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

```
51:             index++;
52:         }
53:         int takestack()
54:         {
55:             index--;
56:             return stack[index];
57:         }
58:         void depthTrav(int k)
59:         {
60:             putstack(k);
61:             while(index != 0)
62:             {
63:                 int j = takestack();
64:                 cout << j << " ";
65:                 for(int i = size; i > 0; i--)
66:                 {
67:                     if(graph[j][i] == 1)
68:                     {
69:                         putstack(i);
70:                     }
71:                 }
72:             }
73:
74:         }
75:
76:         int queue[20];
77:         int rear = 0;
78:         int front = -1;
79:         void queueput(int k)
80:         {
81:             queue[rear] = k;
82:             rear++;
83:         }
84:         int queuetake()
85:         {
86:             front++;
87:             return queue[front];
88:         }
89:         void breadthTrav(int k)
90:         {
91:             queueput(k);
92:             while(rear > front + 1)
93:             {
94:                 int j = queuetake();
95:                 cout << j << " ";
96:                 for(int i = 1; i <= size; i++)
97:                 {
98:                     if(graph[j][i] == 1)
99:                     {
100:                        queueput(i);
```

```
                    }
                }
            }
        }
};

int main()
{
    graph temp;
    temp.info();
    cout << endl;
    temp.print();
    cout << endl;
    temp.depthTrav(4);
    cout << endl;
    temp.breadthTrav(4);

    return 0;
}
```