```python
from queuelinked import LinkedQueue
import numpy as np
class Graph:
    def __init__(self,vertices):
        self._adjMat = np.zeros((vertices, vertices))
        self._vertices = vertices

    def insert_edge(self,u,v,w=1):
        self._adjMat[u][v] = w

    def delete_edge(self,u,v):
        self._adjMat[u][v] = 0

    def get_edge(self,u,v):
        return self._adjMat[u][v]

    def vertices_count(self):
        return self._vertices

    def edge_count(self):
        count = 0
        for i in range(self._vertices):
            for j in range(self._vertices):
                if not self._adjMat[i][j] == 0:
                    count += 1
        return count

    def indegree(self,u):
        count = 0;
        for i in range(self._vertices):
            if not self._adjMat[i][u] == 0:
                count += 1
        return count

    def outdegree(self,u):
        count = 0;
        for i in range(self._vertices):
            if not self._adjMat[u][i] == 0:
                count += 1
        return count

    def display(self):
        print(self._adjMat)

    def BFS(self,source):
        i = source
        q = LinkedQueue()
```

```
48          visited = [0] * self._vertices
49          print(i, end=' - ')
50          visited[i] = 1
51          q.enqueue(i)
52
53          while not q.is_empty():
54              i = q.dequeue()
55              for j in range(self._vertices):
56                  if self._adjMat[i][j] == 1 and visited[j]
    == 0:
57                      print(j, end=' - ')
58                      visited[j] = 1
59                      q.enqueue(j)
60
61 G = Graph(7)
62 G.insert_edge(0,1)
63 G.insert_edge(0,5)
64 G.insert_edge(0,6)
65 G.insert_edge(1,0)
66 G.insert_edge(1,2)
67 G.insert_edge(1,5)
68 G.insert_edge(1,6)
69 G.insert_edge(2,3)
70 G.insert_edge(2,4)
71 G.insert_edge(2,6)
72 G.insert_edge(3,4)
73 G.insert_edge(4,2)
74 G.insert_edge(4,5)
75 G.insert_edge(5,2)
76 G.insert_edge(5,3)
77 G.insert_edge(6,3)
78 G.BFS(0)
79
80
81
```