Answer Script

Question No. 01

Question no 1

Convert the following Adjacency Matrix into an Adjacency List and draw the graph. **20**
(no need to code)

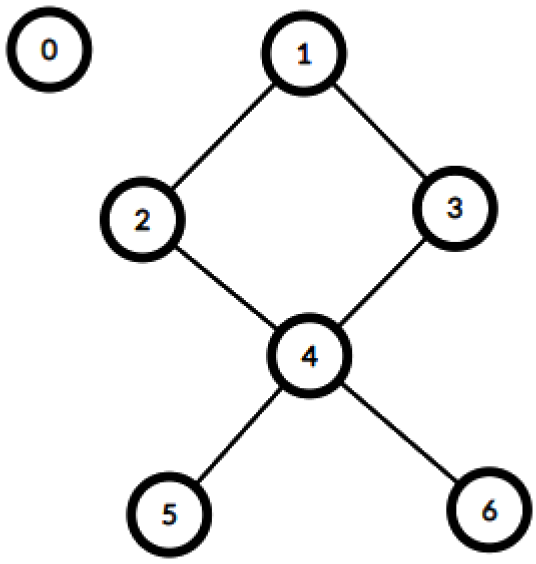
$$\begin{array}{c|ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}$$

Answer No. 01

In the given Adjacency Matrix,
We can see 0 doesn't connect with any edge.
Node 1 connects with 2 edges and they are 2 and 3.
Node 2 connects with 2 edges and they are 1 and 4.
Node 3 connects with 2 edges and they are 1 and 4.
Node 4 connects with 4 edges and they are 2, 3, 5 and 6,
Node 5 connects with 1 edge and its 4.
Node 6 connects with 1 edge and its 4.

Adjacency List:
0 ->
1 -> 2, 3
2 -> 1, 4
3 -> 1, 4
4 -> 2, 3, 5, 6
5 -> 4
6 -> 4

Graph:



Question No. 02

Question no 2

You are given two positive integers n and m . Now calculate the value of n to the power m using recursion.Write a C++ program for it.
20

Sample Input	**Sample Output**
2 4	16

Answer No. 02

Code:

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long int

ll power(int n, int m)
{
    if(m == 0)
        return 1;

    return power(n, m-1) * n;
}

int main()
{
    int n, m;
    cin>>n>>m;

    cout<<power(n, m);

    return 0;
}
```

Question No. 03

Question no 3

What is the difference between BFS and DFS algorithms? (At least Five) **20**

Answer No. 03

BFS	DFS
1.BFS stands for Breadth First Search.	1.DFS stands for Depth First Search.
2.BFS uses a Queue to find the shortest path.	2. DFS uses a Stack to find the shortest path.
3.BFS is slower than DFS.	3.DFS is faster than BFS.
4.BFS builds the tree level by level.	4.DFS builds the tree sub-tree by sub-tree.
5.BFS does not use the backtracking concept.	5.DFS uses backtracking to traverse all the unvisited nodes.
6.BFS requires more memory.	6.DFS requires less memory.

Question No. 04

Question no 4

What is BFS and how does it work? What is DFS and how does it work? (With Figure) **20**

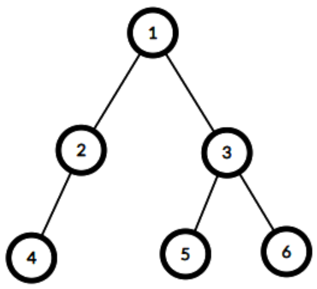
Answer No. 04

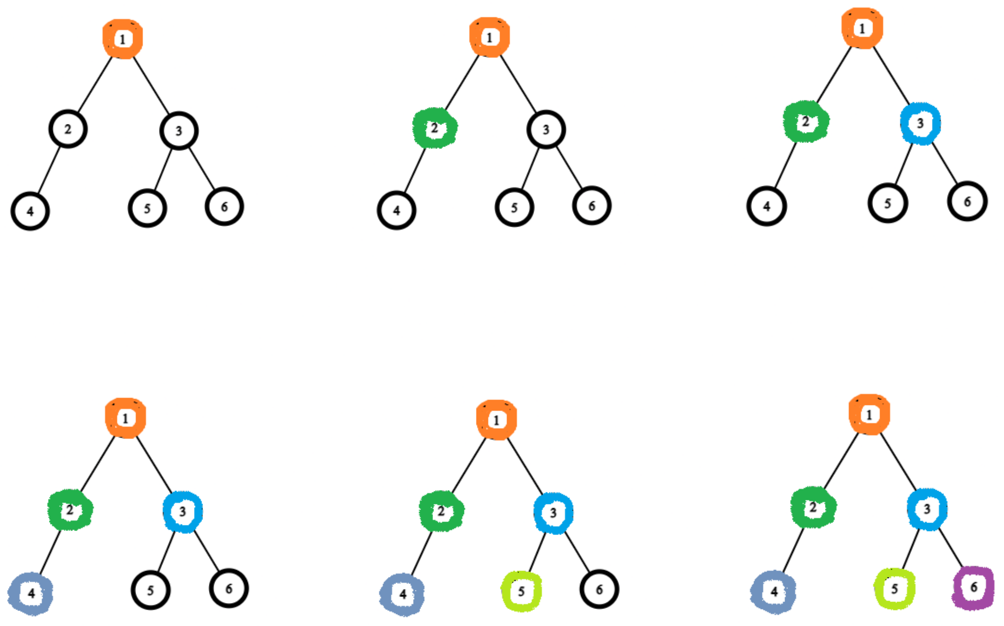
BFS:

BFS stands for Breadth First Search. It is also known as level order traversal. The Queue data structure is used for the Breadth First Search traversal.
BFS is a traversal algorithm that visits all the vertices of a graph or tree in breadth-first order, visiting all the vertices at the same level before moving on to the next level.

In BFS visit all vertices at the same level and then go to the next level.

Example:





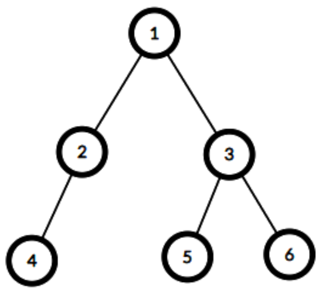
Here, Starting node is 1 and mark it as visited. 1 has two unvisited adjacent nodes in the next level and they are 2 and 3. Now visit 2 and mark it as visited. Next unvisited adjacent node from 1 is 3 and mark it as visited. Then move to the next level. Next unvisited adjacent node from 2 is 4 and mark it visited. Then unvisited adjacent nodes from 3 are 5 and 6. Now visit 5 and mark it as visited. And the last unvisited adjacent node form 3 is 6, visit it and mark it as visited. Node 4, 5 and 6 has no children. All nodes are visited.

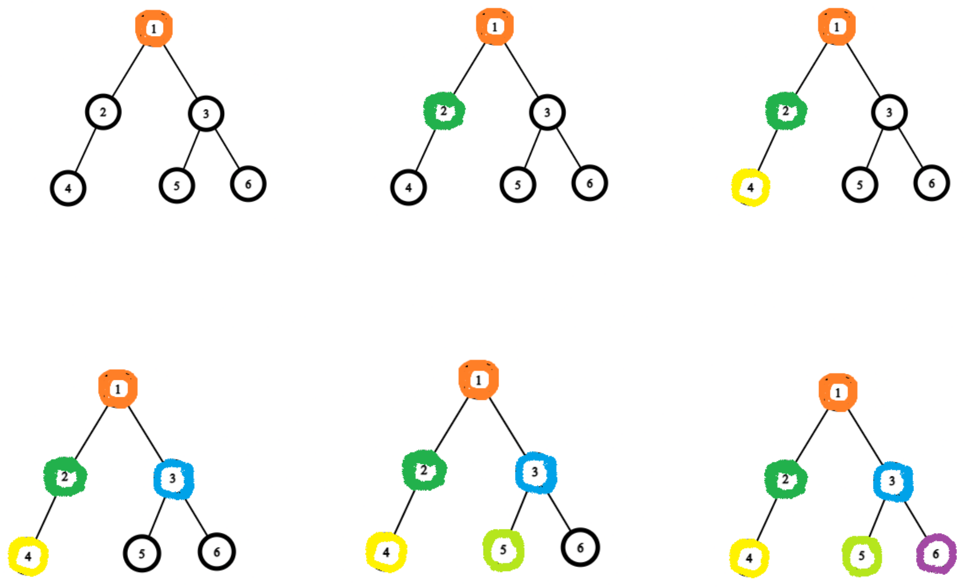
So, Final BFS are: 1, 2, 3, 4, 5, 6

DFS:

DFS stands for Depth First Search. Depth First Search is a recursive algorithm for searching all the vertices of a graph or tree data structure. Traversal means visiting all the nodes of a graph.

Example:

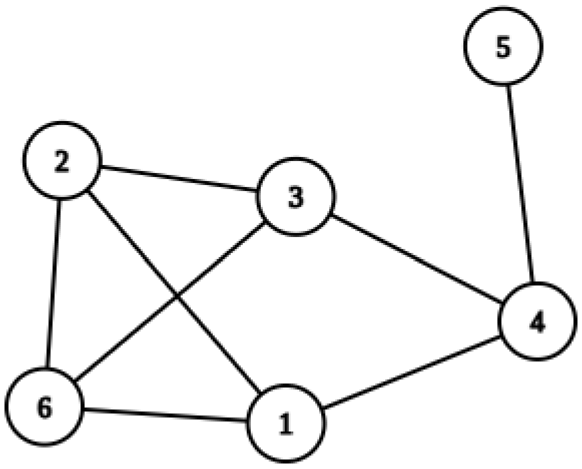




The depth-first search (DFS) algorithm starts with the initial node of the graph and goes deeper until finding the goal node or the node with no children.

Here, Starting node is 1 and mark it as visited. 1 has two unvisited adjacent nodes in the next level and they are 2 and 3. Now visit 2 and mark it as visited. Now 1 has one unvisited adjacent node and its 3. Pause 1 for now. 2 has one unvisited adjacent node and it is 4. Visit 4 and mark it as visited. Pause 2 for now and 4 has no children or unvisited adjacent nodes. Then come back 2 and 2 doesn't have any unvisited adjacent nodes. Then come back 1 and 1 has one unvisited adjacent node and it's 3. Visit 3 and mark it as visited and pause 1 for now. 3 has two unvisited adjacent nodes and they are 5 and 6. Visit 5 and mark it as visited and pause 3 for now. 5 has no children or unvisited adjacent nodes. Then come back 3 and 3 has one univisited adjacent node and its 6. Then visit 6 and mark it as visited. Pause 3 for now. 6 has no children or unvisited adjacent nodes. Then come back 3 and 3 has no unvisited adjacent nodes. Then come back 1 and 1 has no unvisited adjacent nodes. So, finally all nodes are visited.

So, Final DFS are: 1, 2, 4, 3, 5, 6

Question No. 05
<u>Question no 5</u>  Perform BFS and DFS Traversal on the following graph and write the traversal output. Choose node 2 as the source. You must write all steps you perform for doing BFS and DFS Traversal on the following graph. **20**
Answer No. 05

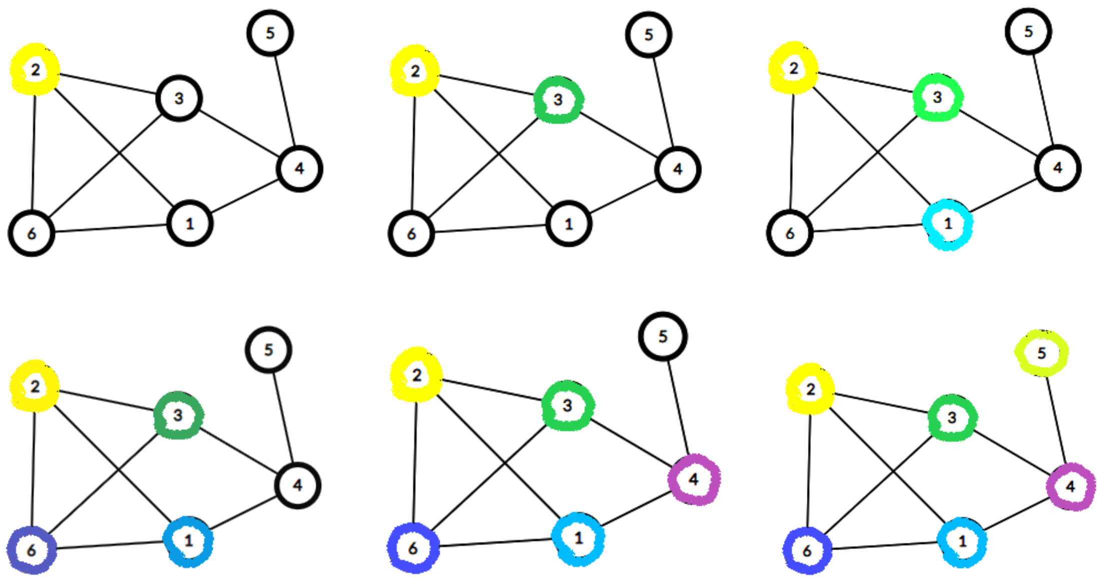
In the given graph it has 6 nodes and 8 vertices.
6 nodes are: 2, 6, 1, 3, 4, 5
8 vertices are:
2 -> 3
2 -> 1
2 -> 6
3 -> 4
3 -> 6
1 -> 4
1 -> 6
4 -> 5

In this graph Adjacent List are:
1 -> 2, 4, 6
2 -> 1, 3, 6
3 -> 2, 4, 6

4 -> 1, 3, 5
5 -> 4
6 -> 1, 2, 3

Given 2 is the source node.

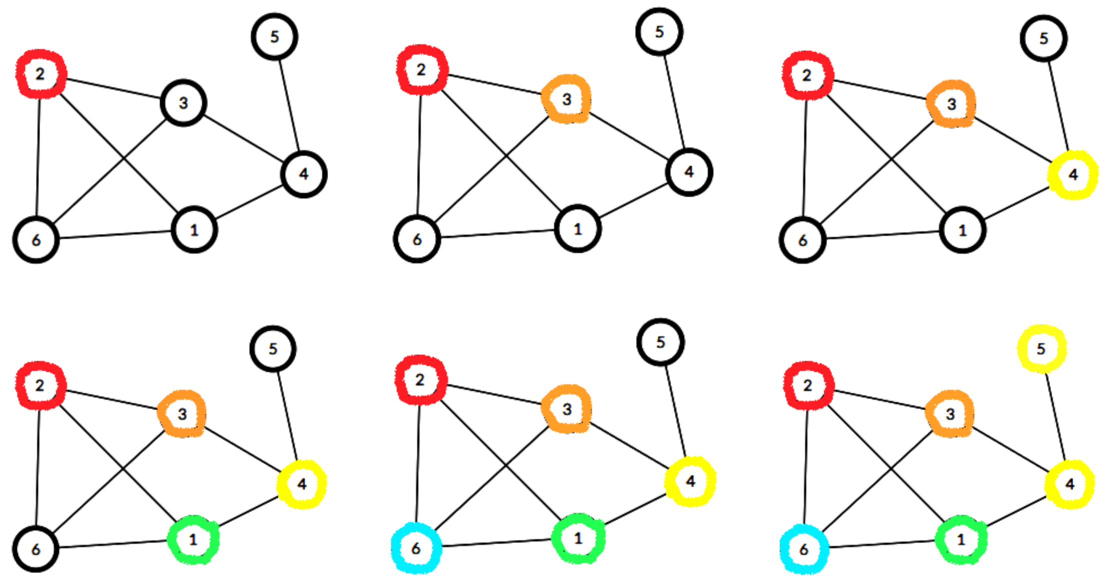
BFS:



Here, given the source node is 2 that's why the starting node is 2. Now visit 2 and mark it as visited. 2 has three unvisited adjacent nodes and they are 3, 1, 6. Now visit node 3 and mark it as visited. Now 2 has two unvisited adjacent nodes 1 and 6. Then visit 1 and mark it as visited. Now 2 has one unvisited adjacent node and it’s 6. Visit 6 and mark it as visited. Finally 2 has no unvisited adjacent node and 6 has no children or unvisited adjacent node. 1 has one unvisited adjacent node and its 4. Now visit it and mark it as visited. 3 has no unvisited adjacent node. Now 4 has one unvisited adjacent node and its 5. Visit 5 and mark it as visited. 5 has no children or unvisited adjacent nodes. Finally they don't have any unvisited adjacent nodes. All nodes are visited.

So, Final BFS are: 2, 3, 1, 6, 4, 5

DFS:



Here, given the source node is 2 that's why the starting node is 2. Now visit 2 and mark it as visited. 2 has three unvisited adjacent nodes and they are 3, 1, 6. Now visit node 3 and mark it as visited. Now 2 has two unvisited adjacent nodes 1 and 6. Pause 2 for now. 3 has one unvisited adjacent node 4. Visit 4 and mark it as visited. Pause 3 for now. 4 has two unvisited adjacent nodes and they are 1 and 5. Visit 1 and mark it as visited. Now 4 has one unvisited adjacent node 5. Pause 4 for now. 1 has one unvisited adjacent node and its 6. Visit 6 and mark it as visited. 6 has no unvisited adjacent node. Come back 1 and 1 has no unvisited adjacent node. Then come back 4 and 4 has one unvisited adjacent node and its 5. Visit 5 and mark it as visited. Pause 4 for now. 5 has no children or unvisited adjacent nodes. Then come back 4 and 4 has no unvisited adjacent node. Come back 3 and 3 has no unvisited adjacent node. Then come back 2 and 2 has no unvisited adjacent node. All nodes are visited.

So, Final DFS are: 2, 3, 4, 1, 6, 5