ID: 19101465

Task: 4

Matrix:

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	0	0	0	0	0	0	0	0	0	0
2	0	0	1	1	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	1	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	1	0	0	0	0	0
6	0	0	0	0	0	0	1	1	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	0	1	1	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0
11	0	0	0	0	0	0	0	0	0	0	0	1
12	0	0	0	0	0	0	0	0	0	0	0	0

Time complexity of BFS in case of Matrix:

In the worst case, the destination will be in the 12th row and 12th column.
The code has to traverse both the column and the row.

$\therefore$ Time complexity $= O(V.E) = O(V.V)$

$= O(V^2)$

Time complexity of DFS in terms of Matrix :

Time complexity will be same as BFS

$= O(V^2)$.

Adjancy list :

1 → $\boxed{2}$

2 → $\boxed{3}$ → $\boxed{4}$ → $\boxed{5}$

3 → $\boxed{9}$ → $\boxed{7}$ → $\boxed{11}$

4 →

5 → $\boxed{6}$ → $\boxed{7}$

6 → $\boxed{7}$ → $\boxed{8}$

7 → $\boxed{11}$

8 → $\boxed{9}$ → $\boxed{10}$

9 → $\boxed{10}$

10 → $\boxed{11}$

11 → $\boxed{12}$

12 →

From the code we can see that,
In this case the loop will run vertex

forms at maximum and the total edges

of the loop will be E.

∴ Time complexity for DFS in terms of

Adjancy list = O (V+E)

```
for j in graph-1 [w]:
    if color [int (des)] != 'white':
        break
    elif color [int (j)] == 'white':
        p[int (j)] = w
    DFS_visit (graph-1, des, int(j), color
                        p, d, f, time)
```

O(V+E)

From the code from Task 2, again, we can see that

The time complexity of BFS in terms of ~~Adjancy~~ Adbacancy list : $O(V+E)$

So, for the both case (BFS and DFS), the adjaency list gets to the victory road first because the time complexity of BFS and DFS in terms of ~~Adjaney~~ ~~Aj~~ Adjacency list is less $O(V+E)$ than Adjacency matrix $O(V^2)$

∴ Adjacency list gets to the victory road first.