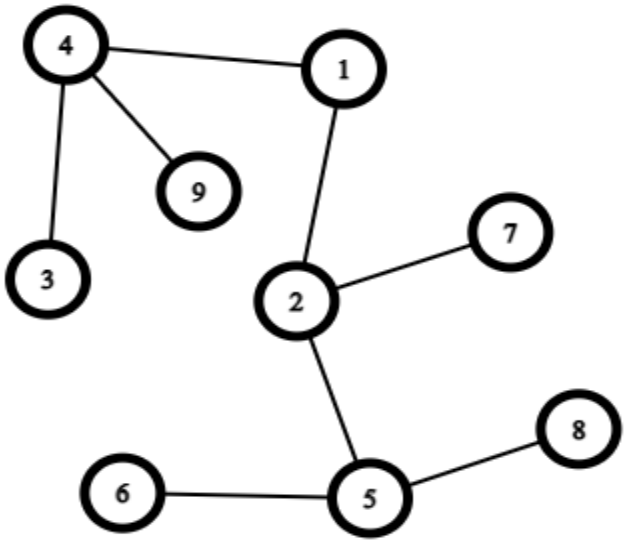
Practice Problem Using DFS

1. Perform **DFS** Traversal on the following graph and write the traversal output. Choose node **4** as the source.



2. Can you think of a connected graph of **6** nodes where **DFS** traversal would produce the output?

3. Your friend wants you to reverse a string in a recursive way. He wants you to write a function **`reverse (string str)`** which takes a string input and returns the reversed string but this function has to be **recursive**.

a. What is the base case?
b. Write one or two examples of how you can solve the larger problem given the smaller problem is already solved.
c. Can you write the generalised formula for this task?
d. Implement the function in C++
e. Simulate your code for an example input.

Example:

```
Input: phitron
Output: nortihp
```

4. Write a **recursive** function **`digitSum(int n)`** that takes a non-negative integer as input and returns the sum of its digits. Follow all the steps of **problem** .
Example:

```
Input: 7464
Output: 21
Explanation: 7 + 4 + 6 + 4 = 21
```

5. Write a **recursive** function **`getCapital(string str)`** that takes a string as input and returns the first capital letter of the input string. Follow all the steps of **problem 4**.
Example:

```
Input: thisStRIng
Output: S
```

6. Rewrite **DFS** in C++ but this time use an **adjacency matrix** as graph representation instead of adjacency list. Analyse the time and space complexity.

7. 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 **DFS** 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?

8. Take the following graph as input and determine whether nodes **2** and **6** are connected using **DFS.** 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
```