PRACTICAL 3

Name : Varsha Valecha
Class: CSE A (3rd year)
Roll: 26
Batch: A 2

AIM : Write a program to implement Breadth First Search. Take a graph and start/goal node an input. Your job is to find goal node. **Print the total cost and path**

Code:

```
graph= {
    'Arad': [("Zerind",75),("Timisoara",118),("Sibiu",140)],
    'Bucharest' : [("Urziceni",85),("Giurgiu",90),("Pitesti",101),("Fag
aras",211)],
    'Craiova' :[("Dobreta",120),("Pitesti",138),("RV",146)],
    'Dobreta': [("Mehadia",75),("Craiova",120)],
    'Eforie' : [("Hirsova",86)],
    'Fagaras': [("Sibiu",99),("Bucharest",211)],
    'Giurgiu' : [("Bucharest",90)],
    'Hirsova' :[("Eforie",86),("Urziceni",98)],
    'Iasi' : [("Neamt",87),("Vaslui",142)],
    'Lugoj' : [("Mehadia",70),("Timisoara",111)],
    'Mehadia' : [("Lugoj",70),("Dobreta",75)],
    'Neamt' : [("Iasi",87)],
    'Oradea' : [("Zerind",71),("Sibiu",151)],
    'Pitesti' : [("RV",97),("Bucharest",101),("Craiova",138)],
    'RV' :[("Sibiu",80),("Pitesti",97),("Craiova",)],
    'Sibiu': [("RV",80),("Fagaras",99),("Oradea",151),("Arad",140)],
    'Timisoara' : [("Lugoj",111),("Arad",118)],
    'Urziceni' : [("Bucharest",85),("Hirsova",98),("Vaslui",142)],
    'Vaslui'  : [("Iasi",92),("Urziceni",142)],
    'Zerind': [("Oradea",71),("Arad",75)]
    }
que=[]

def path (arr,p):
  #  print(arr[p][0],end=" -> ")

   que.append(arr[p][0])
   p=arr[p][1]
   if(p!=-1):
     path(arr,p)
   exit()

def route():
  c=len(que)
  cost =0
```

```
  while(c>0):
     print(que[c-1])
     for n in graph[que[c-1]]:
      if n[0]==que[c-2]:
        cost=cost+n[1]
     c=c-1
  print("Total cost of path is : ")
  return cost

visited=[]
queue=[]


def bfs(visited,graph,start,end):
  arr=[]
  index=-1
  visited.append(start)
  queue.append(start)
  arr.append([start,index])

  while end not in visited:
    m=queue.pop(0)
    print (m, end = " ")
    index+=1

    for neighbour in graph[m]:
      if neighbour[0] not in visited:
        visited.append(neighbour[0])
        queue.append(neighbour[0])
        arr.append([neighbour[0],index])

  # print(arr)
  # print("\nPath from end to start is : ")
  path(arr,len(arr)-1)
  print("\nPath from start to end is : ")
  print(route())


print("Following is the Breadth-
First Search (only till goal node is reached): ")
bfs(visited, graph, 'Arad','Bucharest')
```

OUTPUT:

```
Following is the Breadth-First Search (only till goal node is reached):
Arad Zerind Timisoara Sibiu Oradea Lugoj RV Fagaras

Path from start to end is :
Arad
Sibiu
Fagaras
Bucharest

Total cost of path is :
450
```