USA  Tutor

Fence Planning

Analysis by David Yang

Statement Summary

We have N<=1e5 cows

There are M<=1e5 edges/connections between the cows

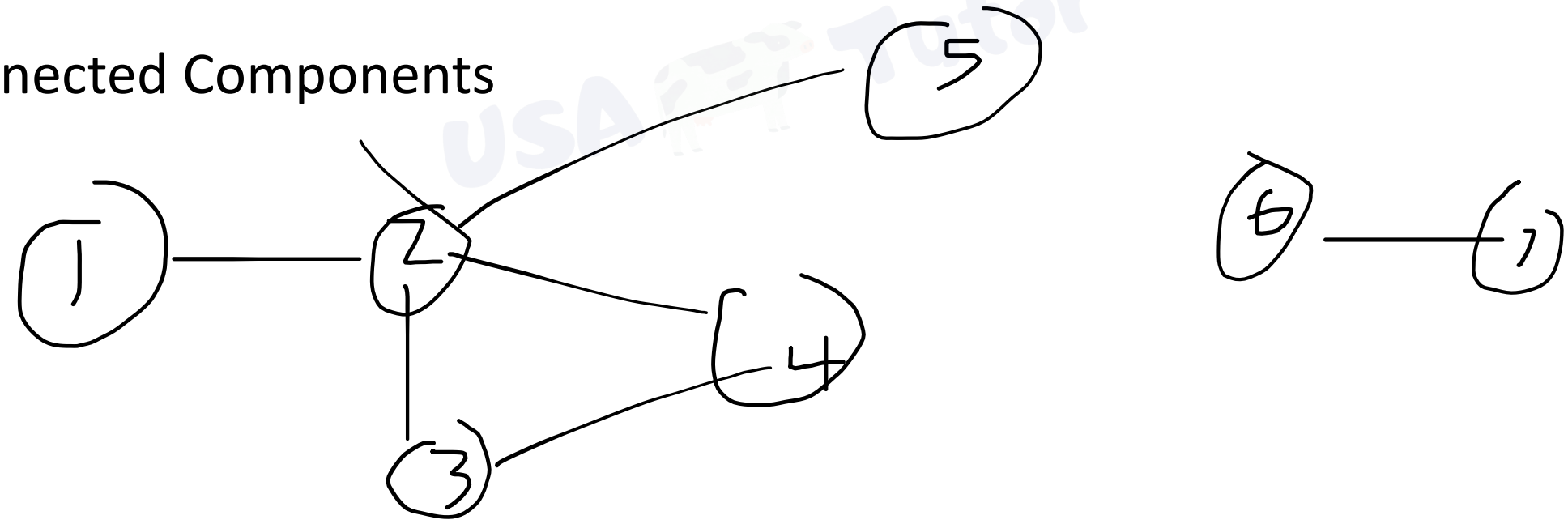
Each cow has a distinct (x,y) coordinate

Find the minimum perimeter of a “moonet”

What's a Moonet?

A moonet is a set of cows that are reachable from any node to all other nodes through some direct/indirect path.

Connected Components



Two Parts to the Solution

1. What are my connected components?
2. What is the perimeter of each connected component?

The answer is just the minimum of (2)

What are my connected components?

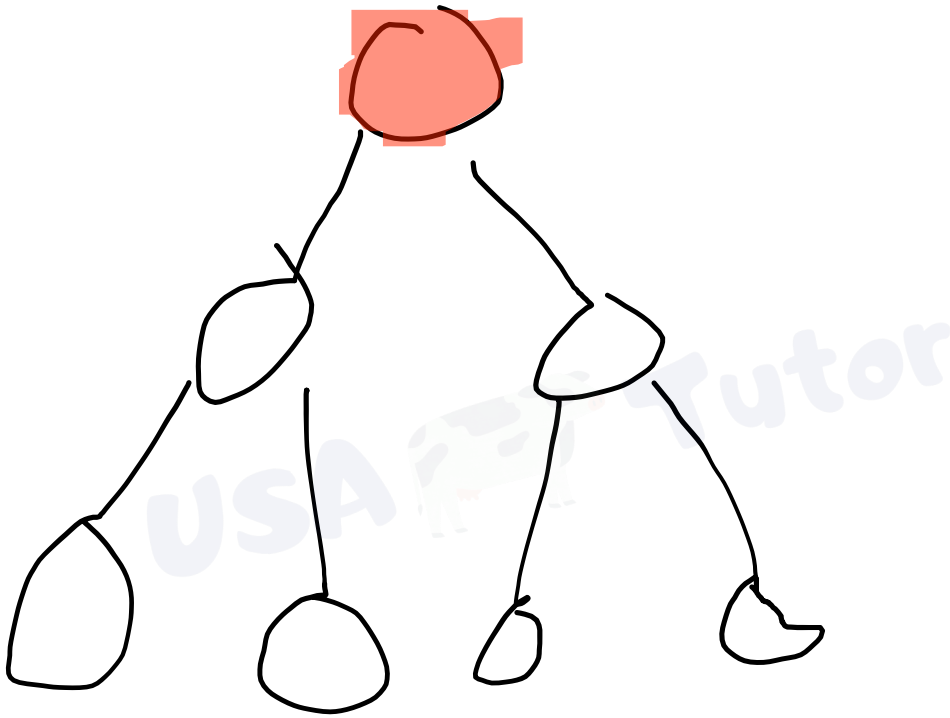
We need DFS

What is DFS?

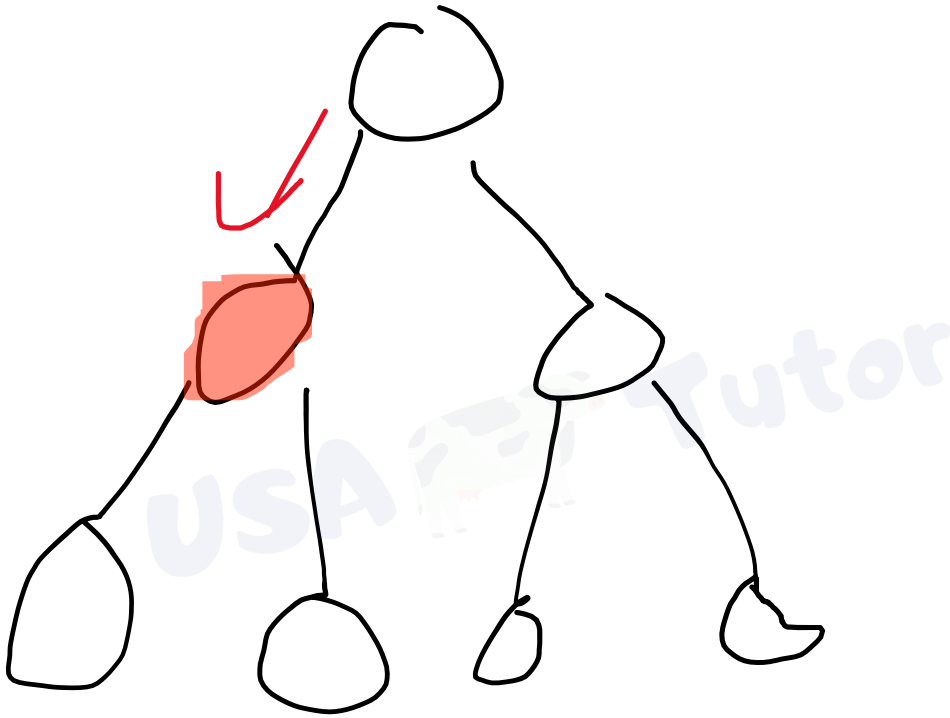
DFS is a graph/tree traversal algorithm that allows you to move along edges of a graph/tree.

We use recursion.

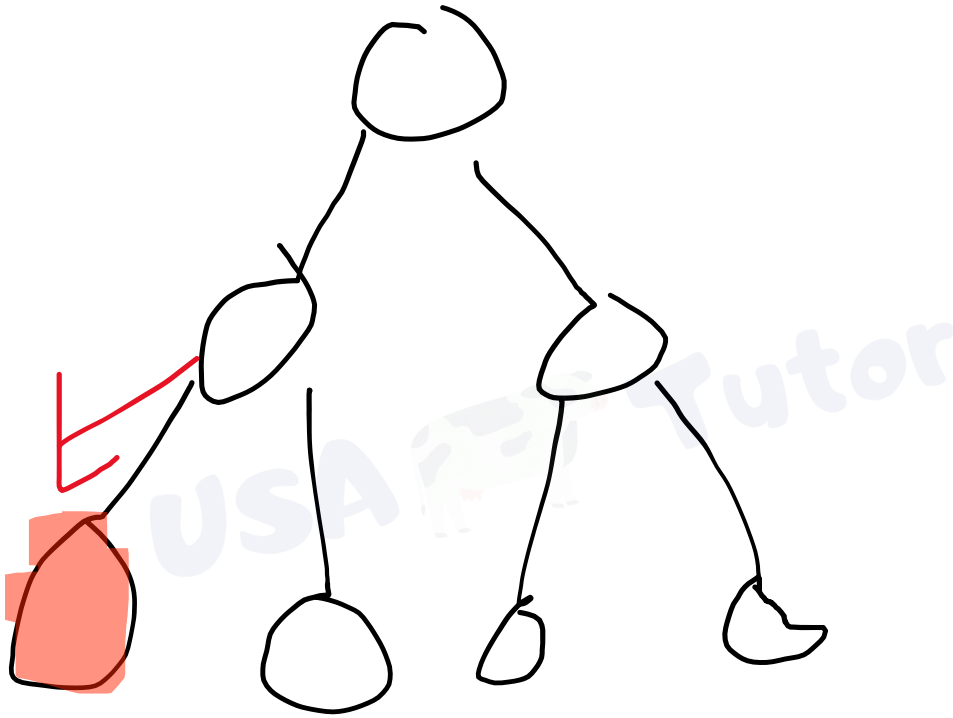
What is DFS?



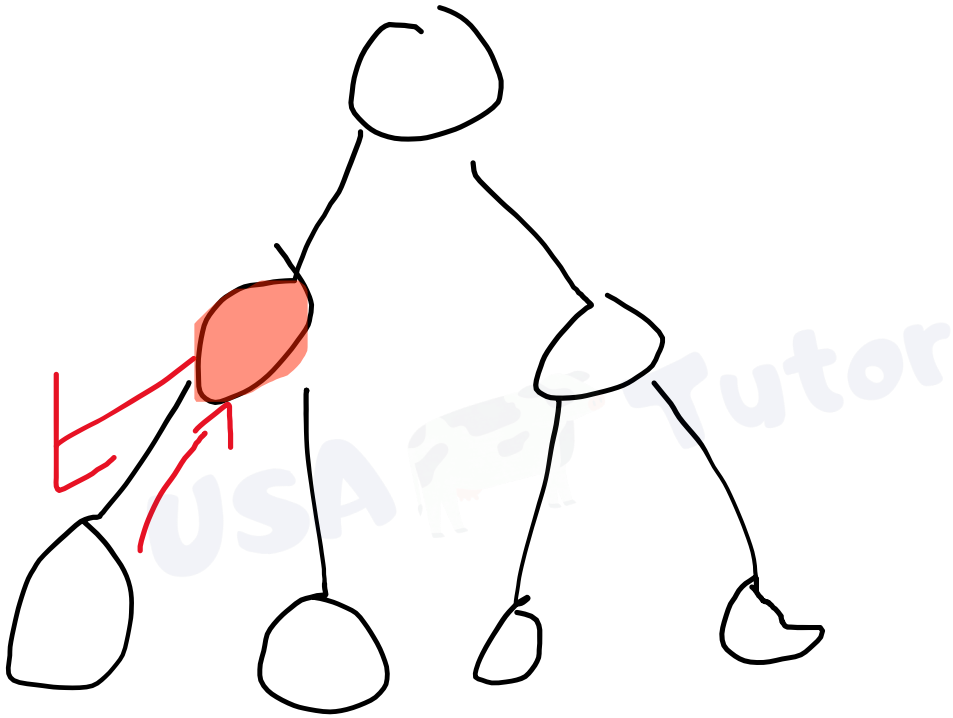
What is DFS?



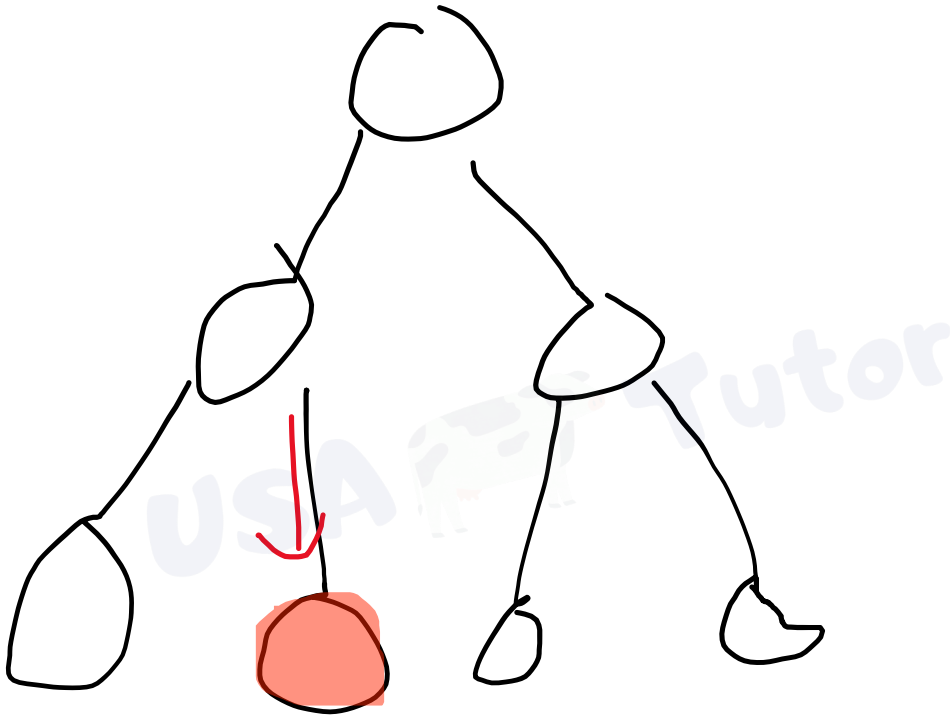
What is DFS?



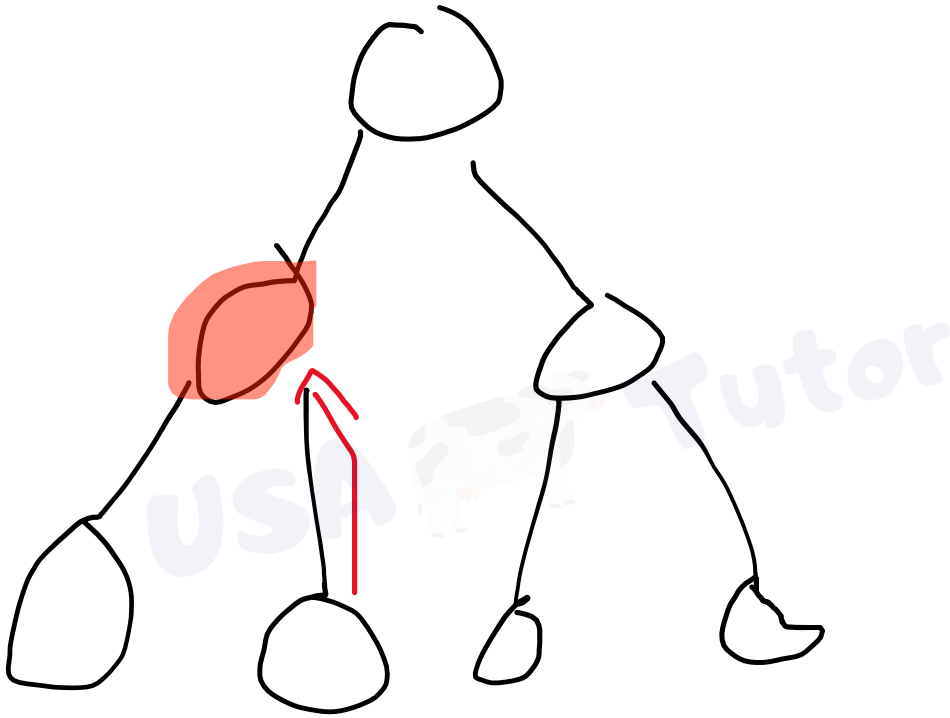
What is DFS?



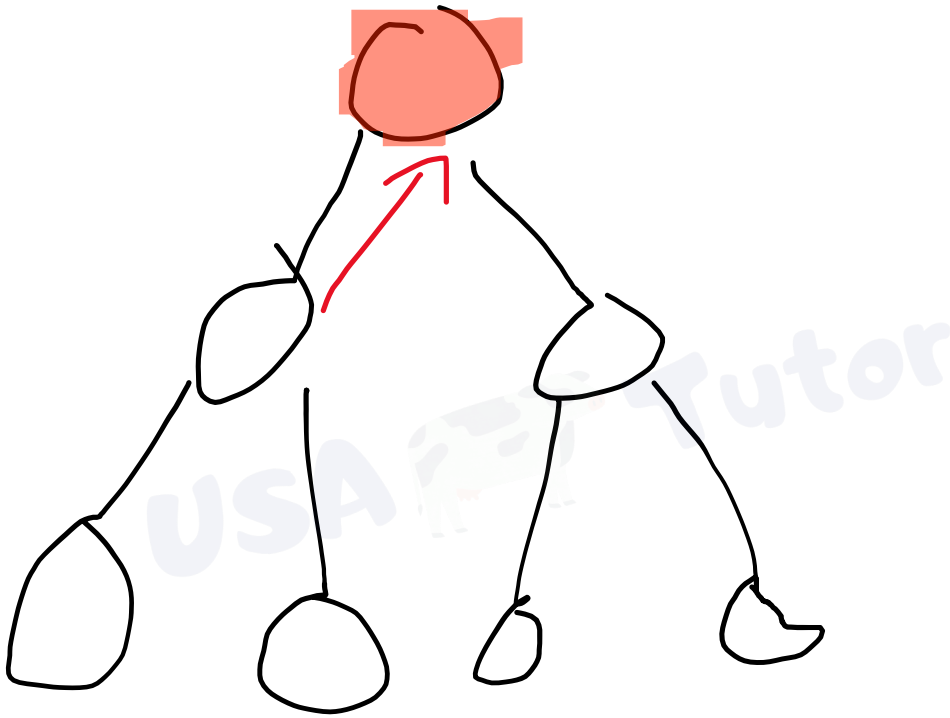
What is DFS?



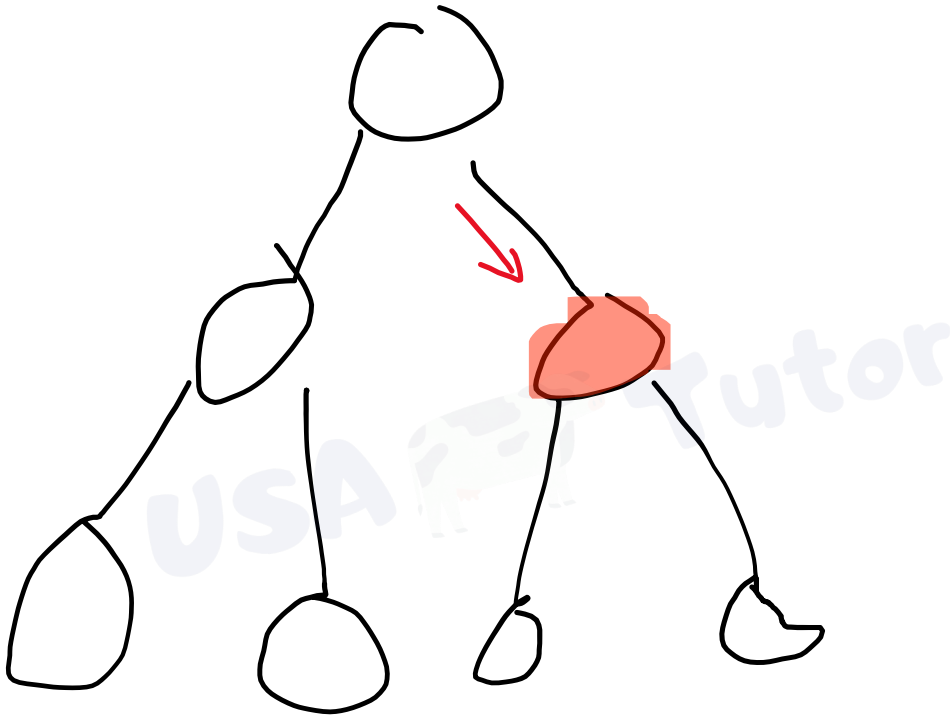
What is DFS?



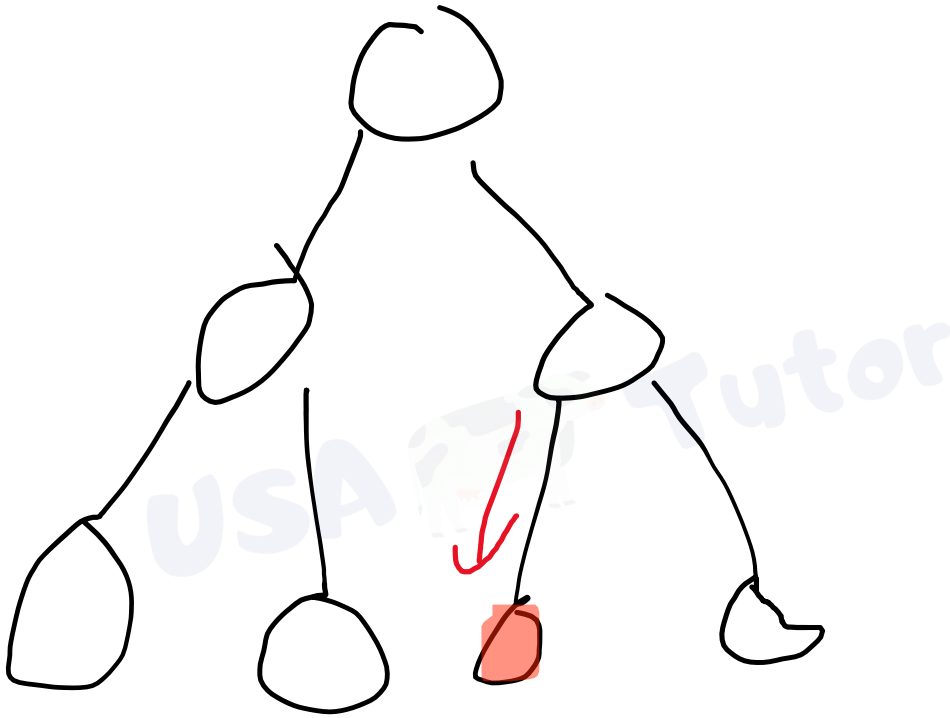
What is DFS?



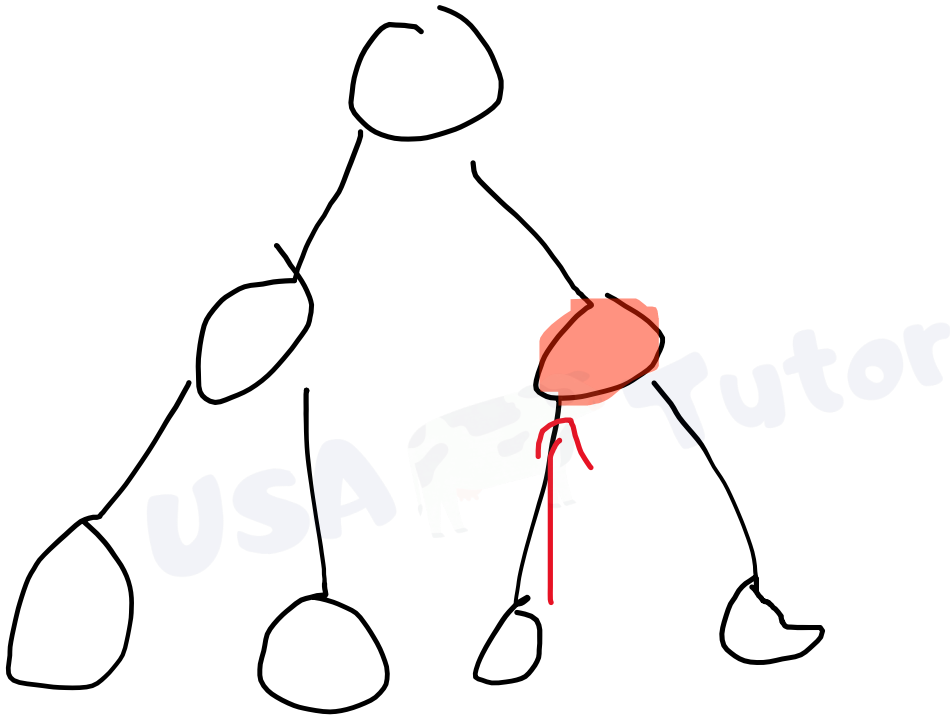
What is DFS?



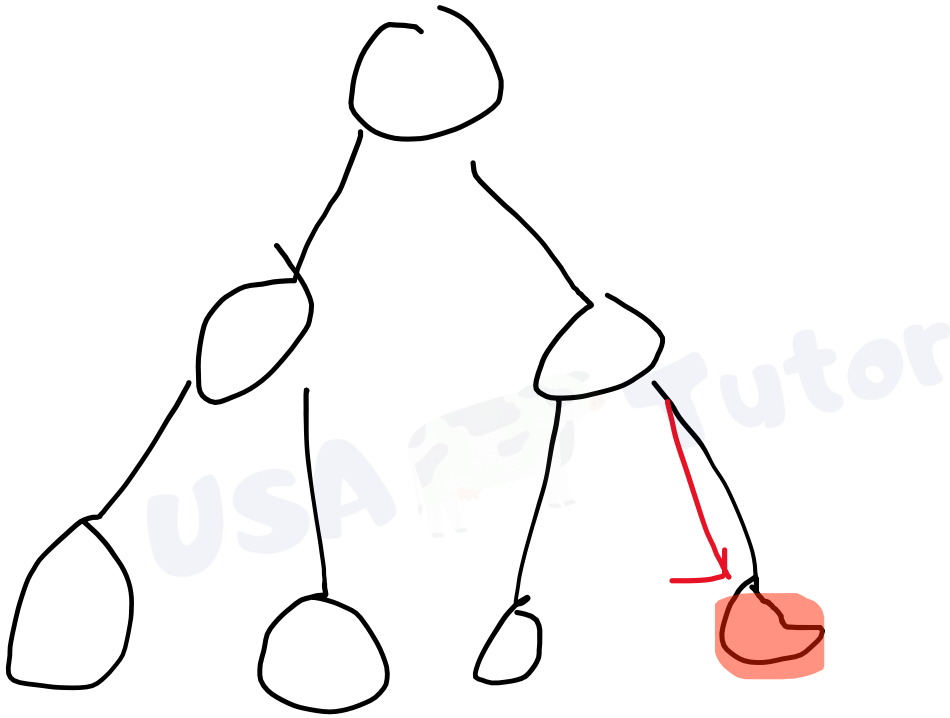
What is DFS?



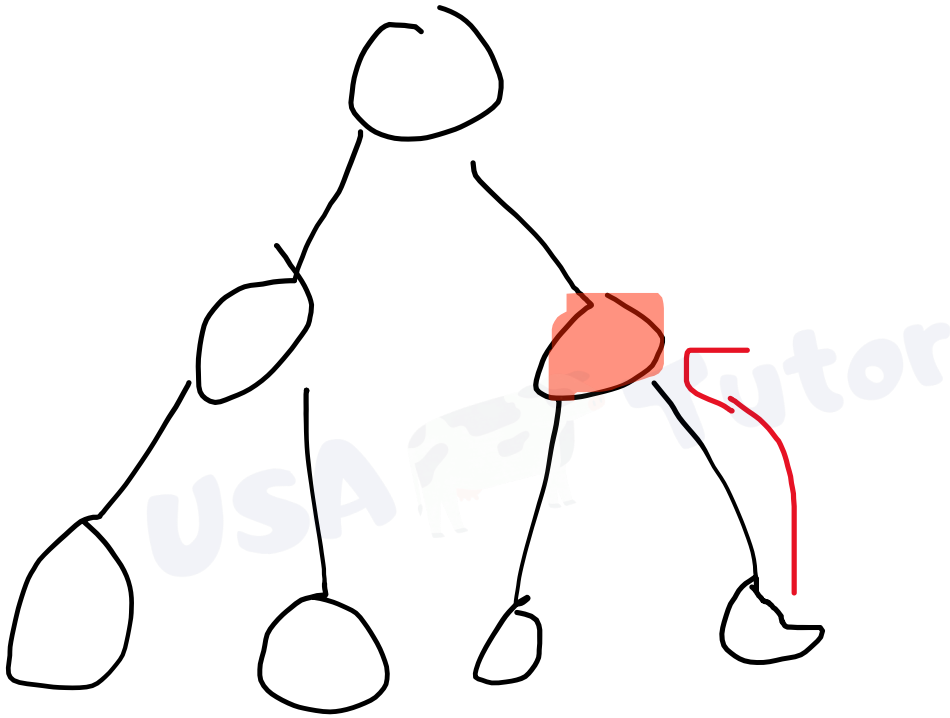
What is DFS?



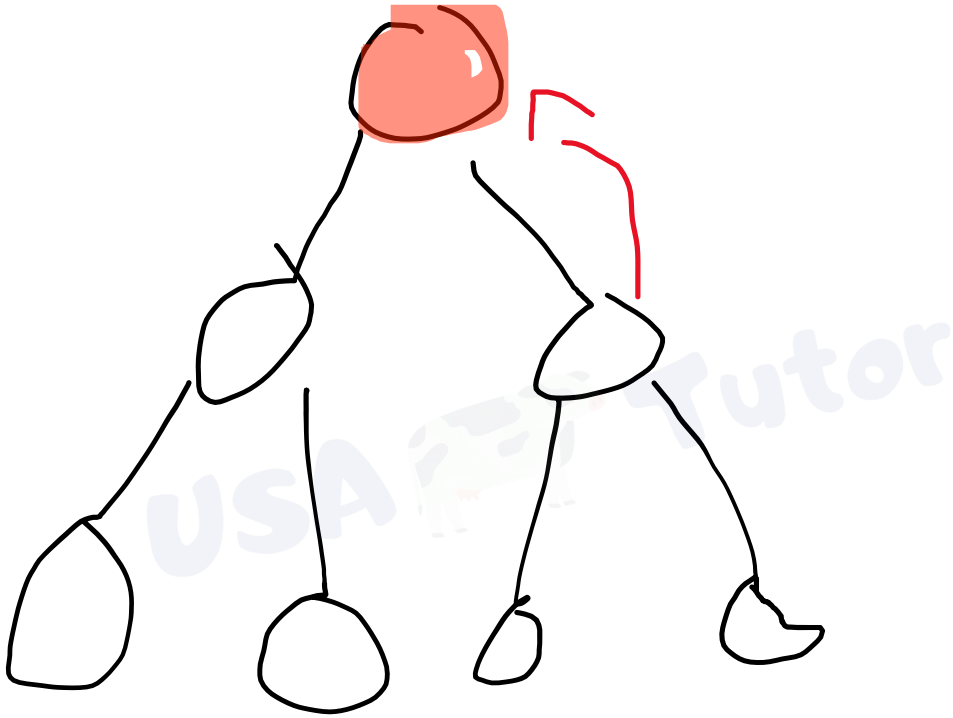
What is DFS?



What is DFS?



What is DFS?



Recursion

```
int recurse(int a){
if(a==0); return 0
int sum=a;
sum+=recurse(a-1);
return sum;
}
```

a(5) =

DFS Making your Graph

```
vector<int> adj[N];
for(int i=0; i<N; i++){
        int a, b; cin>>a>>b;
        a--; b--;
        adj[a].push_back(b);
        adj[b].push_back(a);
}
```

DFS Traversal

```
//this works ONLY for a tree
void dfs(int cur, int par){
        for(int next: adj[cur){
                if(next==par) continue;
                dfs(next, cur);
        }
}
```

DFS Traversal

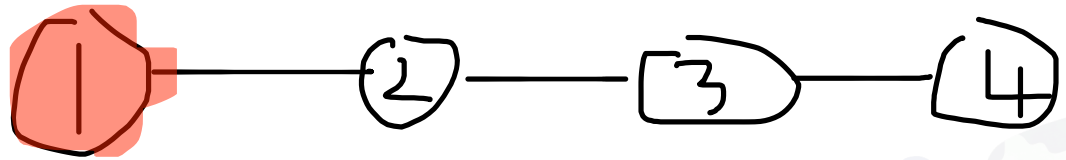
```
//this works on everything
bool visited[N];
void dfs(int cur){
        visited[cur]=true;
        for(int next: adj[cur){
                if(visited[next]) continue;
                dfs(next);
        }
}
```

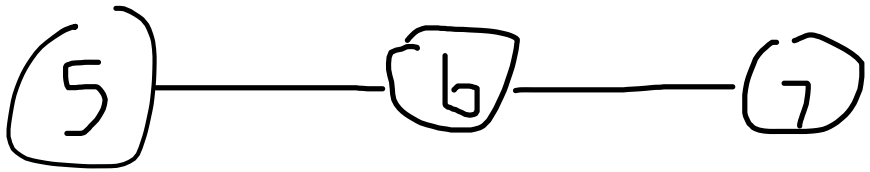
You’re done with DFS

Celebrate yay

Fence Planning 2019 US Open

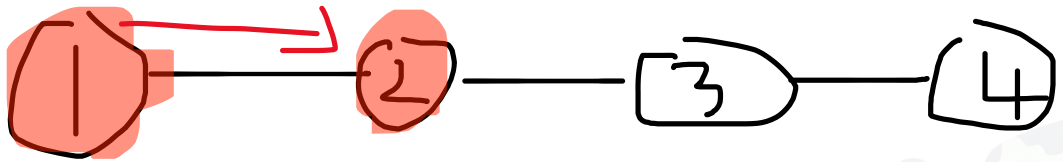
We need to find connected components… how do we do that?

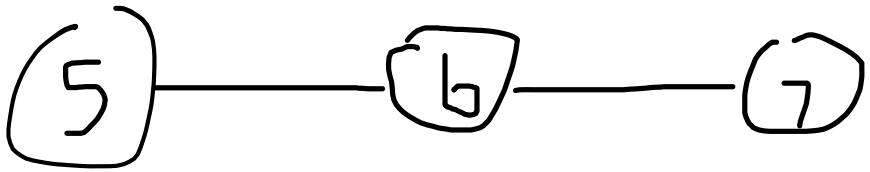




Fence Planning 2019 US Open

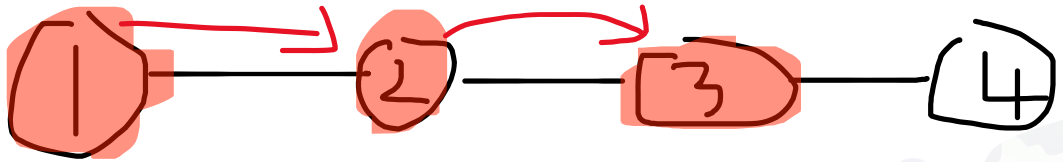
We need to find connected components… how do we do that?

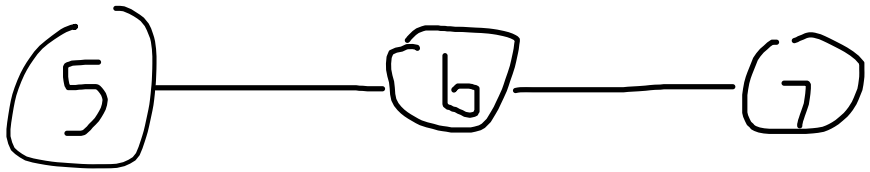




Fence Planning 2019 US Open

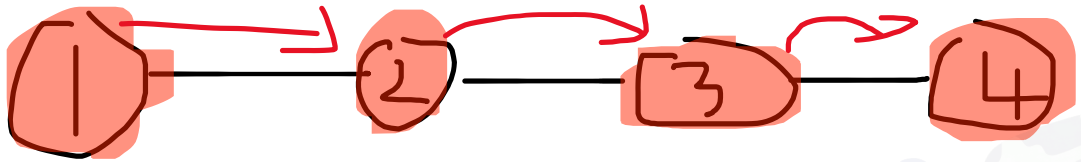
We need to find connected components… how do we do that?

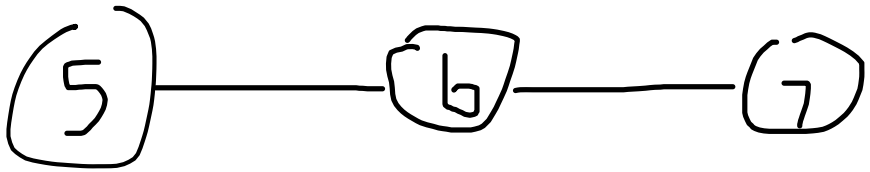




Fence Planning 2019 US Open

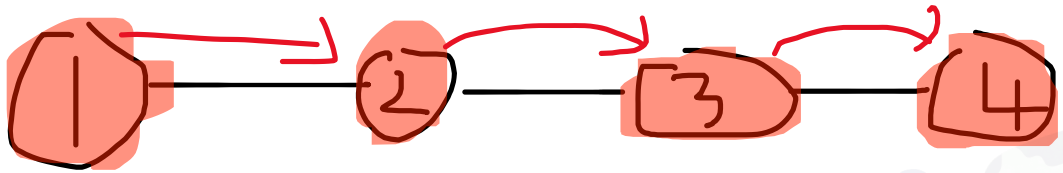
We need to find connected components… how do we do that?

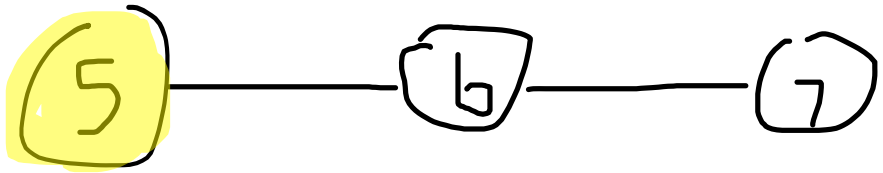




Fence Planning 2019 US Open

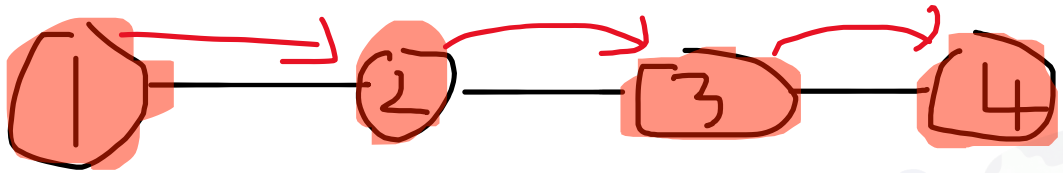
We need to find connected components… how do we do that?

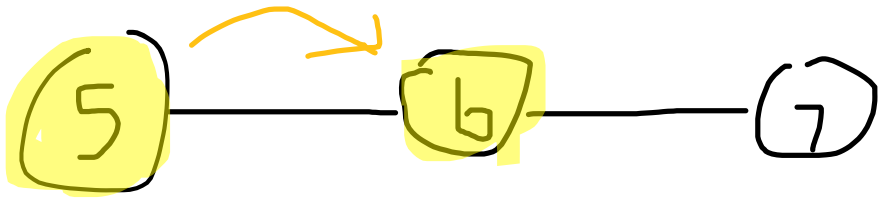




Fence Planning 2019 US Open

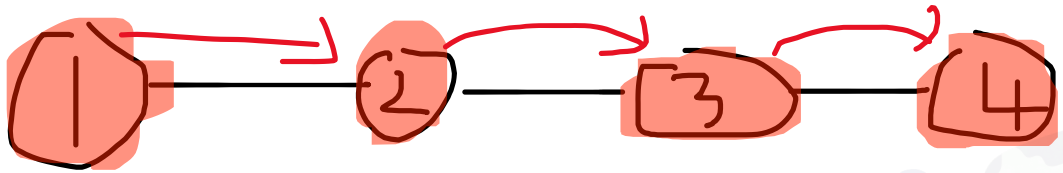
We need to find connected components… how do we do that?

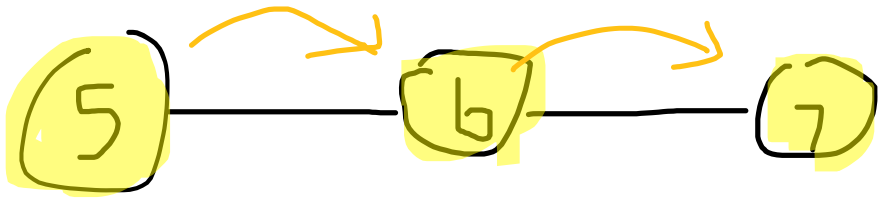




Fence Planning 2019 US Open

We need to find connected components… how do we do that?





Min Rectangle

Take the min and max of Xs

Take the min and max of Ys

Answer is (maX - miX)*2 + (maY – miY)*2

How do we incorporate this in DFS?

During any DFS, we must store

minX, maxX, minY, maxY as variables

(I would make these variables global, then reset every time you dfs)

After you are done configuring,

Just continue to dfs