Number of Vertices!
8. Show Number of Edge!
9. Show Degree of Vertex!
10. Show Indegree of Vertex!
11. Show Outdegree of Vertex!
12. Show Neighbour of Vertex
13. Check if Vertex1 is Neighbour to Vertex2!
14. Display Adjacency Matrix!
15. Generate Graph Corresponding to Adjacency Matrix!
16. Generate Random Graph!
17. Exist

Enter Choice: 8
Num of Edge = 10
Enter Choice: 9
Enter Vertex: 3
Degree of Vertex C is 5

Enter Choice: 10
Enter Vertex: 3
Indegree of Vertex C is 4

Enter Choice: 10
Enter Vertex: 3
Indegree of Vertex C is 4

Enter Choice: 11
Enter Vertex: 3
Outdegree of Vertex C is 1
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT cmd

```
Enter Choice: 12
Enter Vertex: 3
Neighbour/s of Vertex C is/are { A B D E F }


Enter Choice: 12
Enter Vertex: 6
Neighbour/s of Vertex F is/are { C D E }


Enter Choice: 13
Enter Vertex1: 5
Enter Vertex2: 1
Checking Relation Between E and A...
E is Not Neighbour to A

Enter Choice: 13
Enter Vertex1: 5
Enter Vertex2: 3
Checking Relation Between E and C...
E is Neighbour to C

Enter Choice: 13
Enter Vertex1: 5
Enter Vertex2: 6
Checking Relation Between E and F...
E is Neighbour to F

Enter Choice: 0
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

Enter Choice: 0
0. Get Operation List!
1. Check Graph Status?
2. Check is Graph Directed?
3. Add Vertex!
4. Add Edge!
5. Remove Edge!
6. Remove Vertex!
7. Show Number of Vertices!
8. Show Number of Edge!
9. Show Degree of Vertex!
10. Show Indegree of Vertex!
11. Show Outdegree of Vertex!
12. Show Neighbour of Vertex
13. Check if Vertex1 is Neighbour to Vertex2!
14. Display Adjacency Matrix!
15. Generate Graph Corresponding to Adjacency Matrix!
16. Generate Random Graph!
17. Exist

Enter Choice: 6
Enter Vertex To Be Removed: 3
Removing Vertex C...

First Remvoe Edge Between vertex C and A, Isolate Vertex C To Remove

First Remvoe Edge Between vertex C and B, Isolate Vertex C To Remove

First Remvoe Edge Between vertex C and D, Isolate Vertex C To Remove
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

First Remvoe Edge Between vertex C and E, Isolate Vertex C To Remove

First Remvoe Edge Between vertex C and F, Isolate Vertex C To Remove

Enter Choice: 6
Enter Vertex To Be Removed: 1
Removing Vertex A...

First Remvoe Edge Between vertex A and B, Isolate Vertex A To Remove

First Remvoe Edge Between vertex A and C, Isolate Vertex A To Remove

First Remvoe Edge Between vertex A and D, Isolate Vertex A To Remove

Enter Choice: 5
Enter Vertex1: 1
Enter Vertex2: 2
Before Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 1 0 1 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0
Removing Edge Between A and B...
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

Removing Edge Between A and B...
Edge Removed Between A and B!
After Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 0 0 1 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0

Enter Choice: 5
Enter Vertex1: 1
Enter Vertex2: 4
Before Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 0 0 1 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0
Removing Edge Between A and D...
Edge Removed Between A and D!
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

Edge Removed Between A and D!
After Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 0 0 0 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0

Enter Choice: 5
Enter Vertex1: 3
Enter Vertex2: 1
Before Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 0 0 0 0 0
B | 0 0 1 0 1 0
C | 1 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0
Removing Edge Between C and A...
Edge Removed Between C and A!
After Removal:
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

After Removal:

Adjacency Matrix:
   A B C D E F
   _ _ _ _ _ _
A | 0 0 0 0 0 0
B | 0 0 1 0 1 0
C | 0 0 0 0 0 0
D | 0 0 1 0 0 1
E | 0 0 1 0 0 1
F | 0 0 1 0 0 0

Enter Choice: 6
Enter Vertex To Be Removed: 1
Removing Vertex A...
Before Removal:

Adjacency Matrix:
   A B C D E
   _ _ _ _ _
A | 0 1 0 1 0
B | 0 0 0 0 0
C | 0 1 0 0 1
D | 0 1 0 0 1
E | 0 1 0 0 0
Vertex A Removed!
Name of the Vertex are changed According to the removal of vertex
After Removal:

Adjacency Matrix:
```

```
TERMINAL  PROBLEMS  DEBUG CONSOLE  OUTPUT                                  cmd

Adjacency Matrix:
   A B C D E
   _ _ _ _ _
A | 0 1 0 1 0
B | 0 0 0 0 0
C | 0 1 0 0 1
D | 0 1 0 0 1
E | 0 1 0 0 0

Enter Choice: 0
0. Get Operation List!
1. Check Graph Status?
2. Check is Graph Directed?
3. Add Vertex!
4. Add Edge!
5. Remove Edge!
6. Remove Vertex!
7. Show Number of Vertices!
8. Show Number of Edge!
9. Show Degree of Vertex!
10. Show Indegree of Vertex!
11. Show Outdegree of Vertex!
12. Show Neighbour of Vertex
13. Check if Vertex1 is Neighbour to Vertex2!
14. Display Adjacency Matrix!
15. Generate Graph Corresponding to Adjacency Matrix!
16. Generate Random Graph!
17. Exist
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT

```
Generating Graph Corresponding To Adjacency Matrix...
Number of Vertices = 5
Number of Edge = 7
Generated Graph is:
A-> { B D }
B-> { Isolated Vertex }
C-> { B E }
D-> { B E }
E-> { B }

Enter Choice: 1
Graph Is Not Empty!

Enter Choice: 2
Graph Is Directed!

Enter Choice: 16
Random Graph Generation:
This is An Undirected Random Generated Graph...
Number Of Vertex = 4
Number Of Edge = 2

Adjacency Matrix:
    A B C D
    _ _ _ _
A | 0 1 0 0
B | 1 0 1 0
C | 0 1 0 0
D | 0 0 0 0
Graph Visualization!
```

```
Enter Choice: 16
Random Graph Generation:
This is An Undirected Random Generated Graph...
Number Of Vertex = 4
Number Of Edge = 2

Adjacency Matrix:
    A B C D
    _ _ _ _
A | 0 1 0 0
B | 1 0 1 0
C | 0 1 0 0
D | 0 0 0 0
Graph Visualization!
A-> { B }
B-> { A C }
C-> { B }
D-> { Isolated Vertex }
Programme Terminated!

Enter Choice: 16
Random Graph Generation:
This is An Undirected Random Generated Graph...
Number Of Vertex = 24
Number Of Edge = 88

Adjacency Matrix:
    A B C D E F G H I J K L M N O P Q R S T U V W X
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT

```
Adjacency Matrix:
    A B C D E F G H I J K L M N O P Q R S T U V W X
    _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
A | 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 1
B | 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1
C | 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 1 0
D | 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0
E | 1 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
F | 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 1 0 1 0 0 0 1
G | 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0
H | 0 1 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0
I | 0 0 0 0 1 0 0 0 0 1 0 1 0 0 1 0 1 0 1 1 0 0 0 0
J | 0 1 0 0 0 0 0 0 1 0 1 1 0 0 1 1 1 0 1 0 0 1 0 1
K | 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0
L | 0 0 1 1 0 1 0 0 1 1 0 0 1 0 0 1 0 0 1 1 0 0 0 0
M | 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1
N | 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0
O | 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 1 0 1 1
P | 0 0 0 0 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0
Q | 0 0 1 1 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0
R | 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
S | 0 0 0 0 0 1 0 0 1 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0
T | 1 0 1 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1
U | 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0
V | 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0
W | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0
X | 1 1 0 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0
Graph Visualization!
A-> { C E N T U X }
```

TERMINAL PROBLEMS DEBUG CONSOLE OUTPUT cmd

```
T | 1 0 1 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1
U | 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0
V | 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0
W | 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0
X | 1 1 0 0 0 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0
Graph Visualization!
A-> { C E N T U X }
B-> { H J K X }
C-> { A E L Q T W }
D-> { H K L Q }
E-> { A C I O }
F-> { K L M P Q S X }
G-> { H N R U }
H-> { B D G O P R U }
I-> { E J L O Q S T }
J-> { B I K L O P Q S V X }
K-> { B D F J P }
L-> { C D F I J M P S T }
M-> { F L N X }
N-> { A G M S }
O-> { E H I J T U W X }
P-> { F H J K L U V }
Q-> { C D F I J U }
R-> { G H S }
S-> { F I J L N R }
T-> { A C I L O W X }
U-> { A G H O P Q }
V-> { J P W }
W-> { C O T V }
X-> { A B F J M O T }
```

```
Enter Choice: 0
0. Get Operation List!
1. Check Graph Status?
2. Check is Graph Directed?
3. Add Vertex!
4. Add Edge!
5. Remove Edge!
6. Remove Vertex!
7. Show Number of Vertices!
8. Show Number of Edge!
9. Show Degree of Vertex!
10. Show Indegree of Vertex!
11. Show Outdegree of Vertex!
12. Show Neighbour of Vertex
13. Check if Vertex1 is Neighbour to Vertex2!
14. Display Adjacency Matrix!
15. Generate Graph Corresponding to Adjacency Matrix!
16. Generate Random Graph!
17. Exist

Enter Choice: 17
Programme Terminated!

C:\Users\default.LAPTOP-AR8V7UIC\Desktop\Chandan\Programming\Lab_Work_3rd_Sem\CE2019_Lab4_30_31\GraphDSA>
```