B. Disjoint Sets Union 2

1 second

256 megabytes

standard input

standard output

Implement disjoint sets union data structure. You have to perform queries of two types:

* union https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0075.png?V=2.7.5u https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v — unite two sets that contain https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0075.png?V=2.7.5u and https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v, respectively;
* get https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v — find the set to which https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v belongs to, find the minimal and the maximal element of the set, and the total number of elements in it.

**Input**

The first line of the input contains two integers https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5n and https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006D.png?V=2.7.5m (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/002C.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006D.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0033.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/22C5.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0030.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/283/0035.png?V=2.7.51≤n,m≤3⋅105) — the number of elements and the number of queries. Next https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006D.png?V=2.7.5m lines contain queries, one per line. For a query union the line looks like union https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0075.png?V=2.7.5u https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0075.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/002C.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.51≤u,v≤n). For a query get the line looks like get https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5v (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0076.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.51≤v≤n).

**Output**

For each operation get you should output the result on a separate line in the respected order. Each result should consist of three integers: the minimal element, the maximal element and the number of elements in the set.

**Example**

**input**

**Copy**

5 11

union 1 2

get 3

get 2

union 2 3

get 2

union 1 3

get 5

union 4 5

get 5

union 4 1

get 5

**output**

**Copy**

3 3 1

1 2 2

1 3 3

5 5 1

4 5 2

1 5 5

#include<bits/stdc++.h>

#define int long long int

#define pb push\_back

#define pp pop\_back

#define pii pair<int,int>

#define vec vector<int>

#define mp make\_pair // DISJOINT SET UNION(DSU)

#define inf 1e10

#define F first

#define S second

using namespace std;

int parent[300001],rak[300001],mn[300001],mx[300001],siz[300001];

void make\_set(int v) // making a new set

{

parent[v]=v;

siz[v]=1;

mx[v]=v;

mn[v]=v;

}

int find\_set(int v) // finding the leader of the set containing v

{

if(parent[v]==v)

return v;

else

return parent[v]=find\_set(parent[v]);

}

void union\_set(int u,int v) // DOING UNION BY RANK

{

u=find\_set(u); // finding leader of the set containing u

v=find\_set(v); // finding leader of the set containing v

if(u!=v)

{

if(siz[u]<siz[v])

swap(u,v);

parent[v]=u;

siz[u]+=siz[v];

mx[u]=max(mx[u],mx[v]);

mn[u]=min(mn[u],mn[v]);

}

}

int32\_t main()

{

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

cout.tie(NULL);

int tt=1;

//cin>>tt;

while(tt--)

{

int n,m;

cin>>n>>m;

for(int i=1;i<=n;i++)

make\_set(i);

for(int i=1;i<=m;i++)

{

string s;

cin>>s;

if(s=="union")

{

int a,b;

cin>>a>>b;

union\_set(a,b);

}

else

{

int a;

cin>>a;

a=find\_set(a);

cout<<mn[a]<<" "<<mx[a]<<" "<<siz[a]<<"\n";

}

}

}

}