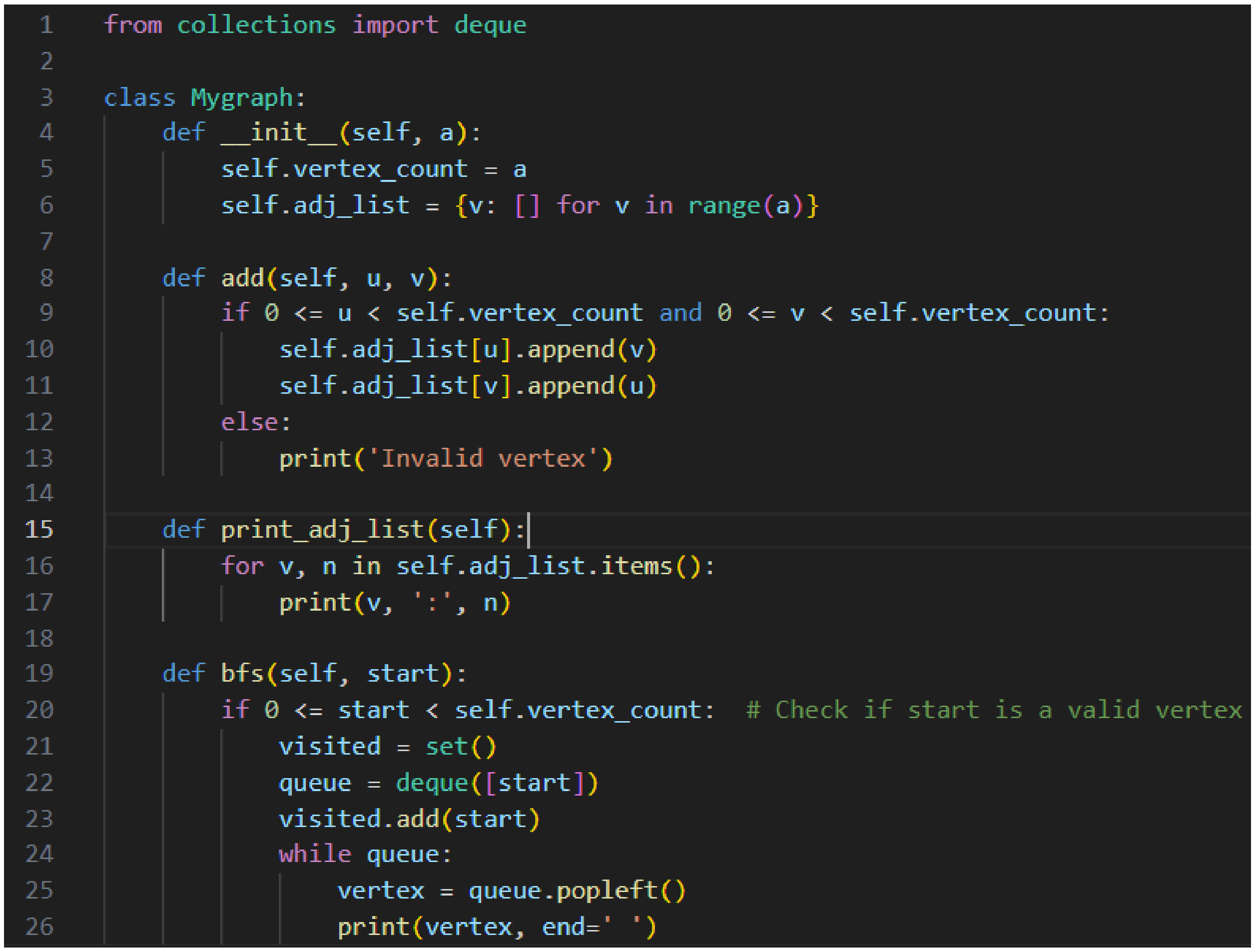
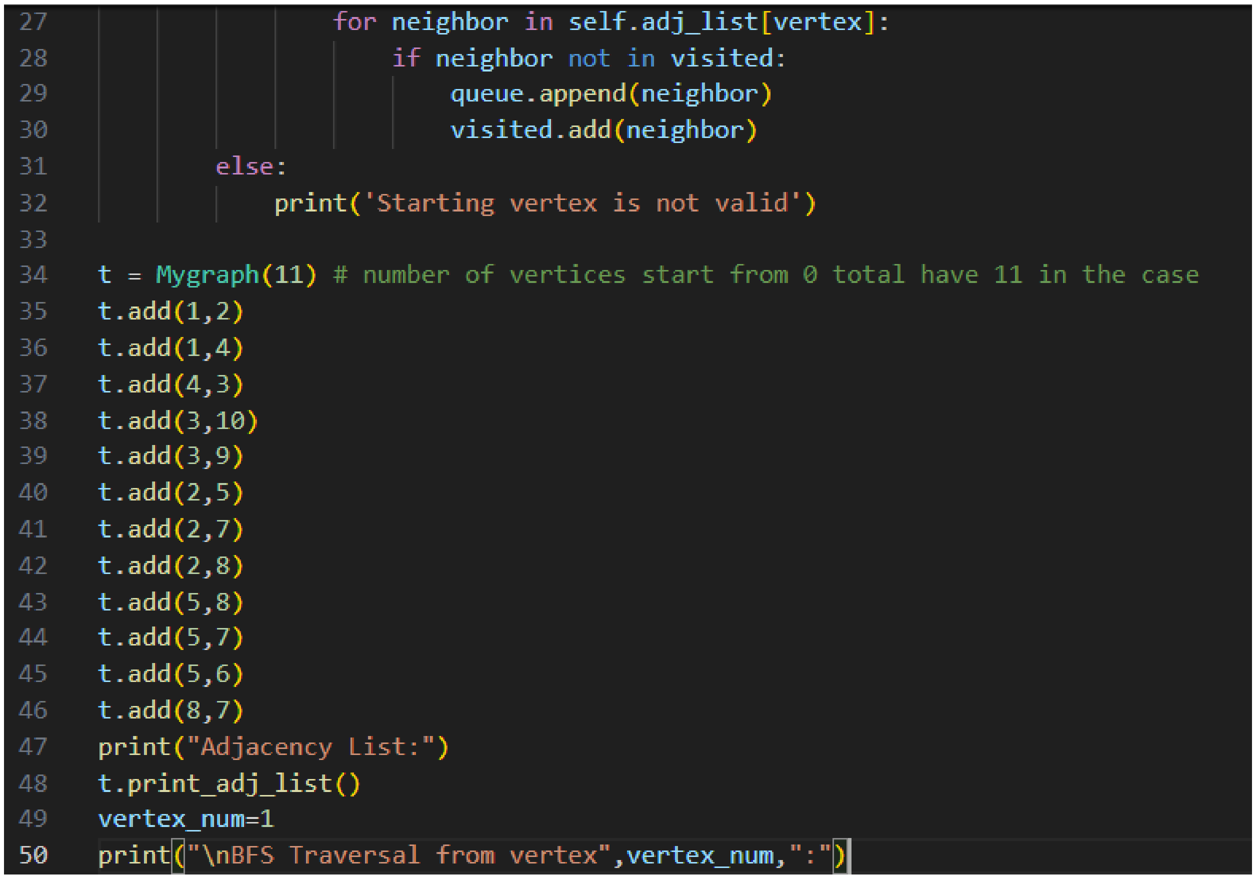
LAB#16

Example#1: Write a program to create a graph, then traverse it by using BFS(Breadth First Search).

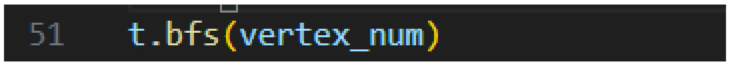
Solution:



```python
from collections import deque

class Mygraph:
    def __init__(self, a):
        self.vertex_count = a
        self.adj_list = {v: [] for v in range(a)}

    def add(self, u, v):
        if 0 <= u < self.vertex_count and 0 <= v < self.vertex_count:
            self.adj_list[u].append(v)
            self.adj_list[v].append(u)
        else:
            print('Invalid vertex')

    def print_adj_list(self):
        for v, n in self.adj_list.items():
            print(v, ':', n)

    def bfs(self, start):
        if 0 <= start < self.vertex_count:  # Check if start is a valid vertex
            visited = set()
            queue = deque([start])
            visited.add(start)
            while queue:
                vertex = queue.popleft()
                print(vertex, end=' ')
```

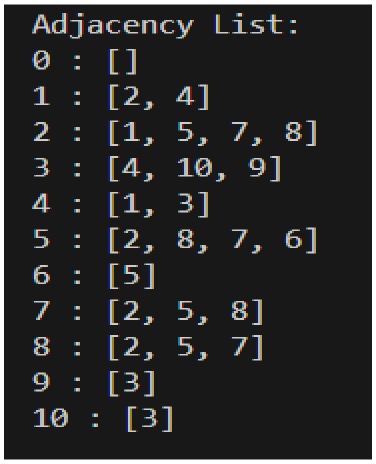


```python
                for neighbor in self.adj_list[vertex]:
                    if neighbor not in visited:
                        queue.append(neighbor)
                        visited.add(neighbor)
        else:
            print('Starting vertex is not valid')

t = Mygraph(11) # number of vertices start from 0 total have 11 in the case
t.add(1,2)
t.add(1,4)
t.add(4,3)
t.add(3,10)
t.add(3,9)
t.add(2,5)
t.add(2,7)
t.add(2,8)
t.add(5,8)
t.add(5,7)
t.add(5,6)
t.add(8,7)
print("Adjacency List:")
t.print_adj_list()
vertex_num=1
print("\nBFS Traversal from vertex",vertex_num,":")
```

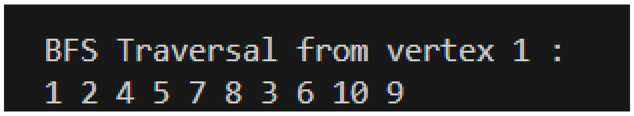


```python
t.bfs(vertex_num)
```

Result:

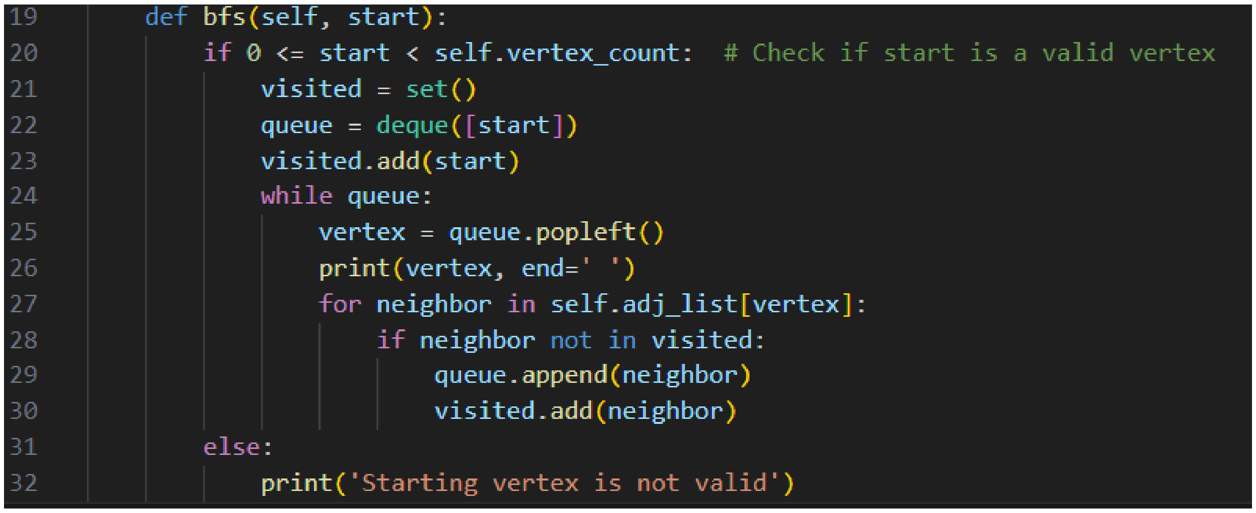


```
Adjacency List:
0 : []
1 : [2, 4]
2 : [1, 5, 7, 8]
3 : [4, 10, 9]
4 : [1, 3]
5 : [2, 8, 7, 6]
6 : [5]
7 : [2, 5, 8]
8 : [2, 5, 7]
9 : [3]
10 : [3]
```



```
BFS Traversal from vertex 1 :
1 2 4 5 7 8 3 6 10 9
```

Explanation of the code (from bfs function):



```python
    def bfs(self, start):
        if 0 <= start < self.vertex_count:  # Check if start is a valid vertex
            visited = set()
            queue = deque([start])
            visited.add(start)
            while queue:
                vertex = queue.popleft()
                print(vertex, end=' ')
                for neighbor in self.adj_list[vertex]:
                    if neighbor not in visited:
                        queue.append(neighbor)
                        visited.add(neighbor)
        else:
            print('Starting vertex is not valid')
```

```
def bfs(self,1)
        visited=()
        queue=1
        visited=1
```

```
1) while queue
    Vertex=1
    Print(1)
    For neighbor in list[1]
        Neighbor not in visited
            Queue.2
            Visited.2

            Queue.4
            Visited.4
```

BFS:1

Queue=1

visited(1)

List[1]->2,4

queue 2,4

visited(1,2,4)

```
(2) while queue
        vertex=queue.popleft()
        vertex=2
        print(2)
    for neighbor in list[2]
            q.5
            v.5
            q.7
            v.7
            q.8
            v.8
```

BFS: 1 2

visited(1,2,4,5,7,8)
Queue: 4 5 7 8

```
(3) while queue
        vertex= 4
        print( 4 )
    for neighbor in list[ 4 ]
                        ⇩
                      1,3
            queue.3
            visited.3
```

queue: 5 7 8
BFS:1 2 4
visited(1,2,4,5,7,8)

queue:5 7 8 3

v(1,2,4,5,7,8,3)

```
(4) while queue
        vertex=5
        print(5)
    for neighbor in list[5]
                        ⇩
                    2,6,7,8
                  queue.6
                  visited.6
```

queue: 7 8 3
BFS:1 2 4 5
visited(1,2,4,5,7,8,3)

queue: 7 8 3 6

```
(5) while queue
        vertex=7
        print(7)
    for neighbor in list[7]
                      ⇩
                    2,5,8

                  X
```

BFS:1 2 4 5 7
queue:8 3 6

visited(1,2,4,5,7,8,3,6)

```
(6)    while queue

            vertex=8
            print(8)
         for neighbor in list[8]
                           ⇩
                          2,5
                         X
```

BFS:1 2 4 5 7 8
queue: 3 6

visited(1,2,4,5,7,8,3,6)

```
(7)   while queue
          vertex=3
          print(3)
      for neighbor in list[3]
                q.10
                v.10
                q.9
                v.10
```

queue:6
BFS:1 2 4 5 7 8 3
visited(1,2,4,5,7,8,3 ,6,10,9)

queue:6,10,9

```
(8) while queue
        vertex=6
         print(6)
      for neighbor in list[6]
                          ⇩
                          5
                    X
```

queue:10,9

BFS: 1 2 4 5 7 8 3 6

visited(1,2,4,5,7,8,3,6,10,9)

```
(9) while queue
        vertex=10
        print(10)
    for neighbor in list [10]
                          ⇩
                   X      3
```

BFS: 1 2 4 5 7 8 3 6 10

```
(10) while queue
        vertex=9
         print(9)
       for neighbor in list[9]
                          ⇩
                   X
```

queue:9

BFS:1 2 4 5 7 8 3 6 10 9

visited(1,2,4,5,7,8,3,6,10,9)

queue:[]

Class Assignment

Q. Write a program to create a graph having 1 nodes (1 to 12) then traverse the nodes by using DFS method.

Note:

Explain the DFS function line by line in detail.