Task 5:

BFS:

Adjacency list:

```
BFS (visited, graph, node, endPoint):
    Do visited[int(node)-1] ← 1                    ——— O(1)
    Do queue ← append (node).                      ——— O(1)
    while queue not empty                          ——— O(|V|)
        Do m ← pop()                               ——— O(1)
        Print m                                    ——— O(1)
        If m == endPoint break                     ——— O(1)
        for each neighbour of m in graph:          ——— O(|E|).
            If visited[int(neighbour)-1] == 0      ——— O(1)
            Do visited[int(neighbour)-1] ← 1       ——— O(1)
            Do queue ← append (neighbour)          ——— O(1)
```

Inside the while loop we have,

$0 \xrightarrow{\quad} c + E_0$

$1 \xrightarrow{\quad} c + E_1$

$2 \xrightarrow{\quad} c + E_2$

$V \xrightarrow{\quad} c + E_V$

c means constant operations

$E_0, E_1 \ldots$ means edges in each vertices.

while loop will run for all vertices.

add them we get,

$$Vc + E_0 + E_1 + E_2 \cdots E_V$$

$$\Rightarrow Vc + \sum_{\forall V}^{i} E_i$$

$$\Rightarrow |V| + |E|$$

$\therefore O(|V| + |E|)$ is the time complexity of Bfs using adjacent list.

Matrix: we already know if we have a $n \times n$ matrix for all the combination for the worst case matrix traversal is $O(n^2)$.

So, we will assume in the for loop where we are trying to get the edges, there we are now traversing a matrix for edges. and that will result in $|E| = |V|^2$.

now,

$$\text{Bfs matrix} = O(|V| + |E|)$$

$$= O(|V| + |V|^2)$$

$$\geq O(|V|^2)$$

DFS:

Adjacency list:

```
DFS_VISIT(graph, node):
    Do visited[int(node)-1] ← 1          ——— O(1)
    Printed.append(node)                 ——— O(1)
    for each node in graph[node]         ——— O(|E|)
        If node not visited              ——— O(1)
            DFS_VISIT(graph, node)       ———
```

O(|V|)

```
DFS(graph, endPoint):
    for each node in graph               ——— O(|V|)
        If node not visited.             ——— O(1)
            DFS_VISIT(graph, node)       ———
```

from DFS_VISIT we get,

$$(c + f_0) + (c + f_1) + \cdots + (c + f_v) \leftarrow v \text{ times}$$

$$= cV + \sum_{\forall V}^{i} f_i$$

$$= |V| + |E|.$$

for DFS we get only $|V|$.

for the whole program,

$$|V| + |E| + |V|$$

$$= 2|V| + |E|$$

$$= O(|V| + |E|).$$

Matrix: we will assume $|E| = |V|^2$ because of the matrix. Then the time complexity will be $O(|V|^2)$.

I am using Bfs algo and Gary is using Dfs. The Dfs algo goes to victory road first because victory road is far from starting vertex and Dfs algo traverse each vertex to its depth without exploring the neighbours first, that is why it finds the victory road also known as the last vertex first. On the other hand, Bfs algo goes in level order it explores each vertex first then goes to the next one. That makes this algo

reaching its last vertex after all the vertex explonation.

Not mandatory task:

1. Yes.
2. In the code we have to add another function. The importance of that function is that it will traverse all the vertiees itself without without depending on anything.

```
new BFS (graph, endPoint):
    for each node in graph:
        If node not visited:
            BFS(graph, node, endPoint)

BFS(graph, ...)
```

3. Yes we can detect cycle both directed and undirected graph using DFS traversal.