AI LAB EXP – 5a

BFS ALGORITHM FOR REAL WORLD PROBLEMS

Tejas Ashok

RA1911030010090

AIM

To implement Best First Algorithm using python.

ALGORITHM

- Define a list, OPEN, consisting solely of a single node, the start node, s.
- IF the list is empty, return failure.
- Remove from the list the node n with the best score (the node where f is the minimum), and move it to a list, CLOSED.
- Expand node n.
- IF any successor to n is the goal node, return success and the solution (by tracing the path from the goal node to s).
- FOR each successor node:
    1. Apply the evaluation function, f, to the node.
    2. IF the node has not been in either list, add it to OPEN.
- Looping structure by sending the algorithm back to the second step.

CODE

```
from queue import PriorityQueue

v = 14

graph = [[] for i in range(v)]



def best_first_search(source, target, n):

    visited = [0] * n

    visited[0] = True

    pq = PriorityQueue()

    pq.put((0, source))
```

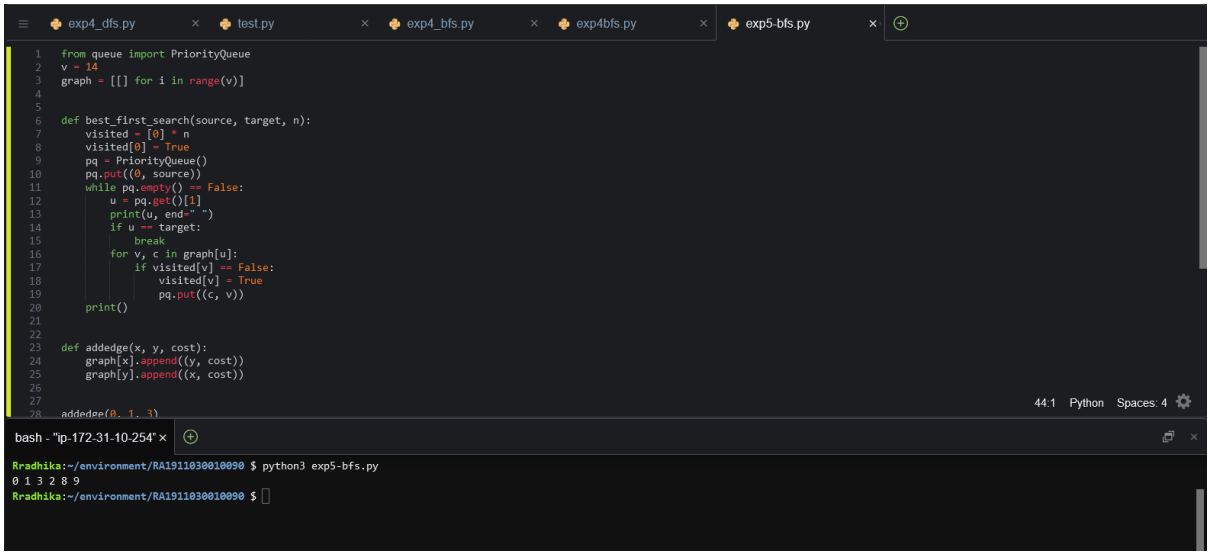
```
    while pq.empty() == False:
        u = pq.get()[1]
        print(u, end=" ")
        if u == target:
            break
        for v, c in graph[u]:
            if visited[v] == False:
                visited[v] = True
                pq.put((c, v))
    print()


def addedge(x, y, cost):
    graph[x].append((y, cost))
    graph[y].append((x, cost))


addedge(0, 1, 3)
addedge(0, 2, 6)
addedge(0, 3, 5)
addedge(1, 4, 9)
addedge(1, 5, 8)
addedge(2, 6, 12)
addedge(2, 7, 14)
addedge(3, 8, 7)
addedge(8, 9, 5)
addedge(8, 10, 6)
addedge(9, 11, 1)
addedge(9, 12, 10)
addedge(9, 13, 2)
source = 0
target = 9
```

```
best_first_search(source, target, v)
```

OUTPUT



RESULT

Best first search algorithm was successfully executed in python.