

Module - 6.5: Practice Problems

1. Rewrite **BFS** in C++ but this time use an **adjacency matrix** as graph representation instead of adjacency list. Analyse the time and space complexity.
2. Rewrite **DFS** in C++ but this time use an **adjacency matrix** as graph representation instead of adjacency list. Analyse the time and space complexity.
3. During graph traversal we saw that there were two steps. One is selecting a graph and the other is exploring the graph. During exploration of a node all its adjacent nodes are “**checked**” and the already visited nodes are ignored.

Now we want to calculate how many times a particular node gets “**checked**”. Modify the existing **BFS** algorithm to calculate how many times each of the nodes get “**checked**”.

Can you guess how many times a particular node gets “**checked**” without coding it?

4. Repeat **problem 3** but this time using **DFS**.
5. Take the following graph as input and determine whether nodes **2** and **6** are connected using **BFS**.
Use the code in this link to take input: <https://ideone.com/t1OAZs>

```
9 11
0 2
7 8
0 4
0 5
6 7
1 4
1 5
2 3
2 4
4 5
8 6
```

6. Redo **problem 5** using **DFS**