Shri Ramdeobaba College of Engineering and Management, Nagpur
Department of Computer Science and Engineering
Session: 2022-2023

Practical 3 - Artificial Intelligence Lab

Name: Prathamesh Rajbhoj
Roll No: A - 53

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

Methods:

1. Using Alphabetical Order
   `graph[i].sort(key = lambda x: x[0])`

2. Using Minimum Path
   `graph[i].sort(key = lambda x: x[1])`

```
graph = {
    'Bangalore' : [('Hyderabad',1), ('Kolkata',6), ('Mumbai',3)],
    'Hyderabad' : [('Kolkata',2), ('Lucknow',4), ('Mumbai',2)],
    'Kolkata' : [('Bangalore',6), ('Hyderabad',2), ('Lucknow',3)],
    'Lucknow' : [('Hyderabad',4), ('Kolkata',3), ('New Delhi',2)],
    'Mumbai' : [('Bangalore',3), ('Hyderabad',2), ('New Delhi',5)],
    'New Delhi' : [('Lucknow',2), ('Mumbai',5)]
}


for i in graph:
  graph[i].sort(key = lambda x: x[1])
  print(i, " : ", graph[i])
```

```
Bangalore  :  [('Hyderabad', 1), ('Mumbai', 3), ('Kolkata', 6)]
Hyderabad  :  [('Kolkata', 2), ('Mumbai', 2), ('Lucknow', 4)]
Kolkata  :  [('Hyderabad', 2), ('Lucknow', 3), ('Bangalore', 6)]
Lucknow  :  [('New Delhi', 2), ('Kolkata', 3), ('Hyderabad', 4)]
Mumbai  :  [('Hyderabad', 2), ('Bangalore', 3), ('New Delhi', 5)]
New Delhi  :  [('Lucknow', 2), ('Mumbai', 5)]
```

```python
def bfs(graph):

  q = []
  vis = []
  parent = []

  q.append('New Delhi')
  vis.append('New Delhi')

  ancs = []

  while(q):
    sz = len(q)
    for loop in range(sz):
      node = q.pop(0)

      if(node=='Bangalore'):
        break

      for i in graph[node]:
        if(i[0] not in vis):
          pair = (i[0],node)
          parent.append(pair)
          q.append(i[0])
          vis.append(i[0])

  return parent

parent = bfs(graph)

ans = ['Bangalore']
child = 'Bangalore'
```

```python
while(child != 'New Delhi'):

  par = -1


  for i in parent:
    if(i[0]==child):
      par = i[1]
      ans.append(par)
      child = par

ans.reverse()
print('Path found from BFS Traversal is : ', ans)
```

```python
pathSize = len(ans)
cost = 0

for i in range(1,pathSize):

  source = ans[i-1]
  destination = ans[i]

  for j in graph[source]:
    if(j[0]==destination):
      cost = cost + j[1]

print('Path found from BFS Traversal is : ', ans)
print('Total Cost is : ', cost)
```

```
Path found from BFS Traversal is :
['New Delhi', 'Mumbai', 'Bangalore']

Total Cost is :  8
```