```
 1:  /*----Graph-----*/
 2:
 3:  #include<iostream>
 4:  using namespace std;
 5:
 6:  class graph
 7:  {
 8:     private:
 9:         int size, edge, graph[20][20];
10:     public:
11:         void info()
12:         {
13:             cout << "Enter size: \n";
14:             cin >> size;
15:             cout << "Enter edges: \n";
16:             cin >> edge;
17:             for(int i = 0; i < size; i++)
18:             {
19:                 for(int j = 0; j < size; j++)
20:                 {
21:                     graph[i][j] = 0;
22:                  }
23:              }
24:
25:             for(int k = 0; k < edge; k++)
26:             {
27:                 int l, m;
28:                 cout << "Enter start: ";
29:                 cin >> l;
30:                 cout << "Enter end: ";
31:                 cin >> m;
32:                 graph[l - 1][m - 1] = 1;
33:             }
34:          }
35:
36:         void print()
37:         {
38:             for(int i = 0; i < size; i++)
39:             {
40:                 for(int j = 0; j < size; j++)
41:                 {
42:                     cout << graph[i][j] << " ";
43:                  }
44:                 cout << endl;
45:              }
46:         }
47:
48:         void depthTrav(int k)
49:         {
50:             int stack[20], printstatus[size], index = 0;
```

```
51:             stack[index] = k;
52:             for(int i = 0; i < size; i++)
53:             {
54:                 printstatus[i] = 0;
55:             }
56:
57:             while(index >= 0)
58:             {
59:                 if(printstatus[stack[index] - 1] == 0)
60:                 {
61:                     cout << stack[index] << endl;
62:                     printstatus[stack[index] - 1] = 1;
63:                     index--;
64:                     for(int j = size - 1; j >= 0; j--)
65:                     {
66:                         if(graph[k - 1][j] == 1)
67:                         {
68:                             stack[index + 1] = j + 1;
69:                             index++;
70:                         }
71:                     }
72:                     k = stack[index];
73:                 }
74:                 else
75:                 {
76:                     index--;
77:                 }
78:             }
79:         }
80:
81:         void breadthTrav(int k) // Not completed
82:         {
83:             int queue[20], printstatus[size], index1 = 0, index2 = 0;
84:             queue[index1] = k;
85:             for(int i = 0; i < size; i++)
86:             {
87:                 printstatus[i] = 0;
88:             }
89:
90:             int i = 0;
91:             while(i < size)
92:             {
93:                 if(printstatus[queue[index2] - 1] == 0)
94:                 {
95:                     cout << queue[index2] << endl;
96:                     printstatus[queue[index2] - 1] = 1;
97:                     index2++;
98:                     for(int j = 0; j < size; j++)
99:                     {
100:                        if(graph[k - 1][j] == 1)
```

```
                        {
                            queue[index1 + 1] = j + 1;
                            index1++;
                        }
                    }
                    k = queue[index1];
                }
                i++;
            }
        }
};

int main()
{
    graph temp;
    temp.info();
    cout << endl;
    temp.print();
    cout << endl;
    temp.depthTrav(2);
    cout << endl;
    temp.breadthTrav(2);

    return 0;
}
```