Name: Ariyan Hossain
ID: 20101099
Sec: 09
Course: CSE 221

Lab-03

Answer-4

BFS

For Adjacency list,

```
visited = [0] * d  — O(V)
queue = [] — O(1)
def BFS(visited, graph, node, endPoint): -
    visited[int(node)-1] = 1 — O(1)
    queue.append(node) — O(1)
    while queue: — O(V)
        m = queue.pop(0) — O(1)
        print(m, end = " ") — O(1)
        if m = endPoint: — O(1)
            break — O(1)
        for i in graph[m]: — O(Eaj)
            if visited[int(i)-1] == 0: — O(1)
                visited[int(i)-1] = 1 — O(1)
                queue.append(i) — O(1)
```

The while loop will run V times, where V is the total number of vertices in graph.

The for loop will run Eaj times, where Eaj is number of adjacent edges to current vertex

Time Complexity:

$V + 1 + V*(1 + E_{aj})$

$= 1 + 2V + V*E_{aj}$

$= V + E$ where E is the total number of edges in the graph

$\therefore O(V+E)$

For Adjacency Matrix,

The while loop will run V times, where V is the total number of vertices in the graph.

The for loop will run V times

Time Complexity:

$1 + V*(1+V)$

$= 1 + V + V^2$

$\therefore O(V^2)$

DFS

For Adjacency list,

```
visited = [0] * d  — O(V)
printed = [ ]  — O(1)
def DFS_VISIT (graph, node):
    visited [int(node) - 1] = 1  — O(1)
    printed.append (node)  — O(1)
    for node in graph [node]:  — O(Σaj)
        if visited [int(node) - 1] == 0:  — O(1)
            DFS_VISIT (graph, node)

def DFS (graph, endPoint):
    for node in graph:  — O(V)
        if visited [int(node) - 1] == 0:  — O(1)
            DFS_VISIT (graph, node)
    for i in printed:  — O(V)
        print (i, end = " ")  — O(1)
        if i == endPoint:  — O(1)
            break  — O(1)
```

DFS_VISIT method will run Σa_j times, where Σa_j is number of adjacent edges to current vertex.

The first for loop in DFS method will run for V times, where V is the total number of vertices in the graph.

DFS

Time Complexity:

$V + 1 + V(1 + Eaj) + V$

$= 3V + V * Eaj$

$= V + E$ where E is the total number of edges in the graph.

$\therefore O(V+E)$

for Adjacency Matrix,

DFS_VISIT method will run for V times.

The first for loop in DFS method will run for V times.

$1 + V * (1 + V) + V$

$= 1 + V + V^2 + V$

$= O(V^2)$

Gary gets to the victory road first.

The output for Task 2: 1 2 3 4 5 7 11 6 12

The output for Task 3: 1 2 3 4 7 11 12

from the outputs, we can see Gary needs to visit less places to reach the end point.