Everyone's Connected

Contents

1 Template

```cpp
#include <bits/stdc++.h>
#define endl '\n'
#define ll long long int
#define ull unsigned long long int

using namespace std;

/*------------------SOLBEGIN------------------*/

void solve() {
    return;
}

int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);

    int t = 1; cin >> t;
    while (t--) solve();

    return 0;
}
```

2 Data structures

2.1 Simplified DSU (Stolen from GGDem)

2.2 Disjoint Set Union

2.3 Segment tree

2.4 Segment tree Lazy

2.5 Trie

3 Graphs

3.1 Graph Transversal

3.1.1 BFS

```cpp
#include <bits/stdc++.h>
```

```cpp

using namespace std;

vector<bool> visited;
vector<vector<int>> adj;

void breadth_first_search(int node) {
    // BFS requieres queue data structure, starting from initial node 0
    queue<int> q;
    q.push(node);
    visited[node] = true;
    // While queue is not empty, pop the first element and push its
        children
    while (!q.empty()) {
        int v = q.front();
        cout << v << "␣";
        q.pop();
        // Push all children of v
        for (int u : adj[v]) {
            // If not visited, push and mark as visited
            if (!visited[u]) {
                q.push(u);
                visited[u] = true;
            }
        }
    }
}

int main() {
    int nodes, edges;
    cin >> nodes >> edges;
    // Initialize visited and adjacency list
    visited.assign(nodes, false);
    adj.assign(nodes, vector<int>());
    int u, v;
    // Values of nodes staring from 0, given as pairs of nodes
    for (int i = 0; i < edges; i++) {
        cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u); // Assuming undirected graph
    }
    breadth_first_search(0); // Start BFS from node 0

    return 0;
}
```

3.1.2 DFS

3.2 Topological Sort

3.3 APSP: Floyd Warshall

3.4 SSSP

3.4.1 Lazy Dijkstra

3.4.2 Bellman-Ford

3.5 Strongly Connected Components: Kosaraju

3.6 Articulation Points and Bridges: ModTarjan

4 Math

4.1 Identities

Coeficientes binomiales.

$(a+b)^n = \Sigma_{k=0}^{n} \binom{n}{k} a^{n-k} b^k$

$\binom{n}{k} = \binom{n}{n-k}$

$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$

$k\binom{n}{k} = n\binom{n-1}{k-1}$

$\Sigma_{k=0} n\binom{n}{k} = 2^n$

$\Sigma_{k=0}^{n} (-1)^k \binom{n}{k} = 0$

$\binom{n+m}{t} = \Sigma_{k=0}^{t} \binom{n}{k}\binom{m}{t-k}$

$\Sigma_{j=k}^{n} \binom{j}{k} = \binom{n+1}{k+1}$

Numeros Catalanes.

$C_n = \frac{2(2n-1)}{n+1} C_{n-1}$

$C_n = \frac{1}{n+1}\binom{2n}{n}$

$C_n \sim \frac{4^n}{n^{3/2}\sqrt{\pi}}$

$\Sigma(n) = O(\log(\log(n)))$ (number of divisors of n)

$F_{2n+1} = F_n^2 + F_{n+1}^2$

$F_{2n} = F_{n+1}^2 - F_{n-1}^2$

$\sum_{i=1}^{n} F_i = F_{n+2} - 1$

$F_{n+i}F_{n+j} - F_n F_{n+i+j} = (-1)^n F_i F_j$

(Möbius Function)

0 if n is square-free

1 if n got even amount of distinct prime factors

0 if n got odd amount of distinct prime factors

(Möbius Inv. Formula)

Let $g(n) = \sum_{d|n} f(d)$, then $f(n) = \sum d \mid n g(d)\mu\left(\frac{n}{d}\right))$.

Permutaciones objetos repetidos

$$P(n,k) = \frac{P(n,k)}{n_1!n_2!...}$$

Separadores, Ecuaciones lineares a variables $= b$

$$\binom{\binom{a}{b}}{} = \binom{a+b-1}{b} = \binom{a+b-1}{a-1}$$

Teorema chino

sean $\{n_1, n_2, ..., n_k\}$ primos relativos

$$P = n_1 \cdot n_2 \cdot ... \cdot n_k$$

$$P_i = \frac{P}{n_i}$$

$$x \cong a_1(n_1)$$

$$x \cong a_2(n_2) \ ... \ x \cong a_k(n_k)$$

$P_1S_1 \cong 1(n_1)$ Donde S soluciones.

$$x = P_1S_1a_1 + P_2S_2a_2...P_kS_ka_k$$