Signup and get free access to 100+ Tutorials and Practice Problems Start Now



3 LIVE EVENTS

Algorithms

Solve any problem to achieve a rank

View Leaderboard

Topics: Breadth First Search

# Breadth First Search

TUTORIAL PROBLEMS VISUALIZER BETA

### Graph traversals

Graph traversal means visiting every vertex and edge exactly once in a well-defined order. While using certain graph algorithms, you must ensure that each vertex of the graph is visited exactly once. The order in which the vertices are visited are important and may depend upon the algorithm or question that you are solving.

During a traversal, it is important that you track which vertices have been visited. The most common way of tracking vertices is to mark them.

### Breadth First Search (BFS)

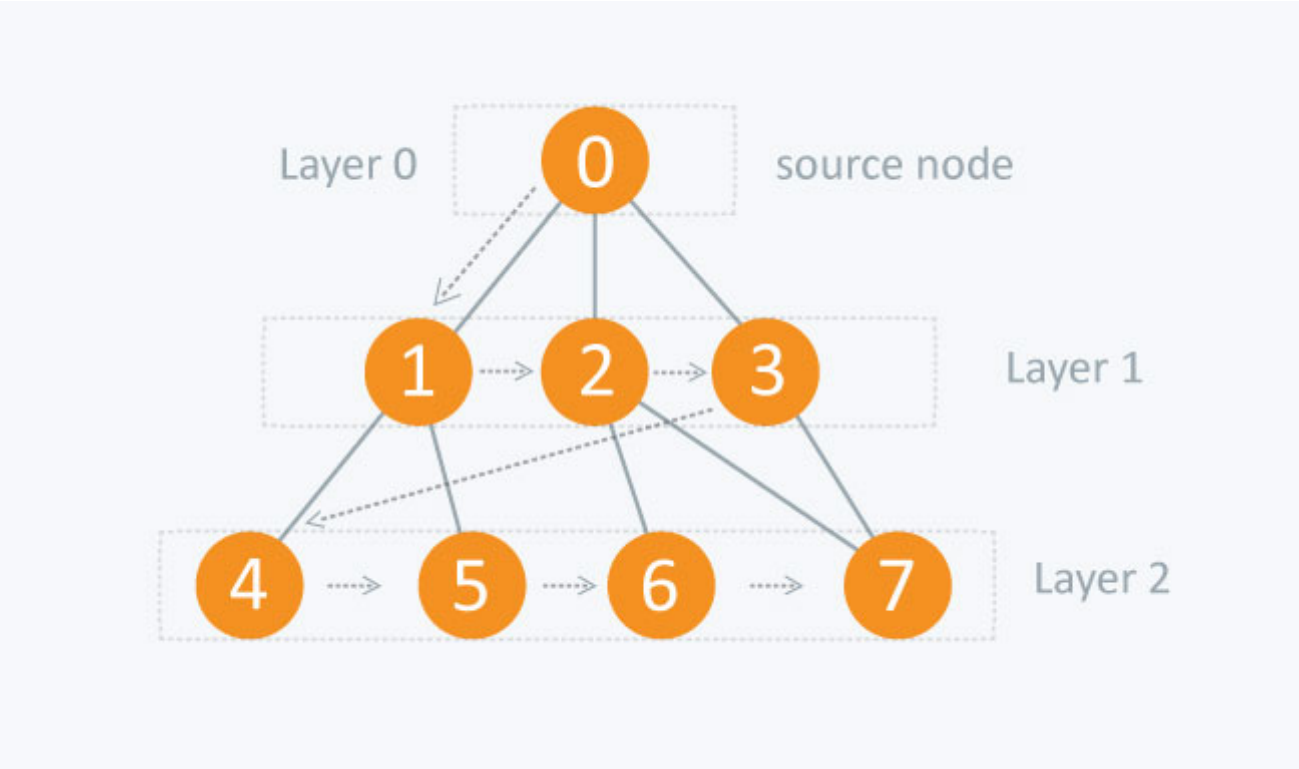
There are many ways to traverse graphs. BFS is the most commonly used approach.

BFS is a traversing algorithm where you should start traversing from a selected node (source or starting node) and traverse the graph layerwise thus exploring the neighbour nodes (nodes which are directly connected to source node). You must then move towards the next-level neighbour nodes.

As the name BFS suggests, you are required to traverse the graph breadthwise as follows:

1. First move horizontally and visit all the nodes of the current layer
2. Move to the next layer

Consider the following diagram.




The distance between the nodes in layer 1 is comparitively lesser than the distance between the nodes in layer 2. Therefore, in BFS, you must traverse all the nodes in layer 1 before you move to the nodes in layer 2.

*Traversing child nodes*

A graph can contain cycles, which may bring you to the same node again while traversing the graph. To avoid processing of same node again, use a boolean array which marks the node after it is processed. While visiting the nodes in the layer of a graph, store them in a manner such that you can traverse the corresponding child nodes in a similar order.

In the earlier diagram, start traversing from 0 and visit its child nodes 1, 2, and 3. Store them in the order in which they are visited. This will allow you to visit the child nodes of 1 first (i.e. 4 and 5), then of 2 (i.e. 6 and 7), and then of 3 (i.e. 7) etc.

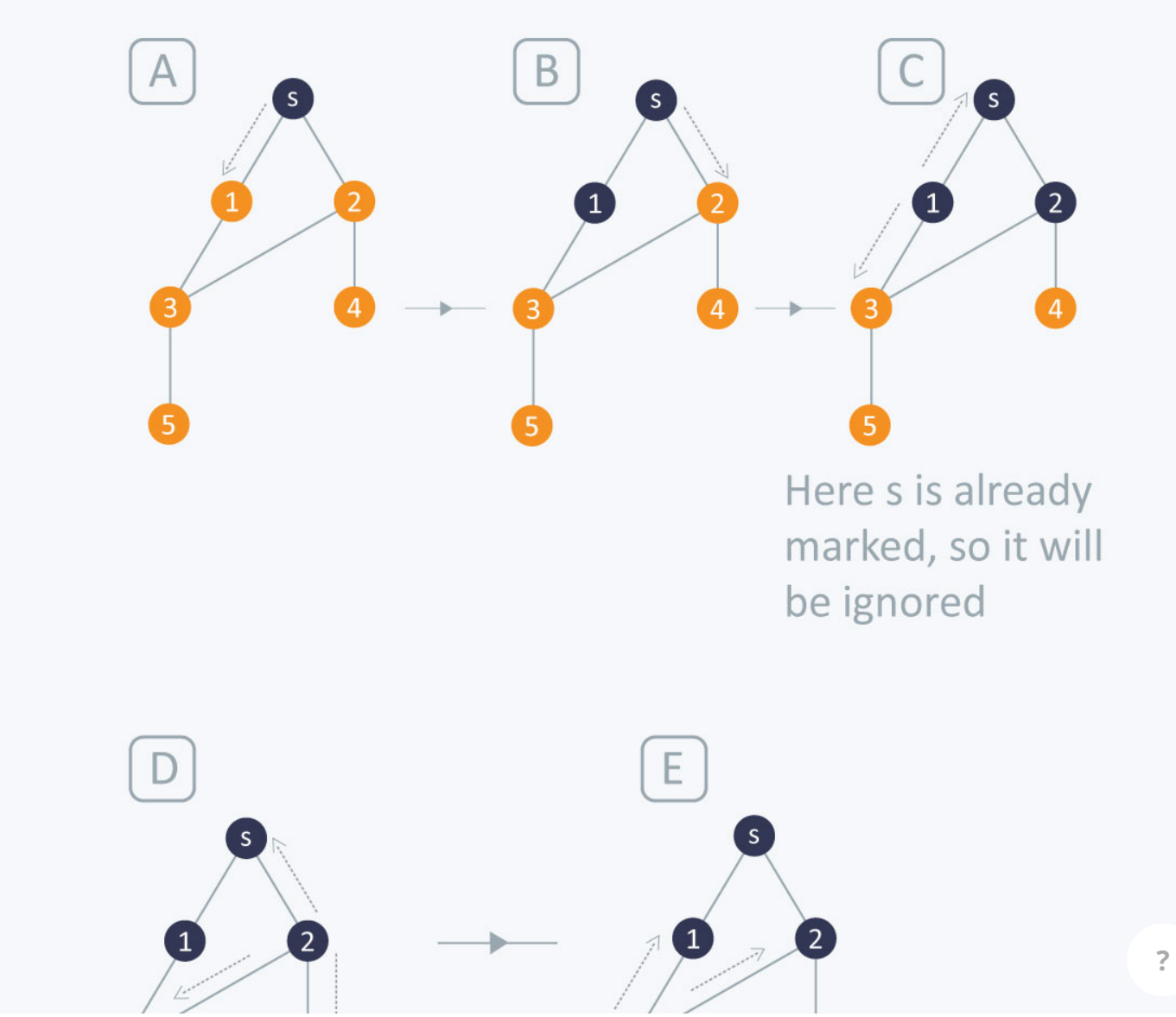
To make this process easy, use a queue to store the node and mark it as 'visited' until all its neighbours (vertices that are directly connected to it) are marked. The queue follows the First In First Out (FIFO) queuing method, and therefore, the neigbors of the node will be visited in the order in which they were inserted in the node i.e. the node that was inserted first will be visited first, and so on.

**Pseudocode**

```
BFS (G, s)                    //Where G is the graph and s is the source
node
      let Q be queue.
```

```
      Q.enqueue( s ) //Inserting s in queue until all its neighbour vertices
are marked.

      mark s as visited.
      while ( Q is not empty)
           //Removing that vertex from queue,whose neighbour will be
visited now
           v   =   Q.dequeue( )

          //processing all the neighbours of v
          for all neighbours w of v in Graph G
               if w is not visited
                         Q.enqueue( w )               //Stores w in Q to
further visit its neighbour
                         mark w as visited.
```

Traversing process




The traversing will start from the source node and push *s* in queue. *s* will be marked as 'visited'.

*First iteration*

- s will be popped from the queue
- Neighbors of s i.e. 1 and 2 will be traversed
- 1 and 2, which have not been traversed earlier, are traversed. They will be:
  - Pushed in the queue
  - 1 and 2 will be marked as visited

*Second iteration*

- 1 is popped from the queue

- Neighbors of 1 i.e. s and 3 are traversed
- s is ignored because it is marked as 'visited'
- 3, which has not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

*Third iteration*

- 2 is popped from the queue
- Neighbors of 2 i.e. s, 3, and 4 are traversed
- 3 and s are ignored because they are marked as 'visited'
- 4, which has not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

*Fourth iteration*

- 3 is popped from the queue
- Neighbors of 3 i.e. 1, 2, and 5 are traversed
- 1 and 2 are ignored because they are marked as 'visited'
- 5, which has not been traversed earlier, is traversed. It is:
    - Pushed in the queue
    - Marked as visited

*Fifth iteration*

- 4 will be popped from the queue
- Neighbors of 4 i.e. 2 is traversed
- 2 is ignored because it is already marked as 'visited'

*Sixth iteration*

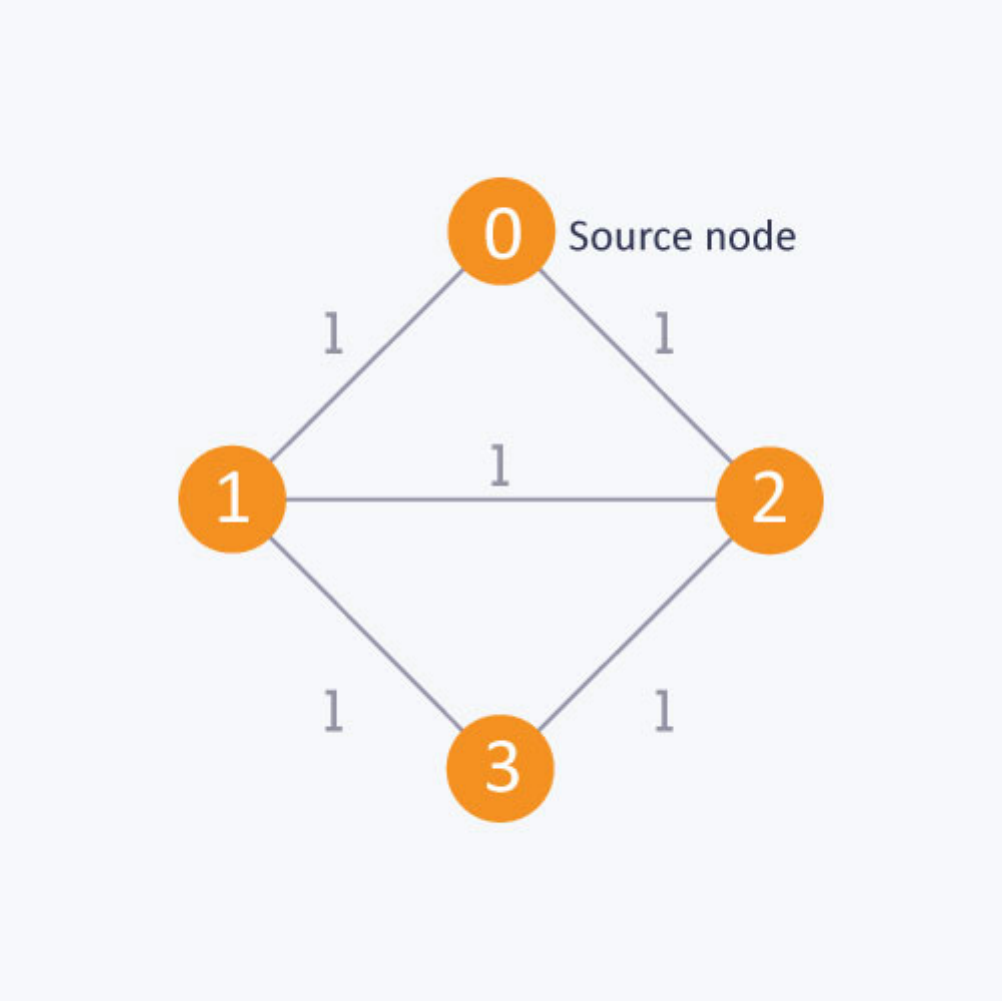
- 5 is popped from the queue
- Neighbors of 5 i.e. 3 is traversed
- 3 is ignored because it is already marked as 'visited'

The queue is empty and it comes out of the loop. All the nodes have been traversed by using BFS.

If all the edges in a graph are of the same weight, then BFS can also be used to find the minimum distance between the nodes in a graph.

***Example***

?




As in this diagram, start from the source node, to find the distance between the source node and node 1. If you do not follow the BFS algorithm, you can go from the source node to node 2 and then to node 1. This approach will calculate the distance between the source node and node 1 as 2, whereas, the minimum distance is actually 1. The minimum distance can be calculated correctly by using the BFS algorithm.

**Complexity**

The time complexity of BFS is **O(V + E)**, where V is the number of nodes and E is the number of edges.

**Applications**

***1. How to determine the level of each node in the given tree?***

As you know in BFS, you traverse level wise. You can also use BFS to determine the level of each node.

*Implementation*

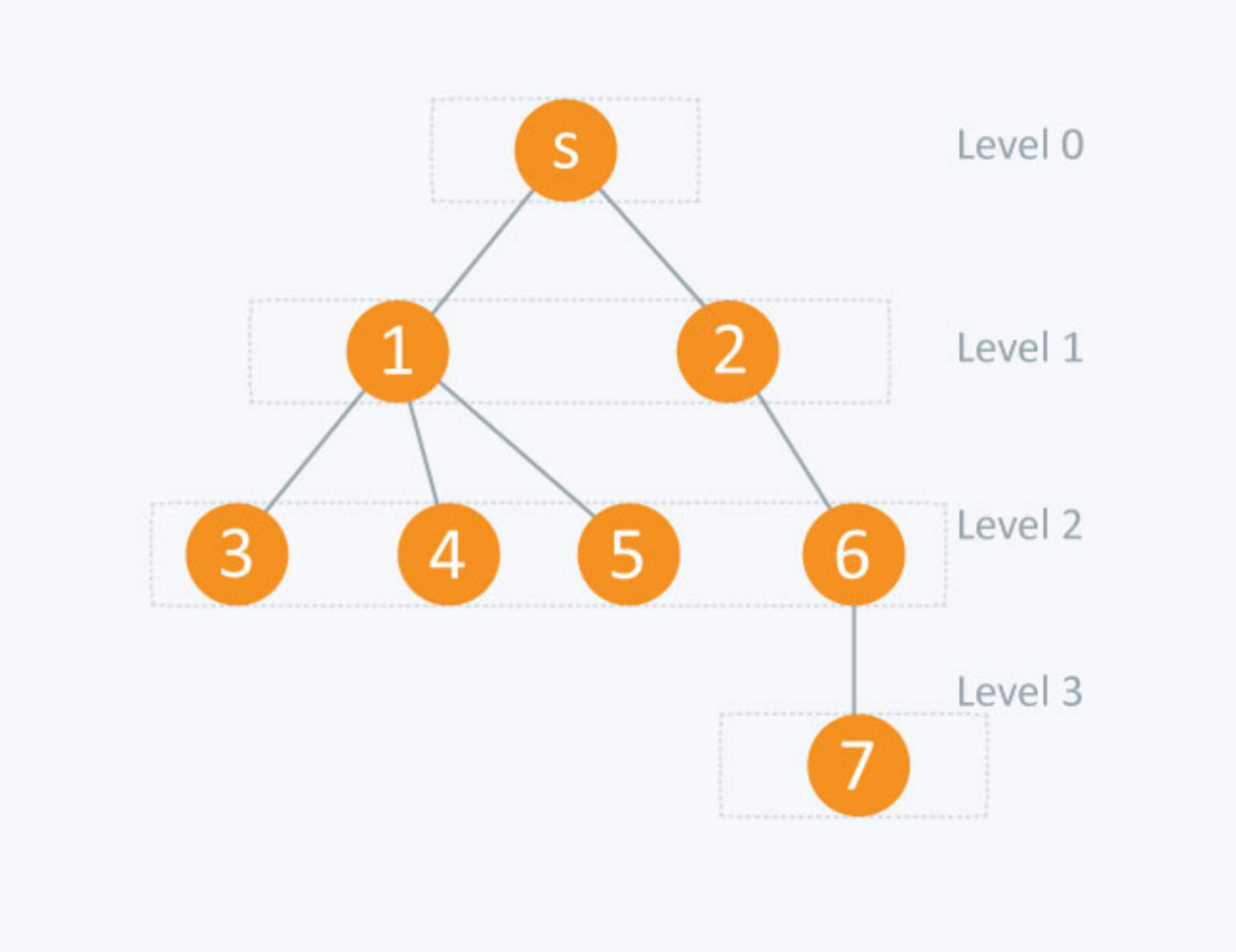
```
   vector <int> v[10] ;   //Vector for maintaining adjacency list explained
above
```

```
    int level[10]; //To determine the level of each node
    bool vis[10]; //Mark the node if visited
    void bfs(int s) {
        queue <int> q;
        q.push(s);
        level[ s ] = 0 ;  //Setting the level of the source node as 0
        vis[ s ] = true;
        while(!q.empty())
        {
            int p = q.front();
            q.pop();
            for(int i = 0;i < v[ p ].size() ; i++)
            {
                if(vis[ v[ p ][ i ] ] == false)
                {
            //Setting the level of each node with an increment in the level
of parent node
                    level[ v[ p ][ i ] ] = level[ p ]+1;
                     q.push(v[ p ][ i ]);
                     vis[ v[ p ][ i ] ] = true;
      }
            }
        }
    }
```

This code is similar to the BFS code with only the following difference:
**level[ v[ p ][ i ] ] = level[ p ]+1;**

In this code, while you visit each node, the level of that node is set with an increment in the level of its parent node. This is how the level of each node is determined.

?




| node | level [node] |
|---|---|
| s (source node) | 0 |
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| 4 | 2 |
| 5 | 2 |
| 6 | 2 |
| 7 | 3 |

*2. 0-1 BFS*

This type of BFS is used to find the shortest distance between two nodes in a graph provided that the edges in the graph have the weights 0 or 1. If you apply the BFS explained earlier in this article, you will get an incorrect result for the optimal distance between 2 nodes.

In this approach, a boolean array is not used to mark the node because the condition of the optimal distance will be checked when you visit each node. A double-ended queue is used to store the node

?

In 0-1 BFS, if the weight of the edge = 0, then the node is pushed to the front of the dequeue. If the weight of the edge = 1, then the node is pushed to the back of the dequeue.

*Implementation*

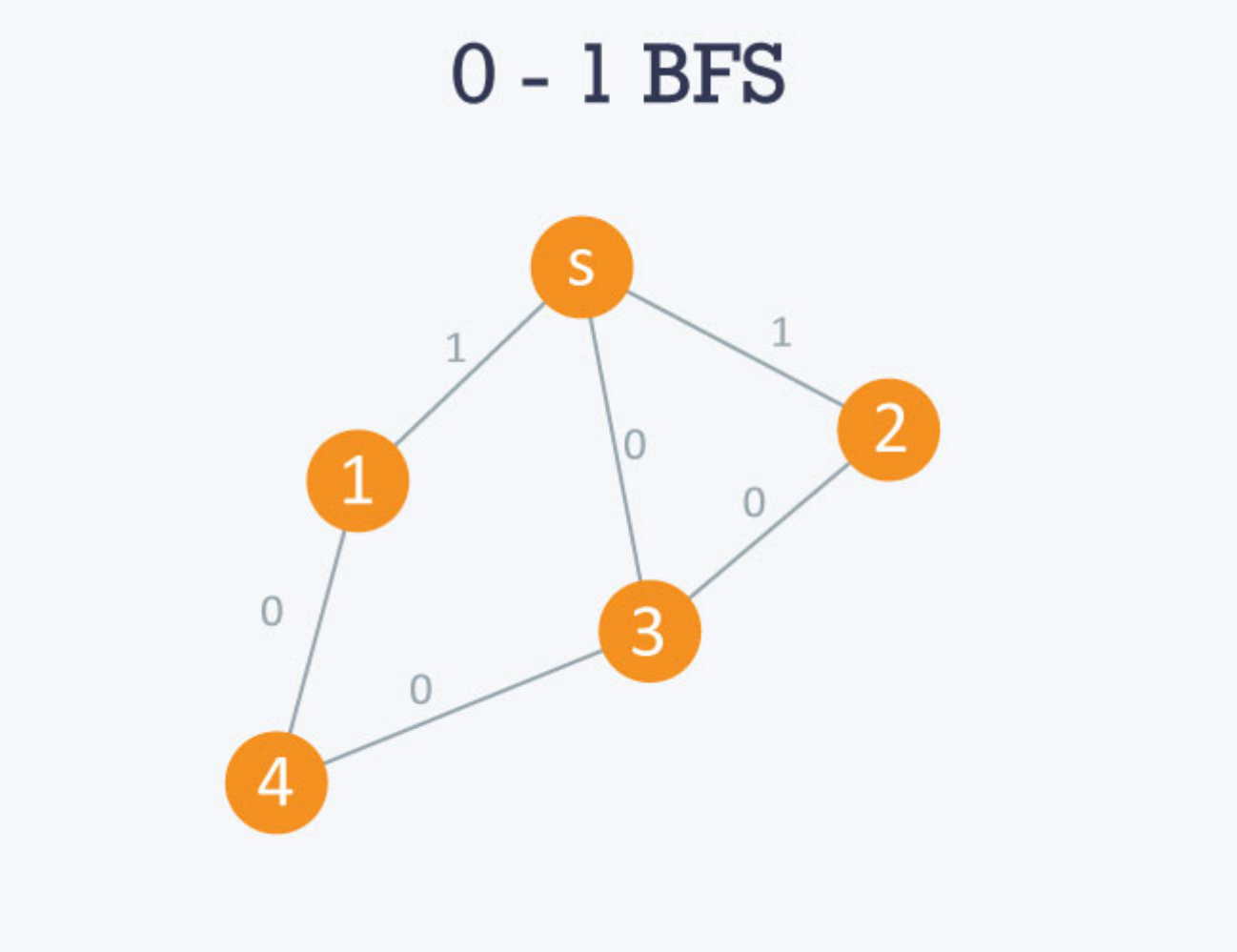
Here, ***edges[ v ] [ i ]*** is an adjacency list that exists in the form of pairs i.e. ***edges[ v ][ i ].first*** will contain the node to which ***v*** is connected and ***edges[ v ][ i ].second*** will contain the distance between ***v*** and ***edges[ v ][ i ].first***.

***Q*** is a double-ended queue. The distance is an array where, ***distance[ v ]*** will contain the distance from the start node to ***v*** node. Initially the distance defined from the source node to each node is infinity.

```
void bfs (int start)
{
            deque <int > Q;      //Double-ended queue
            Q.push_back( start);
            distance[ start ] = 0;
            while( !Q.empty ())
            {
                 int v = Q.front( );
                 Q.pop_front();
                 for( int i = 0 ; i < edges[v].size(); i++)
                {


/* if distance of neighbour of v from start node is greater than sum of
distance of v from start node and edge weight between v and its neighbour
(distance between v and its neighbour of v) ,then change it */


                     if(distance[ edges[ v ][ i ].first ] > distance[ v ] +
edges[ v ][ i ].second )
                 {

                     distance[ edges[ v ][ i ].first ] = distance[ v ] +
edges[ v ][ i ].second;

                 /*if edge weight between v and its neighbour is 0 then
push it to front of
         double ended queue else push it to back*/

                     if(edges[ v ][ i ].second == 0)
                     {
                          Q.push_front( edges[ v ][ i ].first);
                     }
```

```
                        }
                        else
                        {
                                Q.push_back( edges[ v ][ i ].first);

                        }
                    }
                }
            }
    }
```

Let's understand this code with the following graph:




The adjacency list of the graph will be as follows:
Here 's' is considered to be 0 or source node.

0 -> 1 -> 3 -> 2
edges[ 0 ][ 0 ].first = 1 , edges[ 0 ][ 0 ].second = 1
edges[ 0 ][ 1 ].first = 3 , edges[ 0 ][ 1 ].second = 0
edges[ 0 ][ 2 ].first = 2 , edges[ 0 ][ 2 ].second = 1

?

1 -> 0 -> 4
edges[ 1 ][ 0 ].first = 0 , edges[ 1 ][ 0 ].second = 1
edges[ 1 ][ 1 ].first = 4 , edges[ 1 ][ 1 ].second = 0

2 -> 0 -> 3
edges[ 2 ][ 0 ].first = 0 , edges[ 2 ][ 0 ].second = 0
edges[ 2 ][ 1 ].first = 3 , edges[ 2 ][ 1 ].second = 0

3 -> 0 -> 2 -> 4
edges[ 3 ][ 0 ].first = 0 , edges[ 3 ][ 0 ].second = 0
edges[ 3 ][ 2 ].first = 2 , edges[ 3 ][ 2 ].second = 0
edges[ 3 ][ 3 ].first = 4 , edges[ 3 ][ 3 ].second = 0

4 -> 1 -> 3
edges[ 4 ][ 0 ].first = 1 , edges[ 4 ][ 0 ].second = 0
edges[ 4 ][ 1 ].first = 3 , edges[ 4 ][ 1 ].second = 0

If you use the BFS algorithm, the result will be incorrect because it will show you the optimal distance between s and node 1 and s and node 2 as 1 respectively. This is because it visits the children of s and calculates the distance between s and its children, which is 1. The actual optimal distance is 0 in both cases.

**Processing**

Starting from the source node, i.e 0, it will move towards 1, 2, and 3. Since the edge weight between 0 and 1 and 0 and 2 is 1 respectively, 1 and 2 will be pushed to the back of the queue. However, since the edge weight between 0 and 3 is 0, 3 will pushed to the front of the queue. The distance will be maintained in distance array accordingly.

3 will then be popped from the queue and the same process will be applied to its neighbours, and so on.

*Contributed by: Prateek Garg*

**TEST YOUR UNDERSTANDING**

## Level Nodes

You have been given a Tree consisting of *N* nodes. A tree is a fully-connected graph consisting of *N* nodes and $N-1$ edges. The nodes in this tree are indexed from *1* to *N*. Consider node indexed *1*

?

be the root node of this tree. The root node lies at level one in the tree. You shall be given the tree and a single integer *x* . You need to find out the number of nodes lying on level *x*.

**Input Format**

The first line consists of a single integer *N* denoting the number of nodes in the tree. Each of the next $n-1$ lines consists of 2 integers *a* and *b* denoting an undirected edge between node *a* and node *b*. The next line consists of a single integer *x*.

**Output Format**

You need to print a single integers denoting the number of nodes on level *x*.

**Constraints**

$1 \leq N \leq 10^5$

$1 \leq a, b \leq N$

**Note**

It is guaranteed that atleast one node shall lie on level *x*

SAMPLE INPUT

```
20
11 1
1 2
13 3
15 4
17 5
11 6
2 7
1 8
15 9
4 10
15 12
5 13
2 14
17 15
15 16
11 17
15 18
9 19
16 20
2
```

SAMPLE OUTPUT

3

Enter your code or Upload your code as file. Save C (gcc 5.4.0)

```
/*
// Sample code to perform I/O:
#include <stdio.h>

int main(){
    int num;
    scanf("%d", &num);                          // Reading input from STDIN
    printf("Input number is %d.\n", num);       // Writing output to STDOUT
}

// Warning: Printing unwanted or ill-formatted data to output will cause the test cases t
*/

// Write your code here

```

1:1

Provide custom input

COMPILE & TEST SUBMIT

COMMENTS (127) SORT BY: Relevance

Login/Signup to Comment

**Krishna Bisht** Edited 2 years ago

how to know which is the source node from input (given)? Is first value of a considered as the source node?

4 votes ● Reply ● Message ● Permalink

**Prateek Garg** Moderator 2 years ago

It depends from problem to problem. If it is required to know, which node you have to choose as source node, it will be surely mentioned in the problem.

4 votes ● Reply ● Message ● Permalink

**Rishabh Sairawat** a year ago

what's wrong with my code:
import java.util.*;
class TestClass {
int V;

?

```
LinkedList<Integer> adj[];
TestClass(int v){
V=v;
adj=new LinkedList[v];
for(int i=0;i<v;i++)
adj[i]=new LinkedList<>();
}
void addEdge(int u,int v){
adj[u].add(v);
}
void BFS(int s,int x){
boolean visited[]=new boolean[V];
int[] levelcount=new int[s];
int level[]=new int[s];
Queue<Integer> queue=new LinkedList<Integer>();
visited[s]=true;
level[s]=1;
levelcount[level[s]]++;
queue.add(s);
while(!queue.isEmpty()){
int temp=queue.poll();
for(Integer i:adj[temp]){
if(!visited[i]){
level[i]=level[temp]+1;
levelcount[level[i]]++;
queue.add(i);
visited[i]=true;
}
}
}
System.out.println(levelcount[x]);

}
public static void main(String args[] ) throws Exception {

Scanner sc=new Scanner(System.in);
int N = sc.nextInt();
TestClass tree=new TestClass(N);
for(int i=1;i<N;i++){
int a=sc.nextInt();
int b=sc.nextInt();
tree.addEdge(a,b);
}

int x = sc.nextInt();
tree.BFS(1,x);
}
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**GOUTAM KUMAR** 2 years ago

it's a 1indexed tree and root node is given i.e 1

▲ 1 vote ● Reply ● Message ● Permalink

**Krishna Bisht** 2 years ago

yea i have solved it thanks :)

▲ 0 votes ● Reply ● Message ● Permalink

**Hritik Soni** 9 months ago

In an undirected edge there is no one source.. both are sources

?

▲ 0 votes ● Reply ● Message ● Permalink

**akshat sharma** a year ago

help my code is not doing well i cant see any error

```
#include<iostream>
#include<vector>
#include<queue>
using namespace std;
#define N 100000
int k;
vector <int> adj[N];
int level[N];// todetermine the levelof each node
bool visited[N];
void BFS(int s,int x)
{
level[s]=0;
queue <int> q;
q.push(s);
visited[s]=true;
while(!q.empty())
{
int p=q.front();
q.pop();
for(int i=0;i<adj[p].size();i++)
{
if(visited[adj[p][i]]==false)
{
level[adj[p][i]]=level[p]+1;
if(level[adj[p][i]]==x)
{
k++;
}
q.push(adj[p][i]);
visited [adj[p][i]]=true;
}
}
}
}
int main()
{
int x,y,nodes,edges;
cin>>nodes>>edges;
for(int i=0;i<edges;i++)
{
cin>>x;
cin>>y;
adj[x].push_back(y);
}
int b;
cin>>b;
BFS( 1,b);
cout<<k;
return 0;
}
```

▲ 2 votes ● Reply ● Message ● Permalink

**Aditya Nand** 8 months ago

Hey Akshat, I found the following problems in your code.

?

1) Read properly the input format. You are trying to input "nodes" and "edges", whereas according to the question they will only give you the number of nodes. (Now try to figure out the number of edges)
2) Second error is in your initialization of level[s]=0. This is wrong since your are trying to calculate level by following 0-order convention. Whereas in the question it is mentioned to follow 1-order convention. So just change level[s]=1
3) Another major flaw I found is that you are treating the graph as a Directed graph whereas in the question it is mentioned clearly that it is an Undirected graph. Just add another line adj[y].push_back(x)

And now you are good to go!
If still in doubt, get back for sure!!

▲ 3 votes ● Reply ● Message ● Permalink

**Abhay Kumar** 2 months ago

I DON'T UNDERSTAND WHAT IS WRONG

```
#include<bits/stdc++.h>
using namespace std;
int main()
{ int n,x,y,a;cin>>n;vector<int>A[n+1];
for(int i=0;i<n-1;i++)
{cin>>x>>y;
A[x].push_back(y);
A[y].push_back(x);}
cin>>a;
queue<int>q;
int mark[n+1]={0};
int lev[n+1]={0};
q.push(1);mark[1]=1;
lev[1]=1;
while(!q.empty())
{ int c=q.front();q.pop();
for(int j=0;j<A[c].size();j++)
{ if(mark[A[c][j]]==0){q.push(A[c][j]); mark[A[c][j]]=1;lev[A[c][j]]=lev[A[c]]+1;}}}
int N=0;
for(int i=1;i<=n;i++){if(lev[i]==a){N++;}}
cout<<N<<endl;}
```

▲ 0 votes ● Reply ● Message ● Permalink

**sai krishna** Edited 2 months ago

1. vector cannot be taken like that . There should be a constant value or you have to use vector<vector<int>> .so change to vector<int> A[100001]
2. Inside if statement in while loop, it is level[a[c][j]]= level[c]+1; not level[a[c]]
make these changes its working fine

▲ 1 vote ● Reply ● Message ● Permalink

**Harish Gutlapalli** 6 months ago

very small modifiction to your code

```
#include<iostream>
#include<vector>
#include<queue>
using namespace std;
#define N 100000
int k;
vector <int> adj[N];
int level[N];// todetermine the levelof each node
bool visited[N];
void BFS(int s,int x)
{
```

?

```
level[s]=1;
queue <int> q;
q.push(s);
visited[s]=true;
while(!q.empty())
{
int p=q.front();
q.pop();
for(int i=0;i<adj[p].size();i++)
{
if(visited[adj[p][i]]==false)
{
level[adj[p][i]]=level[p]+1;
if(level[adj[p][i]]==x)
{
k++;
}
q.push(adj[p][i]);
visited [adj[p][i]]=true;
}
}
}
}
int main()
{
int x,y,nodes,edges;
cin>>nodes;
for(int i=0;i<nodes-1;i++)
{
cin>>x;
cin>>y;
adj[x].push_back(y);
adj[y].push_back(x);
}
int b;
cin>>b;
BFS( 1,b);
cout<<k;
return 0;
}
```

▲ 1 vote ● Reply ● Message ● Permalink

**prakhar9** Edited a year ago

I'm getting the same output on my system (Ubuntu 16.04) as required by problem but the answer calculated by the judge is different, both the test cases produce the correct output at my system but not on the online judge. Why is that?

▲ 1 vote ● Reply ● Message ● Permalink

**prakhar9** a year ago

I'd forgotten to put visited[] to false. As bool in cpp may assume true or false arbitrarily, probably why it gave error.

▲ 3 votes ● Reply ● Message ● Permalink

**Leon Zaher** 2 years ago

This algorithm is wrong in case where graph is not a tree, for example if graph has a cycle. You can try it on example with undirected graph with 3 vertices and 3 edges, it will visit one of the edges twice. For it to work right, you need to mark vertex as visited inside the IF. To mark vertices as visited, I u set, which will remove duplicates as well. ?

▲ 2 votes ● Reply ● Message ● Permalink

**Prateek Garg** 2 years ago

Hey, the entire tutorial has been updated with major changes. Please specify if you have any doubts.

▲ 1 vote ● Reply ● Message ● Permalink

**Rishabh Rathore** a year ago

There is still a part left while doing this question i have cleared 1 test case but another one failed that case was when my queue is empty but still there are nodes which are not the neighbour of visited nodes ARE LEFT to travese and getting there parent is not possible that how to assign the level for such nodes which has no parent and even there neighbours are having no parents in visited nodes

▲ 0 votes ● Reply ● Message ● Permalink

**Kapil Khandelwal** 2 years ago

https://code.hackerearth.com/4cc82dl

Can anyone tell me why my code is showing NZEC error???

▲ 1 vote ● Reply ● Message ● Permalink

**Sheel Nidhi Pandey** a year ago

```
public class TestClass {
public static int queue[],front=0,rear=-1;

public static void enqueue(int x)
{

queue[++rear]=x;
}
public static int dequeue()
{
front++;
return (queue[front-1]);
}
public static boolean isEmpty()
{
if(front==rear)
return true;

return false;
}
public static void main(String args[] ) {
Scanner sc=new Scanner(System.in);
int n=sc.nextInt();
int a[][]=new int[n][n];
int visited[]=new int[n];
int level[]=new int[n];
queue=new int[n];
for(int i=0;i<n;i++)
{
level[i]=0;
visited[i]=0;
queue[i]=0;
}
for(int i=0;i<n;i++)
for(int j=0;j<n;j++)
a[i][j]=0;
for(int ii=1;ii<n;ii++)
{
```

?

```
int x[]=new int[2];
for(int i=0;i<2;i++)
x[i]=sc.nextInt();
a[x[0]-1][x[1]-1]=1;
a[x[1]-1][x[0]-1]=1;
//System.out.println("n");
}

//sc.nextLine();
int x=sc.nextInt();
// System.out.println(x);
enqueue(x);
visited[0]=1;
level[0]=0;

while(! isEmpty())
{

int temp=dequeue();
for(int i=0;i<n;i++)
{
if(a[temp][i]==1 && visited[i]==0)
{
enqueue(i);
visited[i]=1;
level[i]=1+level[temp];
}
}
}
int count=0;
for(int i=0;i<n;i++)
{
if(level[i]==x)
count++;
}
System.out.println(count);


}
}
```

Now correct your logic..

▲ 1 vote ● Reply ● Message ● Permalink

**Abhinav Tripathi** a year ago

NZEC has nothing to do with logic, atleast it doesn't seem so. Just surrounding the code with try-catch seems to fix the NZEC error.

▲ 1 vote ● Reply ● Message ● Permalink

**Harsh Vardhan Singh** a year ago

Someone please check the output of the test case of question level nodes. I think there is a mistake.

▲ 2 votes ● Reply ● Message ● Permalink

**Rohit** a year ago

MY LOGIC IS WORKING GOOD;

```
#include <iostream>
#include<bits/stdc++.h>
using namespace std;
int main()
{vector<int> v[100001];
int nodes,x,y,level[100000],sum[10000];
```

?

```
bool mark[100000];
memset(mark,false,100000);
cin>>nodes;
for(int i=0;i<nodes-1;i++)
{cin>>x>>y;
v[x].push_back(y);
v[y].push_back(x);}
queue<int> q;
int start=1,c=0;
q.push(start);
level[start]=1;
mark[start]=true;
int i;
cin>>i;
while(!q.empty())
{int p=q.front();
q.pop();
for(vector<int>::iterator it=v[p].begin();it!=v[p].end();++it)
{if(mark[*it]==false)
{level[*it]=level[p]+1;
if(level[*it]==i)c++;
q.push(*it);
mark[*it]=true;
}}}
cout<<c<<endl;
return 0;}
```

▲ 1 vote ● Reply ● Message ● Permalink

**sasuke** 8 months ago

can you explain last part after for loop?

▲ 1 vote ● Reply ● Message ● Permalink

**Harkirat Singh** a year ago

There is some problem in the compiler. Getting right answer on my pc with the same inputs but here it is showing incorrect.

▲ 2 votes ● Reply ● Message ● Permalink

**mhammad othman** 2 years ago

Thank you very much

▲ 0 votes ● Reply ● Message ● Permalink

**Krishna Bisht** 2 years ago

The answer for input1 is wrong. It should be 16. At level 4 there are 16 nodes not 5.Even no level has 5 nodes (level 1=1 node,level 2=2 nodes,level 3=11 nodes,level 4=16 nodes,level 5=13 nodes,level 6=4 nodes,level 7=3 nodes). Correct me if I am wrong.

▲ 1 vote ● Reply ● Message ● Permalink

**Raj S** Edited 2 years ago

Hi Kiran, Level 1 = 1 node , level 2 = 3nodes, level 3 = 4 nodes, Level 4 = 2 nodes, level5 = 5 nodes, level 6 = 4 nodes.

▲ 0 votes ● Reply ● Message ● Permalink

**Krishna Bisht** 2 years ago

but input1 has total 50 nodes. in your case sum of the nodes is not 50.

▲ 0 votes ● Reply ● Message ● Permalink

**Raj S** 2 years ago

got it!! my mistake :). I explained sample input. Very naive of me.

?

0 votes ● Reply ● Message ● Permalink

**Krishna Bisht** 2 years ago

and i think sample input has only 5 levels with level 1=1 node(11), level 2= 3 nodes(17,1,6), level 3=4 nodes(15,5,2,8), level 4= 8 nodes(4,9,12,16,18,13,7,14), level 5=4 nodes(10,19,20,3)?? (sum of nodes is not 20 in your answer)

0 votes ● Reply ● Message ● Permalink

**Raj S** 2 years ago

the graph i got was Level 1 = node 1, level 2 = nodes (11, 2, 8), level3 = nodes (17, 6, 7, 14), level4 = nodes(5, 15), level5 = nodes(13, 16, 18. 9, 4, 12), level6 = nodes(3, 20, 19, 10).

0 votes ● Reply ● Message ● Permalink

**Krishna Bisht** 2 years ago

okay thanks! :)

0 votes ● Reply ● Message ● Permalink

**Raj S** Edited 2 years ago

It did work. I was thinking root node is at level 0. But re-read the problem and found that root node is at level 1.

1 vote ● Reply ● Message ● Permalink

**Vivek Sunderraj** a year ago

```
#include <iostream>
#include<queue>
#include<vector>
using namespace std;
int main()
{
int n,x,y,i,j,l;
int s,vis[100001]={0},level[100001]={0},count=0;
queue <int> q;
vector <int> v[100001];
cin>>n;
for(i=0;i<n-1;i++)
{
cin>>x>>y;
if(i==0)
s=x;
v[x].push_back(y);
v[y].push_back(x);
}
cin>>l;
vis[s]=1;
level[s]=1;
q.push(s);
while(!q.empty())
{
int p=q.front();
q.pop();

for(i=0;i<v[p].size();i++)
{
if(vis[v[p][i]]==0)
{
q.push(v[p][i]);
vis[v[p][i]]=1;
level[v[p][i]]=level[p]+1;
```

```
}
}

}

for(i=0;i<n;i++)
{
if(level[i]==l)
count++;
}
cout<<count;

return 0;
}
```

WHAT IS WRONG IN THIS CODE??

▲ 1 vote ● Reply ● Message ● Permalink

**Ayushman Dey** a year ago

Bro it should be a directed graph, you're doing this " v[x].push_back(y); v[y].push_back(x); " which is done when it's an undirected graph. Maybe that is causing problems.

▲ 0 votes ● Reply ● Message ● Permalink

**Jitendra Kumar Nagar** a year ago

wrong bro...it is undirected graph he is doing right...u r mistaken..read the problem..

▲ 0 votes ● Reply ● Message ● Permalink

**Arvind Rajpurohit** a year ago

your code will work when the nodes are starting from 0 but in this case numbering of nodes are starting from 1.
so just change v[x].push_back(y) to [x-1].push_back(y-1) and v[y].push_back(x) to v[y-1].push_back(x-1) .
Now your should be working!!

▲ 0 votes ● Reply ● Message ● Permalink

**Jitendra Kumar Nagar** a year ago

Bro Index ==1 is root not first input...it is fixed...choose x==1..try it out

▲ 0 votes ● Reply ● Message ● Permalink

**Kaushal Shah** a year ago

What is the error here???

```
import java.io.*;
import java.util.*;
@SuppressWarnings("unchecked")
public class BreadthFirstSearch{
int V;
LinkedList<Integer> adj[];
BreadthFirstSearch(int v){
V = v;
adj = new LinkedList[v];
for(int i=0;i<v;i++){
adj[i] = new LinkedList<>();
}
}
public void addEdge(int v, int w){
adj[v].add(w);
}
public void BFS(int s, int l){
boolean visited[] = new boolean[V];
```

?

```
LinkedList<Integer> queue = new LinkedList<>();
queue.add(s);
visited[s] = true;
int level[] = new int[V];
level[s] = 1;
while(queue.size()!=0){
int current = queue.poll();
Iterator<Integer> i = adj[current].listIterator();
while(i.hasNext()){
int n = i.next();
if(!visited[n]){
queue.add(n);
level[n] = level[current]+1;
visited[n] = true;
}
}
}
int count = 0;
for(int i=0;i<V;i++){
if(level[i] == l){
++count;
}
}
System.out.println(count);
}
public static void main(String args[]){
Scanner in = new Scanner(System.in);
int n = in.nextInt();
BreadthFirstSearch bsf = new BreadthFirstSearch(n);
for(int i=0;i<n-1;i++){
bsf.addEdge(in.nextInt()-1,in.nextInt()-1);
}
int level = in.nextInt();
bsf.BFS(0,level);
}
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Jitendra Singh** a year ago

for(int i=0;i<n-1;i++) why you put n-1??
the error is you are taking input one less.

▲ 0 votes ● Reply ● Message ● Permalink

**Kaushal Shah** a year ago

no! if there are N nodes, there will be N-1 Edges.

▲ 0 votes ● Reply ● Message ● Permalink

**Manas Joshi** a year ago

```
for(int i=0;i<n-1;i++){
int a=in.nextInt();
int b=in.nextInt();
bsf.addEdge(a-1,b-1);
bsf.addEdge(b-1,a-1);
}
```

this was the only problem u worked as a directed graph but u should had workd as an undirected graph. this solved the problem and all ur test cases are running

▲ 1 vote ● Reply ● Message ● Permalink

?

**Kaushal Shah** a year ago

Thank you so much! I just missed that the graph is undirected! Thanks again!

0 votes ● Reply ● Message ● Permalink

**Karthikeyan Jayaraman** a year ago

its just checking the output of only one test case
i just printed the sample output answer
whether this is true
i submitted dfs using correct logic still its showing sample test case passed is it checking only one test case

1 vote ● Reply ● Message ● Permalink

**Sumit Saurav** 10 months ago

```
//Enjoy this also ....
#include<bits/stdc++.h>
using namespace std;
int main()
{
int t,i,j,k,l;
vector<int> v[100000<<1];
queue<int> q;
int n;
cin>>n;
for(i=0;i<n-1;i++)
{
cin>>k>>l;
v[k].push_back(l);
v[l].push_back(k);
}
q.push(1);


map<int,int>visited;
//stack<int>s;
visited[1]=1;
while(!q.empty())
{k=q.front();
q.pop();
//cout<<visited[k]<<" ";
for(i=0;i<v[k].size();i++)
{ if(visited[v[k][i]]==0)
{//cout<<"hello"<<<<" ";
visited[v[k][i]]=visited[k]+1;q.push(v[k][i]);}
}

}
int x;
cin>>x;
int count=0;
for(i=1;i<=n;i++)
if(visited[i]==x)
count++;
cout<<count<<endl;
}
```

1 vote ● Reply ● Message ● Permalink

**UdayNbausj** 3 months ago

MY CPP SOL

```
#include<iostream>
#include<vector>
```

?

```
#include<algorithm>
#include<queue>
#include<stack>
#define ll long long
using namespace std;
vector<ll>adj[100001];
bool vis[100001]={0};
ll level[100001];
ll query[100001];
//undirected graph
void addedge(vector<ll>adj[],ll n1,ll n2){
adj[n1].push_back(n2);
adj[n2].push_back(n1);
}
void bfs(ll node_data){
queue<ll>q;
q.push(node_data);
level[node_data] = 1;
query[node_data] = 1;
//cout<<node_data<<endl;
vis[node_data] = true;
ll count = 1;
ll same = 0;
while(!q.empty()){
ll val = q.front();
q.pop();
count+=1;
same = 0;
for(ll i=0;i<adj[val].size();i++){
if(vis[adj[val][i]]==false){
same+=1;
q.push(adj[val][i]);
level[adj[val][i]] = level[val]+1;
query[count] = same;
//cout<<adj[val][i]<<" ";
vis[adj[val][i]] = true;
}
}
//cout<<same<<endl;
//cout<<endl;
}
}
int main(){
ll nodes,node1,node2,q;
cin>>nodes;
nodes = nodes-1;
while(nodes--){
cin>>node1>>node2;
addedge(adj,node1,node2);
}
bfs(1);
cin>>q;
cout<<query[q]<<endl;
}
```

▲ 1 vote ● Reply ● Message ● Permalink

**Mohd Ejaz Siddiqui** 2 months ago

Solved this problem here using python 3.6 - https://github.com/mejaz/DS_Algos/blob/master/bfs_noOfElementsInALevel.py

?

▲ 1 vote ● Reply ● Message ● Permalink

**Ketul Shah** a month ago

My solution for the given problem:-

```
#include <bits/stdc++.h>
#define F(i,a,b) for(int i = (int) a ; i < (int)b ; ++i)
#define N 100000
using namespace std;
vector <int> adj[N];
int cnt=0;
int level[N];
bool visited[N];
void BFS(int s, int x){
level[s] = 1;
queue <int> q;
q.push(s);
visited[s] = true;
while(!q.empty()){
int p = q.front();
q.pop();
F(i,0,adj[p].size()){
if(visited[adj[p][i]] == false){
level[adj[p][i]] = level[p] + 1;
if(level[adj[p][i]] == x){
cnt ++;
//cout << "yo";
}
q.push(adj[p][i]);
visited[adj[p][i]] = true;
}
}
}

}
int main(){
int x,y,nodes,l;
cin >> nodes;
F(i,0,nodes-1){
cin >> x >> y;
adj[x].push_back(y);
adj[y].push_back(x);
}
cin >> l;
BFS(1,l);
cout << cnt << endl;
return 0;
}
```

▲ 1 vote ● Reply ● Message ● Permalink

**Rushiraj Baxi** 20 days ago

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n;
cin>>n;
vector<int >g[n+1];
for(int i=0;i<n;i++)
{
int s,e;
```

?

```
cin>>s>>e;
g[s].push_back(e);
g[e].push_back(s);
}
int anj;
cin>>anj;
int l[n+1];
int v[n+1];
for(int i=0;i<=n;i++)
{
l[i]=1;
v[i]=0;
}
list<int >q;
q.push_back(1);
while(q.size()>0)
{
int t=q.front();
q.pop_front();
for(int i=0;i<g[t].size();i++)
{
if(v[g[t][i]]==0)
{
l[g[t][i]]=l[t]+1;
q.push_back(g[t][i]);
}
}
v[t]=1;
}
int ans=0;
for(int i=1;i<n+1;i++)
{
if(l[i]==anj)
ans++;
}
cout<<ans;
}
```

▲ 1 vote ● Reply ● Message ● Permalink

**Abhijeet Singh** Edited 2 years ago

in 0-1 bfs, how will the loop { while(!q.empty()) } end?? , isnt it an infinite loop?

▲ 0 votes ● Reply ● Message ● Permalink

**jayant isswani** a year ago

In the last stage of the algorithm, all the vertices would be present in the deque. And, no vertex's neighbours would satisfy the 'if' condition. Thus, no new vertices would be added to the deque and all the vertices would pop one-by-one from the deque. In this way, the deque would become empty and the control would come out of the loop.

▲ 0 votes ● Reply ● Message ● Permalink

**Nitin Rathodiya** a year ago

well in my opinion there was a slight error in explanation.they must maintain a visited array in order to keep track of visited nodes and no visited node can be pushed to queue repeatedly

▲ 0 votes ● Reply ● Message ● Permalink

**Utsav Goel** 2 years ago

```
#include <bits/stdc++.h>
using namespace std;
vector<int> adj[100002];
```

?

```
bool vis[100002]={false};
int level[100002]={0},c=0,lev;
void bfs(int s)
{ vis[s]=true;
queue<int> q;
q.push(s);
level[s]=0;
while(!q.empty())
{ int p=q.front();
q.pop();
for(int i=0;i<adj[p].size();i++)
{ if(vis[adj[p][i]]==false)
{ vis[adj[p][i]]=true;
q.push(adj[p][i]);
level[adj[p][i]]=level[p]+1;
}
}
}
}
int main()
{
int n,a,b;
cin>>n;
for(int i=0;i<n-1;i++)
{ cin>>a>>b;
a--;b--;
adj[a].push_back(b);
adj[b].push_back(a);
}
cin>>lev;
for(int i=0;i<n;i++)
bfs(i);
int c=0;
for(int i=0;i<100005;i++)
{ if(level[i]==lev)
c++;
}
cout<<c<<endl;
return 0;
}
```

what's wrong with this code?

▲ 0 votes ● Reply ● Message ● Permalink

**Manish PERIWAL** 2 years ago

for(int i=0;i<n;i++)
bfs(i);

this code set every vertex level to 0

▲ 0 votes ● Reply ● Message ● Permalink

**Prathmesh Bhadekar** a year ago

What is stored in adj[][] matrix ,only 0 or 1?? ,I don't understand what will vis[adj[p][i]] give ,it will always be 0 or 1 ,also i'm confused with level[adj[p][i]] and level[p] ,can you explain it a bit better?

▲ 0 votes ● Reply ● Message ● Permalink

**Rohit Kumar** Edited a year ago

which node should be the root

▲ 0 votes ● Reply ● Message ● Permalink

?

**Aditya Nand** 8 months ago

1

0 votes ● Reply ● Message ● Permalink

**Bako Binfa Nimmyel** a year ago

I wanted writing a program that execute the item sale in a shop,but am having problems;who can be of help to me.Please

0 votes ● Reply ● Message ● Permalink

**Mohammad Zaid** a year ago

```
#include<bits/stdc++.h>
using namespace std;
vector<int> vec[100001];
queue<int> qu;
int main()
{
int N,tem;
scanf("%d",&N);
int visit[99991]={0},level[100000];
queue<int> qu;
for(int A=0;A<N-1;A++)
{
int X,Y;
scanf("%d %d",&X,&Y);
vec[X].push_back(Y);
vec[Y].push_back(X);
}
qu.push(1);
int fre[99999]={0};
level[1]=1;
visit[1]=1;
fre[1]++;
while(!qu.empty())
{
tem=qu.front();
// cout<<tem<<endl;
qu.pop();
for(int B=0;B<vec[tem].size();B++)
{
if(visit[vec[tem][B]]==0)
{
visit[vec[tem][B]]=1;
// cout<<vec[tem][B]<<endl;
qu.push(vec[tem][B]);
level[vec[tem][B]]=level[tem]+1;
fre[level[vec[tem][B]]]++;
}
}
}
int level_to;
scanf("%d",&level_to);
printf("%d",fre[level_to]);
}
```

0 votes ● Reply ● Message ● Permalink

**Honoya** Edited a year ago

```
#include <iostream>
#include <vector>
#include<queue>
```

?

```
using namespace std;
int bfs(int level);
vector<int> OurNodes[500];
int levels[500] = { 0 };
bool visited[500] = { 0 };
int bfs(int level) {
queue<int> q;
q.push(1);
levels[1] = 1;
visited[1] = 1;
int nuMNodes=0;
while (!q.empty())
{
int p = q.front();
q.pop();
for (int i = 0; i < OurNodes[p].size(); i++)
{
if (visited[OurNodes[p][i]]==0)
{
if (level == levels[p] + 1)
nuMNodes++;
levels[OurNodes[p][i]] = levels[p] + 1;
q.push(OurNodes[p][i]);
visited[OurNodes[p][i]] = 1 ;
}
}
}
return nuMNodes;
}
int main()
{
int Nnodes;
cin >> Nnodes;
int a, b;
for (int i = 0; i < Nnodes - 1; i++)
{
cin >> a >> b;
OurNodes[a].push_back(b);
OurNodes[b].push_back(a);
}
int level;
cin >> level;
int numNodesAtLevel;
numNodesAtLevel=bfs(level);
cout << numNodesAtLevel << endl;
return 1;
}
```

Why I got Runtime Error - NZEC?
Thanksin advance

▲ 0 votes ● Reply ● Message ● Permalink

**Vikram Kumar** a year ago

```
#include <bits/stdc++.h>
using namespace std;
void bfs(vector<int> v[],int s,int n)
{
bool vis[n+1];
int level[n+1];
for(int i=1;i<=n;i++){
```

?

```
vis[i]=0;level[i]=1;}
queue<int> q;
q.push(s);
vis[1]=1;

while(!q.empty())
{
int p=q.front();
q.pop();

for(int i=0;i<v[p].size();i++)
{
if(vis[v[p][i]]==0){
q.push(v[p][i]);
vis[v[p][i]]=1;
level[v[p][i]]=level[p]+1;
}

}
}
int count=0;
for(int i=1;i<=n;i++)
{
if(level[i]==s)
count++;
}
cout<<count<<'\n';
}
int main()
{
vector<int> v[100005];
int n,x,y;
cin>>n;
for(int i=0;i<n-1;i++)
{
cin>>x>>y;
v[x].push_back(y);
}
int s;
cin>>s;
bfs(v,s,n);
return 0;
}
```

what's wrong with this code

▲ 0 votes ● Reply ● Message ● Permalink

**Aditya Nand** 8 months ago

Brother, in the question it is given that the graph is undirected. But you, in your code, treated it as directed graph.
Add another line in your input loop :-
v[y].push_back(x)
Otherwise you are good to go...!

▲ 0 votes ● Reply ● Message ● Permalink

**Tanbir Hossain Talat** a year ago

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int x,y,i,j,k,l,n,s,a[10009]={0},count=0;
```

?

```
vector<int>v[1000];
cin>>n;
for(i=0;i<n-1;i++)
{
cin>>x>>y;
v[x].push_back(y);
v[y].push_back(x);
}
scanf("%d",&l);
queue<int>q;
int level[n];
bool vis[n];
for(i=0;i<n-1;i++)
{level[i]=0,vis[i]=false;}
level[1]=1;
vis[1]=1;
a[1]++;
q.push(1);
while(!q.empty())
{
k=q.front();
q.pop();
for(i=0;i<v[k].size();i++)
{
if(vis[v[k][i]]==0)
{
level[v[k][i]]=level[k]+1;
q.push(v[k][i]);
vis[v[k][i]]=1;
a[level[v[k][i]]]++;
}
}
}
cout<<a[l]<<endl;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Daredevil** a year ago

In function `_GLOBAL__sub_I_ar':
(.text.startup+0x165): relocation truncated to fit: R_X86_64_32 against `.bss'
(.text.startup+0x174): relocation truncated to fit: R_X86_64_32 against `.bss'
error: ld returned 1 exit status

What does this error mean?

▲ 0 votes ● Reply ● Message ● Permalink

**Sheel Nidhi Pandey** Edited a year ago

Can anybody look at this code : this is not able to pass the test cases :- https://code.hackerearth.com/26bb8aO?key=2573bd6e664fdf6b606cfb2c5412f6c1 I am trying to use adjancyList that is not working if we have not direct connection to elements like 5 1.. then 11 5 ,11 14....here 1's adjacency list is 0 while 5 have 5->1->11->14,.

▲ 0 votes ● Reply ● Message ● Permalink

**Jitendra Singh** a year ago

are you talking about directed or undirected graph?

▲ 0 votes ● Reply ● Message ● Permalink

**Ishu Dohare** a year ago

?

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <string>
#include <list>
#include <set>
#include <iterator>
#include <queue>
#include <bits/stdc++.h>
#define endl '\n'
using namespace std;
//ALCATRAZIAN_CODE
//********whosoever_is_reading_this_go_fuck_urself*********
void solve()
{int t,i,j,n,x,y;
cin>>n;
vector<int> v[n+1];
j=n-1;
while(j-->0)
{
cin>>x>>y;
v[x].push_back(y);
v[y].push_back(x);
}
cin>>x;
int level[n];
bool vis[n];
for(i=0;i<n;i++)
{level[i]=0;
vis[i]=0;
}
queue<int>q;
int s=1;
q.push(s);
level[s]=1;
vis[s]=true;
while(!q.empty())
{
int p = q.front();
q.pop();
for( i = 0;i < v[ p ].size() ; i++)
{
if(vis[ v[ p ][ i ] ] == false)
{
//Setting the level of each node with an increment in the level of parent node
level[ v[ p ][ i ] ] = level[ p ]+1;
q.push(v[ p ][ i ]);
vis[ v[ p ][ i ] ] = true;
}
}
}

int sum=0;
for(i=1;i<=n;i++)
if(level[i]==x)
sum++;
cout<<sum;
}
```

```
int main()
{
ios_base::sync_with_stdio(false);
cin.tie(NULL);
solve();
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Gaurav Tolani** a year ago

are you considering the disconnected graph's case?

0 votes ● Reply ● Message ● Permalink

**Priyanshu Varshney** a year ago

This code is working:)
https://code.hackerearth.com/d465ddX

0 votes ● Reply ● Message ● Permalink

**prakhar kumar** a year ago

I did the same thing but the value of x is being entered as 1 instead of 2 -_-

0 votes ● Reply ● Message ● Permalink

**prakhar kumar** a year ago

The hell is with this question sample input gives the value of x as 2 but when i run the code it somehow shows the value of x as 1 but it was supposed to give value of 2 -_-

0 votes ● Reply ● Message ● Permalink

**Saurabh Sharma** a year ago

Finally i solved this problem!!
It feels heavenly...

```
#include <iostream>
#include <queue>
#include <vector>
#include <algorithm>
using namespace std;
int nodes;
void bfs(vector<vector<int> >& g,vector<int>& l ){
vector<bool>visited(nodes, false);
queue<int>q;
q.push(0);l[0]=0;visited[0]=true;
while(!q.empty()){
int cache = q.front();q.pop();
for(auto it:g[cache]){
if(!visited[it]){
q.push(it);
l[it]=l[cache]+1;
visited[it]=true;
}
}
}
}
int main(){
cin>>nodes;
vector<vector<int> >g(nodes);vector<int>level(nodes);int temp = nodes;
while(temp--){
int a, b;cin>>a>>b;
g[a-1].push_back(b-1);
g[b-1].push_back(a-1);
}
```

?

```
bfs(g, level);
int x;cin>>x;
int ans = count_if(level.begin(), level.end(), [x](int a){
return a==x;
});
cout<<ans;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Sumit Kumar** Edited a year ago

my code is not taking the value of x . i'm using c++. here's my code.

```
#include<bits/stdc++.h>
#include<stdio.h>
using namespace std;
// This class represents a undirected graph using adjacency list representation
class Graph
{
int V; // No. of vertices
list<int> *adj; // Pointer to an array containing adjacency lists

public:

Graph(int V); // Constructor
int a[100001];
void addEdge(int v, int w); // function to add an edge to graph
int BFS(int s,int x); // prints BFS traversal from a given source s
};
Graph::Graph(int V)
{
this->V = V;
adj = new list<int>[V];
int a[100001]={0};
}
void Graph::addEdge(int v, int w)
{
adj[v].push_back(w); // Add w to v's list.
}
int Graph::BFS(int s,int x)
{
// Mark all the vertices as not visited
bool *visited = new bool[V];
for(int i = 0; i < V; i++)
visited[i] = false;

// Create a queue for BFS
list<int> queue;
// Mark the current node as visited and enqueue it
visited[s] = true;
queue.push_back(s);
a[s]=1;
// 'i' will be used to get all adjacent vertices of a vertex
list<int>::iterator i;
while(!queue.empty())
{
// Dequeue a vertex from queue and print it
s = queue.front();
// cout << s << " ";
queue.pop_front();
```

```
// Get all adjacent vertices of the dequeued vertex s
// If a adjacent has not been visited, then mark it visited
// and enqueue it
for(i = adj[s].begin(); i != adj[s].end(); ++i)
{
if(!visited[*i])
{
a[*i]=a[s]+1;
visited[*i] = true;
queue.push_back(*i);
}
}
}
int c=0;
// cout<<x;
for(int i=0;i<V;i++) cout<<a[i]<<" ";
for(int i=1;i<V;i++)
{
if(x==a[i])
c++;
}
return c;
}
// Driver program to test methods of graph class
int main()
{
int nodes;
cin>>nodes;
// Create a graph given in the above diagram
Graph g(nodes+1);
int a,b,alpha;
for(int i=0;i<nodes;i++)
{
cin>>a>>b;
g.addEdge(a,b);
g.addEdge(b,a);
}
cin>>alpha;
cout<<alpha;
// cout << "Following is Breadth First Traversal "
// << "(starting from vertex 2) \n";
int p=g.BFS(1,);
// cout<<p;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Manish Sharma** a year ago

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n;
cin>>n;
vector<int> a[100000];
for(int i=1;i<n;i++)
{
int x,y;
cin>>x>>y;
a[x].push_back(y);
```

?

```
a[y].push_back(x);
}
vector<int>l(n+1);
vector<bool>v(n+1);
queue <int> q;
q.push(1);
l[1]=1;
v[1]=true;
while(!q.empty())
{
int temp=q.front();
for(int i=0;i<a[temp].size();i++)
{
if(v[a[temp][i]]==false)
{
q.push(a[temp][i]);
l[a[temp][i]]=l[temp]+1;
v[a[temp][i]]=true;
}
}
q.pop();
}
int x;
cin>>x;
cout<<count(l.begin(),l.end(),x);
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Sagnik Chaudhuri** a year ago

```
#include <bits/stdc++.h>
using namespace std;
vector <int> adj[100001];
bool visited[100001];
int county = 0;
int level[100001];
void ini()
{
for(int i = 0; i <= 100001; i++)
level[i] = 0;
}
int x = 0;
void bfs(int s)
{
queue <int> q;
q.push(s);
level[s] = 1;
visited[s] = true;
while(!q.empty())
{
int p = q.front();
q.pop();
for(unsigned int i = 0; i < adj[p].size(); i++)
{
if(!visited[adj[p][i]])
{
level[adj[p][i]] = level[p] + 1;
q.push(adj[p][i]);
visited[adj[p][i]] = true;
//if(level[adj[p][i]] == x)
```

?

```
// county ++;
}
}
}
}
int main()
{
int n,e;
cin>>n;
e = n - 1;
for(int i = 1; i <= e; i++)
{
int a,y;
cin>>a>>y;
adj[a].push_back(y);
adj[y].push_back(a);
}
cin>>x;
bfs(1);
for(int i = 1; i <= n; i++)
if(level[i] == x)
county ++;
cout<<county;
//cout << "Hello World!" << endl;
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Harshit Singh** a year ago

Check till kth level...AC

```
#include <bits/stdc++.h>
using namespace std;
int lev[100000];
vector<int> edg[1000000];
int main()
{
int n,x,k,y,c=0;
cin>>n;
for(int i=1;i<n;i++)
{
cin>>x>>y;
edg[x].push_back(y);
edg[y].push_back(x);
}
cin>>k;
bool vis[n+1];
for(int i=1;i<=n;i++)
{
vis[i]=false;
lev[i]=0;
}
queue<int> q;
q.push(1);
vis[1]=true;
lev[1]=1;
int i;
while(!q.empty())
{
int f=q.front();
q.pop();
```

```
for(i=0;i<edg[f].size();i++)
{
if(vis[edg[f][i]]==false)
{
vis[edg[f][i]]=true;
q.push(edg[f][i]);
lev[edg[f][i]]=lev[f]+1;
if(lev[edg[f][i]]==k)
c++;
}
}
if(lev[edg[f][i]]==k+1)
break;
}
cout<<c;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Ish Kool** a year ago

see the level of my stupidity, wrote the code ,but added the pop and front functions at the end of the loop instead of start.So the program ran successfully for the nodes which had more than 1 children but gave wrong answers for 1 children.
RESULT: full 3 hours spent on this stupid debugging (:
correct code:) C++

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
int n,i,j,k,a,b,t=0,x;
scanf("%d",&n);

vector <int> v[n];
int level[n];
bool visit[n];
queue <int> q;

for(i=0;i<n-1;i++)
scanf("%d %d",&a,&b),v[a-1].push_back(b-1),v[b-1].push_back(a-1);

scanf("%d",&x);x--;

for(i=0;i<n;i++)
visit[i]=false,level[i]=0;

//level[0]=0;

q.push(0);
a=q.front();
visit[a]=true;

while(!q.empty())
{
a=q.front();
q.pop();
if(!v[a].empty())
{
vector<int> :: iterator p;

for(p=v[a].begin();p!=v[a].end();++p)
```

?

```
{
if(visit[*p]!=true)
{
q.push(*p);visit[*p]=true;
level[*p]=level[a]+1;
if(level[*p]==x)
t++;

// printf("a=%d p=%d t=%d\n",a,*p,t);
}

}
}


}

printf("%d\n",t);

}
```

0 votes ● Reply ● Message ● Permalink

**jayant isswani** a year ago

Why can't boolean array be used? What would be the error caused?

0 votes ● Reply ● Message ● Permalink

**Kunal Gupta** a year ago

Working Code:

```
#include <iostream>
#include <queue>
#include <vector>
using namespace std;
int main()
{
int n;
cin>>n;
vector <int> G[100001];
bool vis[100001];
for(int i=1;i<=n;i++)
{
vis[i] = false;
}

for(int i=0;i<n-1;i++)
{
int a,b;
cin>>a>>b;
G[a].push_back(b);
G[b].push_back(a);
}

int x;
cin>>x;

int L[100001]={0};

queue <int> Q;
Q.push(1);
L[1] = 1;
```

```
while( !Q.empty() )
{
int e = Q.front();
Q.pop();
vis[e] = true;
for(unsigned int i=0;i<G[e].size();i++)
{
if(!vis[G[e][i]])
{
Q.push(G[e][i]);
L[G[e][i]] = L[e]+1;

}
}
}
int ans = 0;
for(int i=1;i<=n;i++)
{
if(L[i]==x)
{
ans++;
}
}
cout<<ans;



return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Akhilesh Soni** a year ago

```
#include<bits/stdc++.h>
using namespace std;
typedef long long int ll;
ll level[100002],vis[100002],q,ans=0;
void dfs(vector<list<ll> >&G,ll s)
{
vis[s]=1;
for(auto it=G[s].begin();it!=G[s].end();it++)
{
if(!vis[*it])
{
level[*it]=level[s]+1;
if(level[*it]==q)
ans++;
dfs(G,*it);
}
}
}
int main()
{
ios::sync_with_stdio(false);
ll i,V,E;
cin >> V ;
E=V-1;
vector< list<ll> >G(V+1);
for(i=0;i<=V;i++)
{
```

```
G[i].clear();
level[i]=1;
vis[i]=0;
}
for(i=0;i<E;i++)
{
ll u,v;
cin >> u >> v;
G[u].push_back(v);
G[v].push_back(u);
}
cin >> q;
dfs(G,1);
if(q==1)
ans++;
cout << ans << "\n";
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**HIMADRI ANJOY** a year ago

simple follow the hackerearth tutorial only

0 votes ● Reply ● Message ● Permalink

**Shubham Singh Manhas** a year ago

C++

```
#include <iostream>
#include <vector>
#include <queue>
using namespace std;
int main()
{
int n,x;
cin >> n;
vector< vector<int> > v(n);
bool vis[n]={false};
for(int i=0;i<n-1;i++){
int a,b;
cin >> a >> b;
v[a-1].push_back(b-1);
v[b-1].push_back(a-1);
}
cin >> x;
int level=1;
queue<int> q;
q.push(0);
vis[0]=true;
int temp=0;
while(!q.empty()){
int k=q.size();
if(x==level){
temp=k;
break;
}
while(k>0){
int p=q.front();
q.pop();
for(int i=0;i<v[p].size();i++){
if(vis[v[p][i]]==false){
q.push(v[p][i]);
```

```
vis[v[p][i]]=true;
}
}
k--;
}
level++;
}
cout << temp;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Raghav Ravi Prakash** a year ago

This question is ambiguous. They mentioned the VALUE at the index 1 to be the root node but what is this VALUE? It could be any node from the adjacency list/matrix, i.e. we could keep any value at index 1 and claim it to be the root node.

▲ 0 votes ● Reply ● Message ● Permalink

**Siddhant Chhabra** a year ago

Hey.
How do I implement this in Cpp.
Here, edges[ v ] [ i ] is an adjacency list that exists in the form of pairs i.e. edges[ v ][ i ].first will contain the node to which v is connected and edges[ v ][ i ].second will contain the distance between v and edges[ v ][ i ].first.

▲ 0 votes ● Reply ● Message ● Permalink

**Siddhant Chhabra** a year ago

Nevermind. I figured it out.

▲ 0 votes ● Reply ● Message ● Permalink

**Shivam kumar** a year ago

The tree seems Undirected!!
My Solution gives wrong ans on directed tree and correct answer on undirected tree.

▲ 0 votes ● Reply ● Message ● Permalink

**Kumar Siddharth** a year ago

```
#include<bits/stdc++.h>
using namespace std;
bool vis[1000000];
vector<vector<int> > graph;
queue<int> q;
int level[1000000];
map<int,int> mp;
void bfs(int s){
int i,x;
q.push(s);
level[s]=1;
mp[level[s]]++;
vis[s]=true;
while(!q.empty()){
x=q.front();
q.pop();
for(i=0;i<graph[x].size();i++)
{
if(vis[graph[x][i]]==false)
{
q.push(graph[x][i]);
vis[graph[x][i]]=true;
level[graph[x][i]]=level[x]+1;
```

?

```
//cnt[level[graph[x][i]]]++;
}
}
}
}
int main(){
int n,x,i,a,b,y;
cin>>n;
graph.resize(n+10);
for(i=1;i<n;i++)
{
cin>>a>>b;
graph[a].push_back(b);
graph[b].push_back(a);
}
cin>>x;
bfs(1);
for(i=1;i<=n;i++)
{
mp[level[i]]++;
}
cout<<mp[x];
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Jitendra Kumar Nagar** Edited a year ago

```
class Graph{
int V;
list<int> *tree;
bool *visited;
public:
Graph( int v ){
this->V = v;
tree = new list<int>[1000];
visited = (bool *) malloc ( sizeof(bool) * (V+1) );
for( int i=0; i<=V; i++ )
{
visited[i]=false;
}
}
void addEdge ( int src , int dest ){
tree[ src ].push_back( dest );
tree[dest].push_back( src );
}
void BFS(int source , int x)
{
if(x==1)
{ cout<<x<<endl;
return ;}
deque<int> queue;
int level=1;
queue.push_back(source);
visited[source]=true;

while( !queue.empty() )
{
int count=0;
int curr_node=queue.front();
queue.pop_front();
```

?

```
list<int>::iterator it;
for(it=tree[curr_node].begin();it!=tree[curr_node].end();it++)
{
if( visited[ *it ]== false )
{
count++;
visited[*it]=true;
queue.push_back(*it);
}
}
level++;
if (level== x )
{
cout<<count<<endl;
return ;
}
}
}
};
int main()
{
int n;
cin>>n;
Graph graph=Graph(n);
for(int i=0;i<n-1;i++)
{
int src,dest;
cin>>src>>dest;
graph.addEdge(src,dest);
}
int x;
cin>>x;
graph.BFS(1,x);
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Jason Whisnant** a year ago

Lots of work, but here is my code (Python 3):

```
"""
Created on Mon Aug 7 09:59:13 2017
@author: whisnaja
"""
N = int(input())
vis = []
for i in range(N):
vis.append(0)
queue = []
nodes =[]
levels = [1]
for i in range(N-1):
nodes.append(list(map(int,input().split(' '))))
levels.append(0)

x = int(input())
level = 1
queue.append(1)
vis[0] = 1
```

?

```
while queue:
q = queue.pop(0)

for i in range(len(nodes)):
if q in nodes[i]:
if nodes[i][0] != q:
nodes[i] = nodes[i][::-1]
if not vis[nodes[i][1]-1]:
queue.append(nodes[i][1])
vis[nodes[i][1]-1] = 1
levels[nodes[i][1]-1] = levels[nodes[i][0]-1]+1

level += 1
count = 0
for i in range(len(levels)):
if levels[i] == x:
count += 1

print(count)
```

▲ 0 votes ● Reply ● Message ● Permalink

**Anshik Nigam** a year ago

please, someone, help I am getting correct output in devc++ but here for test-case 1 and 2 it is showing incorrect output. here is my code.

```
#include <iostream>
#include<vector>
#include<queue>
using namespace std;
int main()
{
int n,a,b,x;
cin>>n;
vector<int> adj[1000];
bool vis[n];
int level[1000];
queue <int> que;
for(int i=1;i<=n-1;i++)
{
cin>>a>>b;
adj[a].push_back(b);
adj[b].push_back(a);
level[i] = 0;
}
cin>>x;
que.push(1);
vis[1] = true;
level[1] = 1;
/*
for(int i=1;i<=n;++i)
{
cout<<"Adjacency list of node "<<i<<" : ";
for(int j=0;j<adj[i].size();++j)
{
if(j == adj[i].size()-1)
cout<<adj[i][j]<<endl;
else
cout<<adj[i][j]<<" --> ";
}
}
```

?

```
*/
while(!que.empty())
{
int v = que.front();
que.pop();
//cout<<"level of "<<v<<" "<<level[v]<<endl;
for(int i=0;i<adj[v].size();i++)
{
if(vis[adj[v][i]] == false)
{
level[adj[v][i]] = level[v]+1;
que.push(adj[v][i]);
vis[adj[v][i]] = true;
}
}
}
int count = 0;

for(int i=1;i<=n;i++)
{
//cout<<level[i]<<" ";
if(level[i] == x)
count++;
}
cout<<count;

return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Mayank Kathuria** a year ago

C++ Solution:

```
#include<bits/stdc++.h>
using namespace std;
typedef long long int ll;
typedef long int l;
ll level[100005]={0};
vector< vector<ll > > v(100005);
void bfs(ll s) //s is the root node here
{
queue<ll>q;
level[s]=1;
q.push(s);
while(!q.empty())
{
ll ele=q.front();
for(ll i=0; i<v[ele].size(); i++)
{
if(level[ v[ele][i] ]==0)
{
q.push(v[ele][i]);
level[ v[ele][i] ]= 1 + level[ele];
}
}
q.pop();
}
}
int main()
{
#ifndef ONLINE_JUDGE
```

?

```
freopen("input.txt", "r", stdin);
freopen("output.txt", "w", stdout);
#endif
ll n,a,b,x,i=0;
cin>>n;
for( i=0; i<n-1; i++)
{
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
cin>>x;
bfs(1);
ll ctr=0;
for(ll i=1; i<=n-1; i++)
{

if(level[i]==x)
{
ctr++;
}
}
cout<<ctr<<endl;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Rohit Kumar** a year ago

In 0-1 BFS,
when every node is processed then why the edge[v][i].first =1 is pushed back and the other one pushed in front.

▲ 0 votes ● Reply ● Message ● Permalink

**Rajat Bansal** a year ago

Because it is not ford fulkerson algorithm as you are thinking!!
I think there should be a stoppind condition that the loop will be over when all nodes are popped out once.
Then you will get why it is important to check 0 and 1 before pushing in the queue and push accordingly

▲ 0 votes ● Reply ● Message ● Permalink

**Rajat Bansal** a year ago

In 0-1 bfs what will be the stopping condition as Q.empty() will not work bcz I think queue will never get empty as there is no checking condition for putting an element in the queue

▲ 0 votes ● Reply ● Message ● Permalink

**Harish Raghav** a year ago

my working solution & can be optimized.

```
#include <iostream>
#include <vector>
#include <queue>
using namespace std;
int BFS();
int main()
{
int nNumOfNodes = BFS();
cout<<nNumOfNodes;
return 0;
}
```

?

```
struct adjList
{
int level;
};
struct valPair
{
int nPairVal1;
int nPairVal2;
};
int BFS()
{
vector <valPair> valPairData;
int nTotNodes, nVertex1, nVertex2, nLevelRet;
nTotNodes=nVertex1=nVertex2=nLevelRet= 0;
cin>>nTotNodes;
vector <adjList> adjListData(nTotNodes);
int nLeveCount[nTotNodes] = {0};
adjListData[0].level = 1;

std::vector<adjList>::iterator it;
valPair TempvalPair;
nLeveCount[0] = 1;
int nIndex = 0;
for(int i =0; i < (nTotNodes-1); i++)
{
cin>>nVertex1>>nVertex2;
if((nVertex1 == 1) && (nVertex2 != 1))
{
adjListData[nVertex2-1].level = 2;
nLeveCount[(adjListData[nVertex2-1].level) - 1] += 1;
}
else if((nVertex1 == 1) || (nVertex2 == 1))
{
adjListData[((nVertex1 == 1) ? nVertex2 : nVertex1)-1].level = 2;
nLeveCount[(adjListData[((nVertex1 == 1) ? nVertex2 : nVertex1)-1].level) - 1] += 1;
}
else if(adjListData[nVertex1-1].level != 0 ){
adjListData[nVertex2-1].level = adjListData[nVertex1-1].level + 1;
nLeveCount[(adjListData[nVertex2-1].level) - 1] += 1;
}
else if(adjListData[nVertex2-1].level != 0){
adjListData[nVertex1-1].level = adjListData[nVertex2-1].level + 1;
nLeveCount[(adjListData[nVertex1-1].level) - 1] += 1;
}
else{ TempvalPair.nPairVal1 = nVertex1;
TempvalPair.nPairVal2 = nVertex2;
valPairData.push_back(TempvalPair);
}
}

std::vector<valPair>::iterator it1;
nIndex = valPairData.size() - 1 ;
while(valPairData.size())
{
if(adjListData[valPairData[nIndex].nPairVal1 - 1].level != 0){
adjListData[valPairData[nIndex].nPairVal2 - 1].level = adjListData[valPairData[nIndex].nPairVal1 -
1].level+1;
nLeveCount[(adjListData[valPairData[nIndex].nPairVal2 - 1].level) - 1] += 1;
}
else if(adjListData[valPairData[nIndex].nPairVal2 - 1].level != 0){
```

```
adjListData[valPairData[nIndex].nPairVal1 - 1].level = adjListData[valPairData[nIndex].nPairVal2 -
1].level+1;
nLeveCount[(adjListData[valPairData[nIndex].nPairVal1 - 1].level) - 1] += 1;
}

if( (adjListData[valPairData[nIndex].nPairVal1 - 1].level != 0) &&
(adjListData[valPairData[nIndex].nPairVal2 - 1].level != 0) )
{
valPairData.erase(valPairData.begin()+nIndex);
}
nIndex -= 1;
nIndex = (valPairData.size() && nIndex < 0) ? valPairData.size() - 1 : nIndex;
}
cin>>nLevelRet;
return nLeveCount[nLevelRet-1];
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Gourav** a year ago

```
#include"bits/stdc++.h"
using namespace std;
const int N = 5e5+5;
vector<long long >tree[N];
vector<long long> v;
long long n;
long long x;
bool vis[N];
long long a, b;
int level;
void BFS( long long node , long long x){
queue<long long >q;
q.push(node);
while(!q.empty()){
++level;

if(level == x){
printf("%lld\n",v.size());
return ;
}

int next = q.front();
vis[next] = true;
q.pop();
v.clear();
for( int i=0 ; i<tree[next].size() ; ++i){
if( vis[ tree[next][i]]==false){
q.push(tree[next][i]);
v.push_back(tree[next][i]);

}
}
}
}
int main( void ){
scanf("%lld",&n);
for(int i=1 ; i<n ; ++i){
scanf("%lld%lld",&a , &b);
tree[a].push_back(b);
```

?

```
tree[b].push_back(a);
}
memset( vis , false , sizeof(vis));
scanf("%lld",&x);
BFS(1 , x);
}
```

0 votes ● Reply ● Message ● Permalink

**Yogen Prajapati** a year ago

```
#include<iostream>
#include<list>
using namespace std;
class bfs{
int nodes;
list<int> *edges;
bool *visited;
public:
int *le;
bfs(){
cin>>nodes;
visited=new bool[nodes];
edges = new list<int>[nodes];
le =new int[nodes];
}
bfs(int nodes)
{
this->nodes=nodes;
visited=new bool[nodes];
for(int i =0;i<nodes;i++)
visited[i]=false;
edges = new list<int>[nodes];
le =new int[nodes];
for(int i =0;i<nodes;i++)
le[i]=0;
}
void addNeighbour(int parent ,int child)
{
edges[parent].push_back(child); //for undirected
edges[child].push_back(parent);
}
void bfsStart(int starting)
{
visited[starting] = true;
list<int> q;
q.push_back(starting);
list<int>::iterator i;
while(!q.empty())
{
starting = q.front();
// cout<<starting<<endl;
q.pop_front();
for(i=edges[starting].begin();i!=edges[starting].end();i++)
{
if(!visited[*i]){
visited[*i]=true;
le[*i]=le[starting]+1;
q.push_back(*i);
}
```

```
}
}
}
};
int main()
{
int n;
cin>>n;
bfs test(n);
for(int i=0;i<n-1;i++)
{
int v,w;
cin>>v;
cin>>w;
test.addNeighbour(v-1,w-1);
}
test.bfsStart(0);
int temp=0;
int x;
cin>>x;
for(int i=0;i<20;i++)
if(test.le[i]==x-1)
temp++;
cout<<endl<<temp;
}
```

output doesnt give correct ans ...why ..?

0 votes ● Reply ● Message ● Permalink

**Amulya Gaur** a year ago

remember root is at level1

0 votes ● Reply ● Message ● Permalink

**Vishwajeet Thoke** a year ago

```
#include <bits/stdc++.h>
using namespace std;
vector<int>vi[100005];
bool vis[100005];
int level[100005];
void init()
{
for(int i=0;i<=100005;i++)vis[i]=false;
}
int bfs(int s,int k)
{
queue<int>qu;
vis[s]=true;
level[s]=1;
qu.push(s);
int c=0;
while(!qu.empty())
{
int v=qu.front();
qu.pop();
for(unsigned int i=0;i<vi[v].size();i++)
{
if(vis[vi[v][i]]==false)
{
qu.push(vi[v][i]);
vis[vi[v][i]]=true;
level[vi[v][i]]=level[v]+1;
```

?

```
if(level[vi[v][i]]==k)c++;
}
}
}
return c;
}
int main()
{
int n;
cin>>n;
init();
n--;
while(n--)
{
int u,v;
cin>>u>>v;
vi[u-1].push_back(v-1);
vi[v-1].push_back(u-1);
}
int k;
cin>>k;
cout<<bfs(0,k)<<endl;
return 0;
}
```

Working!!!!!!!!!!

▲ 0 votes ● Reply ● Message ● Permalink

**Lakshya Aggarwal** a year ago

what's wrong with my code:

```
#include<iostream>
#include<vector>
#include<list>
using namespace std;
vector <int> adj[10];
int level [10];
bool vis[10];
void bfs(int s){
queue <int> q;
q.push(s);
level[s] = 0;
vis[s] = true;
while(!q.empty()){
int p = q.front();
q.pop();

for(int i = 0; i<adj[p].size(); i++){
if( vis[adj[p][i]] == false){
level[adj[p][i]] = level[p]+1;
q.push(adj[p][i]);
vis[adj[p][i]] = true;
}
}
}

}
int main(){
int nodes;
int x,y;
int edges;
cin>>nodes;
```

?

```
cin>>edges;
for(int i=0; i<edges; i++){
cin>>x>>y;
adj[x].push_back(y);
}
bfs(2);
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Ankit Yadav** 10 months ago

Can someone plz tell what's wrong in this code?

```
#include <iostream>
#include <queue>
using namespace std;
int main()
{
int n;
cin>>n;
bool k[n]={false};
int ar[n][n]={0};
int level[n];
int a,b;
for(int i=0;i<n-1;i++)
{
cin>>a>>b;
ar[a-1][b-1]=1;
ar[b-1][a-1]=1;
}
int x;
cin>>x;
x--;
queue <int> q;
q.push(0);
level[0]=1;
k[0]=true;
while(!q.empty())
{
int c=q.front();
q.pop();
if(c==x)
cout<<level[x];
for(int i=0;i<n;i++)
{
if(ar[c][i]!=0&&!k[i])
{ q.push(i);
k[i]=true;
level[i]=level[c]+1;
}
}
}
// cout<<level[x];
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Udit Gaikwad** 9 months ago

how i can improve my code better:

```
#include<bits/stdc++.h>
using namespace std;
```

?

```
int level[50000]= {0};
class graph{
int V;
list<int> *adj;
public:
graph(int V);
void addedge(int v, int w);
void BFS(int s);
};
graph::graph(int V)
{
this->V = V;
adj = new list<int>[V];
}
void graph::addedge(int v , int w)
{
adj[v].push_back(w);
adj[w].push_back(v);
}
void graph::BFS(int s)
{
bool *visited = new bool[V];
for(int i=0; i<V; i++)
{
visited[i] = false;
}
list<int> queue;
visited[s] = true;
level[s] = 0;
queue.push_back(s);
list<int>::iterator i;
while(!queue.empty())
{
s = queue.front();
//cout << s << " ";
queue.pop_front();
for(i= adj[s].begin(); i != adj[s].end(); ++i)
{
if(!visited[*i])
{
visited[*i] = true;
queue.push_back(*i);
level[ *i ] = level[s]+1;
}
}
}
}
int main(){
int n;
cin >>n;
graph g(n+2);
for(int i=0; i<n-1; i++)
{
int n,e; cin >> n>>e;
g.addedge(n,e);
}
g.BFS(1);
int index,t=0;
```

?

```
cin >> index;
//cout << "\n";
for(int i=0; i<50000; i++)
{
if(index-1 == level[i])
{
t++;
}
//cout << level[i] << "\n";
}
cout << t;

return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Xiaohua Yan** 8 months ago

Efficient C++ solution:

```
/*
// Sample code to perform I/O:
cin >> name; // Reading input from STDIN
cout << "Hi, " << name << ".\n"; // Writing output to STDOUT
// Warning: Printing unwanted or ill-formatted data to output will cause the test cases to fail
*/
// Write your code here
#include <iostream>
#include <queue>
#include <vector>
using namespace std;
int main() {
int N;
cin >> N;

vector<vector<int>> neighbors(N);
vector<bool> visited(N);

for (int i = 0; i < N - 1; ++i) {
int n1;
cin >> n1;
int n2;
cin >> n2;
// node index is 1-based
neighbors[n1 - 1].push_back(n2 - 1);
neighbors[n2 - 1].push_back(n1 - 1);
}

vector<int> numNodesPerLevel;
int currLevelNodes = 0;
int nextLevelNodes = 0;

queue<int> Q;
Q.push(0);
visited[0] = true;
++currLevelNodes;
numNodesPerLevel.push_back(1);

while (!Q.empty()) {
auto node = Q.front();
Q.pop();
```

?

```
for (auto neighbor : neighbors[node]) {
if (!visited[neighbor]) {
Q.push(neighbor);
nextLevelNodes++;
visited[neighbor] = true;
}
}
--currLevelNodes;
if (currLevelNodes == 0) {
currLevelNodes = nextLevelNodes;
numNodesPerLevel.push_back(nextLevelNodes);
nextLevelNodes = 0;
}
}

int query;
cin >> query;
cout << numNodesPerLevel[query - 1] << endl;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**niket007** 8 months ago

```
#include <bits/stdc++.h>
using namespace std;
void bfs(int x,int s,int n,vector<list<int> > adj){
int count =0;
queue<int> q;
bool visited[n+1] = {false};
int level[n+1] = {0};
level[0] = -1;
visited[0] = true;
q.push(s);
level[s]=1;
visited[s]=true;
while(!q.empty()){
int temp = q.front();
q.pop();
for(list<int>::iterator it = adj[temp].begin();it != adj[temp].end();it++){
if(!visited[*it]){
q.push(*it);
visited[*it]=true;
level[*it] = level[temp]+1;
}
}
}
for(int i=1;i<n+1;i++){
if(level[i]==x){
count++;
}
}
cout << count << endl;

}
int main(){
int n;
vector<list<int> > adj(n+1);
cin >> n;
for(int i=0;i<n-1;i++){
int a,b;
```

?

```
cin >> a >> b;

adj[a].push_back(b);
adj[b].push_back(a);
}
int x;
cin >> x;

bfs(x,1,n,adj);

}
```

0 votes ● Reply ● Message ● Permalink

**SURAJ KUMAR G SHUKLA** 8 months ago

i am not able to copy my own code from editor . Can somebody help ?

0 votes ● Reply ● Message ● Permalink

**Shubham Jain** 7 months ago

I am getting this error

/hackerearth/CPP14_84/s_d7.cpp: In function ‘void bfs(int, int)’: /hackerearth/CPP14_84/s_d7.cpp:18:14: warning: comparison between signed and unsigned integer expressions [-Wsign-compare] for(int i=0;i

here is my code

```
#include<bits/stdc++.h>
using namespace std;
#define N 100000
int k;
vector <int> a[N];
int level[N];
bool visited[N];
void bfs(int s,int x)
{
level[s]=0;
queue <int> q;
q.push(s);
visited[s]=true;
while(!q.empty())
{
int p=q.front();
q.pop();
for(int i=0;i<a[p].size();i++)
{
if(visited[a[p][i]]==false)
{
level[a[p][i]]=level[p]+1;
if(level[a[p][i]]==x)
{
k++;
}
q.push(a[p][i]);
visited [a[p][i]]=true;
}
}
}
}
int main(){
int i,j,n,m,lvl;
cin>>n;
for(i=0;i<n-1;i++){
int x,y;
```

?

```
cin>>x>>y;
a[x-1].push_back(y-1);
a[y-1].push_back(x-1);
}
cin>>lvl;
bfs(1,lvl);
cout<<co<<endl;
retun 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Sandeep Jangir** Edited 6 months ago

what do I need to change here as I'm getting wrong answer??

```
#include<bits/stdc++.h>
using namespace std;
vector<int> tree[100000];
bool visited[100000];
void addEdge(int a, int b)
{
tree[a-1].push_back(b-1);
tree[b-1].push_back(a-1);
}
void bfs(int x)
{
queue<int> q;
int level[100000];
int count = 0;
//memset(visited, false, visited.size());
q.push(1);
visited[1] = true;
level[1] = 0;
while(!q.empty())
{
int p = q.front();
q.pop();
for(int i = 0; i < tree[p].size(); i++)
{
if(visited[tree[p][i]] == false)
{
q.push(tree[p][i]);
visited[tree[p][i]] = true;
level[tree[p][i]] = level[p] + 1;
if(level[tree[p][i]] == x)
count++;
}
}
}
cout<<count;
}
int main(void)
{
int nodes, edges, a, b;
cin>>nodes;
edges = nodes-1;
while(edges--)
{
cin>>a>>b;
addEdge(a, b);
}
```

?

```
int x;
cin>>x;
bfs(x);
}
```

0 votes ● Reply ● Message ● Permalink

**Mohit Yadav** 6 months ago

How is this?

```
//------------------------------------------------------------------------------------------------------------
#include<iostream>
#include<queue>
#include<vector>
using namespace std;
int BFS(const vector<vector<int> >& myVector,int source, int alevel){
int cnt =0;
vector<bool> visited(myVector.size());
for(int i=0;i<myVector.size();i++){
visited[i] = false;
}
queue <int> q;
vector<int>level;
level.resize(myVector.size());
q.push(source);
visited[source]=true;
level[source] = 0;
while(!q.empty()){
int v = q.front();
q.pop();
for(int i=0;i<myVector[v].size();i++){
if(visited[myVector[v][i]]==false){
q.push(myVector[v][i]);
visited[myVector[v][i]]=true;
level[myVector[v][i]] = 1+level[v];
}
}
}
for(int i=0;i<level.size();i++){
if(level[i]==alevel)cnt++;
}
return cnt;
}
int main(void){
int t=1;
//cin>>t;
while(t--){
int n,x,y;
cin>>n;
vector<vector<int> > myVector;
myVector.resize(n);
for(int i=0;i< n-1;i++){
cin>>x>>y;
myVector[x-1].push_back(y-1);
myVector[y-1].push_back(x-1);
}
int alevel;
cin>>alevel;
cout<< BFS(myVector,0,alevel-1)<<endl;
}
return 0;
}
```

?

▲ 0 votes ● Reply ● Message ● Permalink

**pawan phalak** Edited 6 months ago

your output and correct output is matching but still, it shows Runtime Error - SIGABRT

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
long n,i,j,k,a,b;
cin>>n;
vector< vector<int> > v(n+1);
j=n-1;
while(j--)
{
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
bool visited[21];
for(i=0;i<n+1;i++)
visited[i]=0;
queue<int> q;
q.push(1);
long level[n+1];
for(i=0;i<n+1;i++)
level[i]=1;
while(!q.empty())
{
j=q.front();
visited[j]=1;
q.pop();
for(i=0;i<v[j].size();i++)
if(visited[v[j][i]]==0)
{
q.push(v[j][i]);
visited[v[j][i]]==1;
level[v[j][i]]=level[j]+1;
}
}
long x,count=0;
cin>>x;
for(i=1;i<=n;i++)
if(level[i]==x)
count++;
cout<<count<<endl;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Jasmeet Singh** 6 months ago

Hey, wouldn't the last example of 0-1 graph will run in infinite loop.?? Because every time a node is popped from the queue...all its neighbours (even if they are visited before) are added either at front or at the back of queue.

▲ 0 votes ● Reply ● Message ● Permalink

**Ali Teoman Demir** 5 months ago

```
#include <iostream>
#include <vector>
#include <queue>
using namespace std;
```

?

```
void bfs(vector<int>*G,int size,int s,int* level){
bool visited[size]={0};
queue <int> q;
q.push(s);
level[s]=1;
visited[s]=true;
while(!q.empty()){
int p;
p=q.front();
q.pop();
for(int i=0;i<G[p].size();i++){
if(visited[G[p][i]]==false){
level[G[p][i]]=level[p]+1;
q.push(G[p][i]);
visited[G[p][i]]=true;
}
}
}
}
int main(){
int N;
cin>>N;
vector <int> v[N+1];
int level[N+1]={0};
for(int i=0;i<N-1;i++){
int a;
int b;
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
bfs(v,N+1,1,level);
int q;
cin>>q;
int count=0;
for(int i=0;i<=N;i++){
if(level[i]==q){
count++;
}
}
cout<<count;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**sjtuzihan** 5 months ago

This is my code, i run it in CLion and it gives the right answer, but when online there is a compile error:
In function `bfs()':
(.text+0x54): relocation truncated to fit: R_X86_64_PC32 against symbol `level' defined in .bss section in /tmp/ccaPqkRc.o
(.text+0xa4): relocation truncated to fit: R_X86_64_PC32 against symbol `connect' defined in .bss section in /tmp/ccaPqkRc.o
(.text+0x11e): relocation truncated to fit: R_X86_64_PC32 against symbol `level' defined in .bss section in /tmp/ccaPqkRc.o
(.text+0x17a): relocation truncated to fit: R_X86_64_PC32 against symbol `connect' defined in .bss section in /tmp/ccaPqkRc.o
In function `main':
(.text.startup+0x3b): relocation truncated to fit: R_X86_64_PC32 against symbol `level' defined in .bss section in /tmp/ccaPqkRc.o
(.text.startup+0x60): relocation truncated to fit: R_X86_64_PC32 against symbol `connect' defined in

?

```
.bss section in /tmp/ccaPqkRc.o
(.text.startup+0x8d): relocation truncated to fit: R_X86_64_PC32 against symbol `connect' defined in
.bss section in /tmp/ccaPqkRc.o
(.text.startup+0xf9): relocation truncated to fit: R_X86_64_PC32 against symbol `connect' defined in .bss
section in /tmp/ccaPqkRc.o
In function `_GLOBAL__sub_I_level':
(.text.startup+0x192): relocation truncated to fit: R_X86_64_32 against `.bss'
(.text.startup+0x1b5): relocation truncated to fit: R_X86_64_32 against `.bss'
error: ld returned 1 exit status
```

I don't know why..

```
#include <iostream>
#include <queue>
using namespace std;
int *level, *connect, tree[100001][100001], x, N, a, b, cnt = 0;
bool *visited;
queue<int> q;
void bfs()
{
q.push(1);
level[1] = 1;
visited[1] = true;
while(!q.empty())
{
int t = q.front();
q.pop();
for(int i = 1; i <= connect[t]; ++i)
if(visited[tree[t][i]] == false)
{
q.push(tree[t][i]);
level[tree[t][i]] = level[t] + 1;
if(level[tree[t][i]] == x)
cnt++;
visited[tree[t][i]] = true;
}
}
}
int main()
{
cin >> N;
level = new int[N+1];
connect = new int[N+1];
visited = new bool[N+1];
for(int i = 1; i <= N; ++i)
{
connect[i] = 0;
visited[i] = false;
}
for(int i = 1; i < N; ++i)
{
cin >> a >> b;
++connect[a];++connect[b];
tree[a][connect[a]] = b;
tree[b][connect[b]] = a;
}
cin >> x;
bfs();
cout << cnt << endl;
return 0;
}
```

?

▲ 0 votes ● Reply ● Message ● Permalink

**Shubham Sawlani** 5 months ago

My solution

```
//Graph _programmming
#include<iostream>
#include <vector>
#include <queue>
#include <cstring>
using namespace std;
int main()
{
vector<int> v[100000];
queue <int> q;
int nodes;
cin >> nodes;
int j;

bool vis[nodes+1];
memset(vis,0,sizeof(vis));
int level[100000];
int i;
for(i=0;i<nodes-1;i++)
{
int x,y;
cin >> x;
cin >> y;
v[x].push_back(y);
v[y].push_back(x);
}
cin >> j;
int r=1;//rot node
vis[r]=true;
level[r]=1;
q.push(r);
int c=0;
while(!q.empty())
{
int p=q.front();
q.pop();
for(i=0;i<v[p].size();i++)
{
int m=v[p][i];
if(vis[m]!=true)
{
vis[m]=true;
q.push(m);
level[m]=level[p]+1;
if(level[m]==j)
c++;
}
}
}
cout << c;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Nasar uddin** 5 months ago

Probably the easiest solution :

```
public class Breth {
```

?

```
int[][] graph;
LinkedList<Integer> result = new LinkedList<Integer>();
boolean visited[];
public Breth(int[][] graph){
this.graph= graph;
visited = new boolean[graph.length];
for(int i=0;i<visited.length;i++){
visited[i]=false;
}
}
public void Bsf(int start){
LinkedList<Integer> queue = new LinkedList<Integer>();
queue.add(start);
visited[start]=true;
while(queue.size()!=0){
start = queue.poll();
result.add(start);
for(int i=0;i<graph.length;i++){
if(!visited[i]&&graph[start][i]==1){
queue.add(i);
visited[i]=true;
}
}
}
for(int a:result){
System.out.print(a+">");
}
}
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Hayat Hayati** 4 months ago

```
#include<stdio.h>
#include<conio.h>
typedef int bool;
#define true 1
#define false 0

int que[20];
int front=0, rear=0, n;
int arr[20];
int bfs(int );
int ajMat[20][20];
int b[20];
void display();
int p=0;

int main()
{
char v;
int vi;
printf("Enter the number of nodes in a graph : ");
scanf("%d",&n);
printf("Enter the value of node of graph : ");
for(int i=0; i<n; i++)
{
scanf("%d",&b[i]);
}

printf("Enter the value in adjancency matrix in from of 'y' or 'n'\n");
```

?

```
printf("If there exits an edge between two vertices than 'y' otherwise 'n'\n");

printf(" ");

for(int i=0; i<n; i++)
printf(" %d",b[i]);

for(int i=0;i<n; i++)
{
printf("\n%d ",b[i]);
for(int j=0; j<n; j++)
{
v=getch();
if(v==48)
vi=0;
else
vi=1;
printf("%d ",vi);
ajMat[i][j]=vi;
}
printf("\n");
}
for(int i=0;i<n;i++)
bfs(i);

display();
getch();
}

void display()
{
printf("\n\nBFS of Graph : ");
for(int i=0; i<n; i++)
printf("%d ",arr[i]);
}

void insert(int val)
{
que[front]=val;
front++;
}

char del()
{
rear=rear+1;
return que[rear-1];
}

int unVisit(int val)
{
for(int i=0; i<front; i++)
{
if(val==que[i])
return false;
}
return true;
}

int bfs(int i)
{
int m;
```

```
if(front==0)
{
insert(b[i]);
}
for(int j=0; j<n; j++)
{
if(ajMat[i][j] == 1)
{
if(unVisit(b[j]))
{
insert(b[j]);
}
}
}
m=del();
arr[p]=m;
p++;
return 0;
}
```

WHATS WRONG WITH THIS CODE?

▲ 0 votes ● Reply ● Message ● Permalink

**Ishan Sri** 4 months ago

My python code:-

```
a=[None]*100001
k2=[]
n=int(input())
level=[0]*100001
vis=[False]*100001
for i in range(n-1):
k,m=map(int,input().split())
if(a[k] is None):
a[k]=[m]
else:
a[k].append(m)
if(a[m] is None):
a[m]=[k]
else:
a[m].append(k)
#print(a)
def bfs(s,x):
k1=0
level[s]=1
k2.append(s)
vis[s]=True
while(len(k2)!=0):
p=k2[0]
k2.pop(0)
#print(a[p])
#if(a[p]!=None):
for i in range(len(a[p])):
#print(a[p][i])
if(vis[a[p][i]]==False):
vis[a[p][i]]=True
level[a[p][i]]=level[p]+1
if(level[a[p][i]]==x):
k1+=1
k2.append(a[p][i])

return k1
```

?

```
x=int(input())
print(bfs(1,x))
```

▲ 0 votes ● Reply ● Message ● Permalink

**Saumya Verma** 4 months ago

```
#include<iostream>
#include<vector>
#include<queue>
using namespace std;
vector<int> v[100];
int level[100];
bool visited[100];
void bfs(int s,int x)
{ int count=0;
queue <int> q;
q.push(s);
level[s]=1;
visited[s]=true;
while(!q.empty())
{
int p = q.front();
q.pop();
for(int i=0;i<v[p].size();i++)
{
if(visited[v[p][i]]==false)
{
level[v[p][i]] = level[p]+1;
if(level[v[p][i]]==x)
{
count++;
}
q.push(v[p][i]);
visited[v[p][i]] = true;
}
}
}
cout<<count<<endl;
}
int main()
{
int n,x,s=1;
cin>>n;
int a,b;
for(int i=0;i<n;i++)
{
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
cin>>x;
bfs(s,x);
return 0;
}
```

My output evaluates to zero. Why?

▲ 0 votes ● Reply ● Message ● Permalink

**dharmendra maurya** 4 months ago

please explain error in the code
def adjacent(start):

?

```
adj=[]
for x in nodes:
if start in x :
if x[0]==start:
a=x[1]
else :
a=x[0]
if visited[a]==False:
adj.append(a)

return adj

n=int(input())
nodes=[]
for i in range(n-1):
a,b=input().split()
a=int(a)-1
b=int(b)-1
nodes.append([a,b])
lev=int(input())
parent=[0]*n
visited=[False]*n
level=[0]*n
level[0]=1
que=[0]
while len(que)!=0:
a=que.pop()
ad=adjacent(a)
if len(ad):
for x in ad:

level[x]=level[a]+1
visited[x]=True
que.append(x)
print(level.count(lev))
```

▲ 0 votes ● Reply ● Message ● Permalink

**Chandrala Jeshwanth Kumar** 3 months ago

why to use a deque in 0-1 bfs cant we directly push in a normal queue if distance get modified

▲ 0 votes ● Reply ● Message ● Permalink

**Spoorthi Vaidya** 3 months ago

```
#include<iostream>
#include<vector>
#include<queue>
using namespace std;
#define N 100000
int k;
vector <int> v[N];
int level[N];// todetermine the levelof each node
bool visited[N];
int BFS(int s,int x)
{
level[s]=1;
queue <int> q;
q.push(s);
visited[s]=true;
k=0;
```

?

```
while(!q.empty())
{
int p=q.front();
q.pop();
for(int i=0;i<v[p].size();i++)
{
if(visited[v[p][i]]==false)
{
level[v[p][i]]=level[p]+1;
if(level[v[p][i]]==x)
{
k++;
}
q.push(v[p][i]);
visited [v[p][i]]=true;
}
}
}
return k;
}
int main()
{
int x,y,nodes,edges;
cin>>nodes;
int k=0;
for(int i=0;i<nodes-1;i++)
{
cin>>x;
cin>>y;
v[x].push_back(y);
v[y].push_back(x);
}
int b;
cin>>b;
k=BFS( 1,b);
cout<<k;
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Mukesh Kumar** 3 months ago

working code :

```
#include<bits/stdc++.h>
using namespace std;
int level[100005]={0};
bool vis[100005]={false};
vector<int>v[100005];
void bfs(int s)
{
queue<int>q;
q.push(s);
level[s]=0;
vis[s]=true;
while(!q.empty()){
int p=q.front();
q.pop();
for(int i=0;i<v[p].size();i++)
{
if(vis[v[p][i]]==false){
```

```
level[v[p][i]]=level[p]+1;
q.push(v[p][i]);
vis[v[p][i]]=true;
}
}
}
}
int main()
{
int n;
cin>>n;
int x,y;
for(int i=1;i<n;i++)
{
cin>>x>>y;
v[x].push_back(y);
v[y].push_back(x);
}
int qr;
cin>>qr;
qr--;
bfs(1);
int cnt=0;
for(int i=1;i<=100005;i++)
{
if(level[i]==qr)
cnt++;
}

cout<<cnt<<"\n";
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**Bharat Kumar** 2 months ago

// plz anyone help me what problem in my code

```
#include<iostream>
#include<string.h>
#include<vector>
#include<queue>
using namespace std;
int main(){
int n;
cin>>n;
vector<int>array[n+1];
int b[n];
for(int i=1;i<n;i++){
int j,k;
cin>>j>>k;
b[i]=j;
array[j].push_back(k);
}

int s,f=1,v,level=1;
cin>>s;
queue<int>q;
queue<int>q1;
q.push(b[1]);
v=q.front();
q.pop();
vector<int>degree[n];
```

?

```
degree[level].push_back(f);
f=0;
for(int i=0;i<array[v].size();i++){
q1.push(array[v][i]);
f++;
}
level++;
degree[level].push_back(f);
while(!q.empty() || !q1.empty()){
degree[level].push_back(f);
f=0;
level++;
while(!q1.empty()){
int t=q1.front();
q1.pop();
for(int i=0;i<array[t].size();i++){
q.push(array[t][i]);
f++;
}
}
while(!q.empty()){
q1.push(q.front());
q.pop();
}
}

cout<<degree[s][0]<<"\n";
return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

**SAGAR KUMAR** 2 months ago

IMPLEMENTATION USING MAP:

```
*-----------------------------------------*
#include <bits/stdc++.h>
using namespace std;
#define ll long long int
vector<ll> v[100001];
ll level[100001];
bool vis[100001];
unordered_map<ll,ll> m;
void bfs(ll s){
m.clear();
queue <int> q;
q.push(s);
level[ s] = 1 ;
vis[ s ] = true;
while(!q.empty()){
int p = q.front();
q.pop();
for(int i = 0;i < v[ p ].size() ; i++){
if(vis[ v[ p ][ i ] ] == false){
level[ v[ p ][ i ] ] = level[ p ]+1;
m[level[ p ]+1]++;
q.push(v[ p ][ i ]);
vis[ v[ p ][ i ] ] = true;
}
}
}
```

```
}
int main(){
ll n;
cin>>n;
ll a,b;
while(--n){
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
bfs(1);
ll x;
cin>>x;
cout<<m[x]<<endl;
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**raghav singh** 2 months ago

in 0-1 bfs given in the tutorial how will the program terminate since the node is not marked visited how will the program terminate

0 votes ● Reply ● Message ● Permalink

**anshika.jain14aj** 2 months ago

can anyone tell me my mistake;

```
#include <iostream>
#include <vector>
#include <queue>
using namespace std;
vector<int> adj[20];
int level[20];
bool vis[20];
int bfs(int s,int ele)
{
int l=1;
queue<int> q;
q.push(s);
level[s]=1;
vis[s]=true;
while(!q.empty())
{
int p=q.front();
q.pop();
for(int i=0;adj[p].size();i++)
{
if(vis[adj[p][i]]==false)
{
level[adj[p][i]]=level[p]+1;
q.push(adj[p][i]);
vis[adj[p][i]]=true;
if(level[adj[p][i]]==ele)
l++;

}
}

}
/* for(int i=0;i<20;i++)
{
if(level[i]==l)
```

```
c++;
}*/
return l;
}
int main()
{
int x,y,n,e;
cin>>n;
e=n-1;
for(int i=0;i<e;i++)
{
cin>>x>>y;
adj[x].push_back(y);
}
int count=0,ele;
cin>>ele;
count=bfs(1,ele);
cout<<count;
return 0;
}
```

0 votes ● Reply ● Message ● Permalink

**Saransh Sharma** a month ago

I was getting error in my code from past week ,after a week the problem was that i didn't intialised my bool array with false.Sometimes little mistakes leads to great blunders.

0 votes ● Reply ● Message ● Permalink

**Abhishek Javali** a month ago

The correct solution of input #1 is 6 (12,20,41,6,14,17) and for input #2 is 16 (19, 6, 7, 46, 37, 64, 95, 17, 23, 28, 81, 74, 84, 30, 60, 41). Please correct it.

0 votes ● Reply ● Message ● Permalink

**DIVYANSHI MANGAL** a month ago

It's working fine..

```
#include<bits/stdc++.h>
using namespace std;
#define n 100000
vector <int> adj[n];
int level[n];
bool vis[n];
int bfs(int s,int k){
level[s]=1;
queue <int> q;
q.push(s);
vis[s]=true;
int coun=0;
while (!q.empty()){
int p=q.front();
q.pop();
vis[p]=true;
for (int i=0;i<adj[p].size();i++){
if (vis[adj[p][i]]==false){
level[adj[p][i]]=level[p]+1;
q.push(adj[p][i]);
vis[adj[p][i]]=true;
if (level[adj[p][i]]==k){
coun++;
}
}}
```

?

```
}
return coun;
}
int main(){
int nodes,edges;
cin>>nodes;
edges=nodes-1;
//vector <int> adj[]
int x,y;
for (int i=0;i<edges;i++){
cin>>x;
cin>>y;
adj[x].push_back(y);
adj[y].push_back(x);
}
int k;
cin>>k;
cout<<bfs(1,k)<<endl;

return 0;
}
```

▲ 0 votes ● Reply ● Message ● Permalink

About Us | Innovation Management | Technical Recruitment
University Program | Developers Wiki | Blog
Press | Careers | Reach Us

Site Language: English ▼ | Terms and Conditions | Privacy |© 2018 HackerEarth