**TASK 1:**

**Implement Breadth First Search in python.**

**Code:**

def BFS(graph, start, path=[]):

    q = [start]

    while q:

        v = q.pop(0)

        if not v in path:

            path = path+[v]

            q = q+graph[v]

    return path

graph = {

    'A':['B','C','D'],

    'B':['A','E'],

    'C':['A','F','D'],

    'E':['B'],

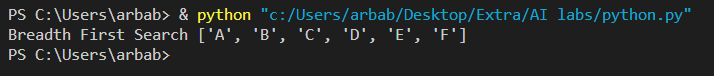
    'F':['C'],

    'D':['A','C']

}

print("Breadth First Search", BFS(graph,'A'))

**Output:**

****

**TASK 2:**

**Find shortest path between two nodes using BFS algorithm.**

**(Note: Please use comments to describe each line of code)**

**Code:**

def part\_short(graph,start,goal):

    explored=[]

    q=[start]

    if start==goal:

        return "start=goal"

    while q:

        path=q.pop(0)

        node=path[-1]

        neighbour=graph[node]

        for neighbour in neighbour:

            newpath=list(path)

            newpath.append(neighbour)

            q.append(newpath)

            if neighbour==goal:

                return newpath

        explored.append(node)

    return "sorry! connecting path does not exist"

graph={ 'A':['C','D','G'],

        'B':['C','D','E'],

        'C':['A','B','F'],

        'D':['A','B','G'],

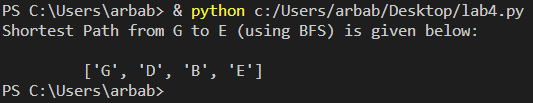
        'E':['B'],

        'F':['C'],

        'G':['A','D']}

print('Shortest Path from G to E (using BFS) is given below:\n\n\t',part\_short(graph,'G','E'))

**Output:**

****