

TREE-2:-

You have already solved this challenge ! Though you can run the code with different logic !

Course	DS	Session	Tree 2	Question Information	Level 1	Challenge 71
--------	----	---------	--------	----------------------	---------	--------------

Problem Description

A new species is trying to rule the planet. This species is creating their own population outburst to dominate other species. It all started with 1 single member of the species. The population increases in tree-like fashion abiding by few rules as listed below.

- Single member is able to reproduce by itself.
- A new member is added to the population every minute.
- Every member is associated with integral name.
- Multiple members can share a common name.
- Every member has it's own reproduction capacity, that is maximum number of children it can reproduce.
- A member can start to reproduce only if all members older than it have exhausted their reproduction capacity.
- Level 0 is family tree of this species comprise of single member at the start of multiplication.
- Integral name of single member at the start is 0.
- The population grows level wise, where number of members at level i is dependent on reproduction capacity of members at prior level.

Given the integral name of new member and it's reproduction capacity that is added to the population, you have to find it's parent, level at which it is added and it's ascending age wise rank among siblings.

Input:
First line of the input contains 2 integers, N, RC_0 , representing number of minutes we will be examining the population increase and reproduction capacity of member at epoch. Next N line contains 2 integers each, ID_i, RC_i , representing integral name and reproduction capacity of new member born at time i .

Output:
 N lines, each line containing 3 integers, P, L, C , representing integral name of the parent, level at which it is added and it's ascending age wise rank among siblings.

Note :
It will always be possible to reproduce a new child or in other words, through out the given time, there exists atleast one member which can still accomodate new child.

Constraints:
 $1 \leq N \leq 10^6$
 $-10^9 \leq ID_i \leq 10^9$
 $0 \leq RC_i \leq 10^9$

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct cell{

    int name;

    int level;

    int capacity;

};

struct cell queue[1000001];

struct cell arr[1000001];

int front;

int end;

void init_queue(){

    front = 0;

    end = 0;

}

void enqueue(int name,int capacity,int level){

    queue[end].name = name;

    queue[end].level = level;

    queue[end].capacity = capacity;

    end = end + 1;
```

```

}

int is_empty(){
    if(end == front)
        return 1;
    return 0;
}

void dequeue()
{
    if(!is_empty())
        front++;
}

int main(){
    int n,rc;

    init_queue();

    scanf("%d %d",&n,&rc);

    int i,j,k;
    for(i=0;i<n;i++){
        scanf("%d %d",&arr[i].name,&arr[i].capacity);
    }

    enqueue(0,rc,0);

    i=0;

    while(!is_empty()){
        int par = queue[front].name;
        int cap = queue[front].capacity;
        int lev = queue[front].level+1;

        k=1;

        for(j=0;j<cap&& i<n;j++,i++){
            printf("%d %d %d\n",par,lev,k++);

            enqueue(arr[i].name,arr[i].capacity,lev);
        }

        dequeue();
    }

    return 0;
}

```

care.srmup.in/srmncrctelab/#srmncrctelab/student/home

You have already solved this challenge! Though you can run the code with different logic!

Course	DS	Session	Tree 2	Question Information	Level 1	Challenge 72
<p>Problem Description: Mancunian and Liverbird decide to go camping for the weekend after a long week at work. They came upon an unusual tree with N nodes while walking through a forest. From 1 to N, the vertices are numbered.</p> <p>A colour is allocated to each node in the tree (out of C possible colours). They decide to work together (for a change) and put their reasoning abilities to the test because they are bored. At vertex 1, the tree is rooted. They aim to locate the nearest ancestor with the same hue for each node.</p> <p>Constraints</p> <ul style="list-style-type: none"> $1 \leq N \leq 100,000$ $1 \leq C \leq 100,000$ <p>Input format The first line contains two integers N and C denoting the number of vertices in the tree and the number of possible colors. The second line contains $N-1$ integers. The ith integer denotes the parent of the $i+1$th vertex. The third line contains N integers, denoting the colors of the vertices. Each color lies between 1 and C inclusive.</p> <p>Output format Print N space-separated integers. The ith integer is the vertex number of lowest ancestor of the ith node which has the same color. If there is no such ancestor, print 1 for that node.</p>						
<p>Logical Test Cases</p> <div> <p>Test Case 1</p> <p>INPUT (STDIN)</p> <pre>5 4 1 1 3 3 1 4 2 1 2</pre> </div> <div> <p>Test Case 2</p> <p>INPUT (STDIN)</p> <pre>5 5 1 2 3 3 4 1 3 4 4 2</pre> </div>						

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

using namespace std;

int main() {

    int n,i,c;

    scanf("%d %d", &n, &c);

    int tree[n+1][2];

    tree[1][0] = -1;

    for(i=2;i<=n;i++) {

        scanf("%d", &tree[i][0]);

    }

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

        scanf("%d", &tree[i][1]);

    }

    int parent;

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

        parent = tree[i][0];

        while(parent != -1 && tree[parent][1] != tree[i][1]) {

            parent = tree[parent][0];

        }

        printf("%d ", parent);

    }

    return 0;

}
```

care.srmup.in/srmncrctelab/#/srmncrctelab/student/home

You have already solved this challenge! Though you can run the code with different logic!

Course	DS	Session	Tree 2	Question Information
				Level 1 Challenge 73

Question description

A beautiful code of a tree of n nodes is a sequence of $n-2$ integers that uniquely specifies the structure of the tree.

The code is constructed as follows: As long as there are at least three nodes left, find a leaf with the smallest label, add the label of its only neighbour to the code, and remove the leaf from the tree.

Given a beautiful code of a tree, your task is to construct the original tree.

Constraints

- $3 \leq n \leq 10^5$
- $1 \leq a_i \leq n$

Input

The first input line contains an integer n : the number of nodes. The nodes are numbered $1, 2, \dots, n$.

The second line contains $n-2$ integers: the beautiful code.

Output

Print $n-1$ lines describing the edges of the tree. Each line has to contain two integers a and b : there is an edge between nodes a and b . You can print the edges in any order.

Logical Test Cases

Test Case 1 Test Case 2

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

using namespace std;

#define f(i,a,n) for(int i=a;i<n;i++)
#define X(a,b) if(a==b)

vector< int > vi;

const int maxN = 2e5+5;

int N, a[maxN], deg[maxN];

priority_queue<int, vector<int>, greater<int>> q;

int main(){

    scanf("%d", &N);

    fill(deg+1, deg+N+1, 1);

    //for(int i = 0; i < N-2; i++)

    f(i,0,N-2){

        scanf("%d", &a[i]);

        deg[a[i]]++;

    }

    //for(int i = 1; i <= N; i++)

    f(i,1,N+1)

        //if(deg[i] == 1)

        X(deg[i],1)
```

```

q.push(i);

f(i,0,N-2){
    int u = a[i];
    int v = q.top(); q.pop();

    deg[u]--; deg[v]--;
    //if(deg[u] == 1)
    X(deg[u],1)
    q.push(u);

    printf("%d %d\n", v, u);
}

//for(int i = 1; i <= N; i++)
f(i,1,N+1)
if(deg[i])
    printf("%d ", i);
}

```

care.srmup.in/srmncrrelab/#/srmncrrelab/student/home

You have already solved this challenge! Though you can run the code with different logic!

Course	DS	Session	Tree 2	Question Information
				Level 1 Challenge 74

Problem Description:
 You're given a K -ary infinite tree rooted at a vertex numbered 1. All its edges are weighted 1 initially.
 Any node X will have exactly K children numbered as:
 $[K * X + 0, K * X + 1, K * X + 2, K * X + 3, K * X + 4, \dots, K * X + (K - 1)]$
 You are given Q queries to answer which will be of the following two types:
 1. $1UV$: Print the shortest distance between nodes U and V .
 2. $2UVW$: Increase the weight of all edges on the shortest path between U and V by W .

Constraints
 $2 \leq K \leq 10$
 $1 \leq Q \leq 10^3$
 $1 \leq U, V \leq 10^{18}$
 $U \neq V$
 $1 \leq W \leq 10^9$

Input format
 • The first line contains two space-separated integers K and Q .
 • Next Q lines contain queries which will be of 2 types:
 • Three space-separated integers 1, U , and V
 • Four space-separated integers 2, U , V , and W

Output format
 For each query of type $(1UV)$, print a single integer denoting the shortest distance between U and V .

```

#include <iostream>

#include <map>

#include <assert.h>

```

```

using namespace std;

#define int long long

map < pair < int, int >, int > adj;

int find_depth( int u, int k ) {
    int depth = 0;
    while ( u > 0 ) {
        u = u / k;
        depth = depth + 1;
    }
    return depth - 1;
}

int dist( int u, int v, int k ) {
    int dist = 0;
    int depth_u = find_depth( u, k );
    int depth_v = find_depth( v, k );
    if ( depth_u < depth_v ) {
        swap ( u, v );
        swap ( depth_u, depth_v );
    }
    while( depth_u != depth_v ) {
        if ( adj.count( { u, u / k } ) ) {
            dist = dist + adj[ { u, u / k } ];
        } else {
            dist = dist + 1;
        }
        depth_u = depth_u - 1;
        u = u / k;
    }
    while ( u != v ) {
        if ( adj.count( { u, u / k } ) ) {
            dist = dist + adj [ { u, u / k } ];
        } else {
            dist = dist + 1;
        }
    }
    if ( adj.count( { v, v / k } ) ) {

```

```

        dist = dist + adj [ { v, v / k } ];

    } else {

        dist = dist + 1;

    }

    u = u / k;

    v = v / k;

}

return dist;

}

```

```

void add_weight( int vertex, int parent, int w ) {

    if ( !adj.count ( { vertex, parent } ) ) {

        adj[ { vertex, parent } ] = 1;

    }

    adj[ { vertex, parent } ] = adj[ { vertex, parent } ] + w;

}

```

```

void increase_weights ( int u, int v, int w, int k ) {

    int depth_u = find_depth( u, k );

    int depth_v = find_depth( v, k );

    if ( depth_u < depth_v ) {

        swap ( u, v );

        swap ( depth_u, depth_v );

    }

    while( depth_u != depth_v ) {

        add_weight( u, u / k, w );

        depth_u = depth_u - 1;

        u = u / k;

    }

    while ( u != v ) {

        add_weight( u, u / k, w );

        add_weight( v, v / k, w );

        u = u / k;

        v = v / k;

    }

}

```

```

signed main() {

```

```

int k, q, x, u, v, w;

cin >> k >> q;

while(q--) {

    cin >> x;

    if ( x == 1 ) {

        cin >> u >> v;

        cout << dist( u, v, k ) << "\n";

    } else {

        cin >> u >> v >> w;

        increase_weights( u, v, w, k );

    }

}

}

```

The screenshot shows a web browser window with the URL `care.srmup.in/srmncretelab/#/srmncretelab/student/home`. A green notification bar at the top states: "You have already solved this challenge! Though you can run the code with different logic!". Below this, a table shows the course as "DS", session as "Tree 2", and question information as "Level 1 Challenge 75".

Problem Description:

Football is Monk's favourite sport, and his favourite team is "Manchester United." Manchester United has qualified for the Champions League Final, which will take place at London's Wembley Stadium. As a result, he decided to go watch his favourite team play.

When he arrived at the stadium, he noticed that there was a long wait for match tickets. He is aware that the stadium has M rows, each with a distinct seating capacity. They could or might not be comparable. The cost of a ticket is determined by the row. If there are K (always higher than 0) empty seats in a row, the ticket will cost K pounds (units of British Currency).

Now, every football fan standing in the line will get a ticket one by one.

Given the seating capacities of different rows, find the maximum possible pounds that the club will gain with the help of the ticket sales.

Constraints:

- $1 \leq M \leq 1000000$
- $1 \leq N \leq 1000000$
- $1 \leq X[i] \leq 1000000$
- Sum of $X[i]$ for all $1 \leq i \leq M$ will always be greater than N .

Input:

The first line consists of M and N . M denotes the number of seating rows in the stadium and N denotes the number of football fans waiting in the line to get a ticket for the match.

Next line consists of M space separated integers $X[1], X[2], X[3], \dots, X[M]$ where $X[i]$ denotes the number of empty seats initially in the i^{th} row.

Output:

Print in a single line the maximum pounds the club will gain.

Logical Test Cases:

Test Case 1	Test Case 2
INPUT (STDIN)	INPUT (STDIN)
3 4	8 4

```

#include <bits/stdc++.h>

using namespace std;

#define PII pair<int, int>

priority_queue<int> seats;

map<int, int> x;

int main()

{

```



```

int N, M; cin >> N >> M;

assert (1<=N and N<=1000000);

assert (1<=M and M<=1000000);

for (int g=1; g<=N; g++){

    int a; cin >> a;

    seats.push(a);

    assert (1<=a and a<=1000000);

    x[a]++;

}

long long ans = 0;

for (int g=0; g<M; g++){

    int x = seats.top(); ans+=x; seats.pop();seats.push(x-1);

}

cout <<ans;

return 0;

cout<<"void heapify(int arr[],int n,int i)";

}

```

You have already solved this challenge! Though you can run the code with different logic!

Course	DS	Session	Tree 2	Question Information
				Level 1 • Challenge 76

Problem Description:
 You are given a rooted tree that contains N nodes. Each node contains a lowercase alphabet.

You are required to answer Q queries of type u, c , where u is an integer and c is a lowercase alphabet. The count of nodes in the subtree of the node u containing c is considered as the answer of all the queries.

Constraints

- $1 \leq N, Q \leq 10^5$
- $1 \leq u, v \leq N$
- c is a lowercase alphabet
- s_i is a lowercase alphabet for all $1 \leq i \leq N$
- 1 is the root node

Input format

- First line: Two space-separated integers N and Q respectively
- Second line: A string S of length N (where the i^{th} character of S represents the character stored in node i)
- Next $N-1$ line: Two space-separated integers u and v denoting an edge between node u and node v
- Next Q lines: An integer u and a space-separated character c

Output format
 For each query, print the output in a new line.

Note: It is guaranteed that the input generates a valid tree.

Explanation for test case 1
 Tree given in the sample input will look like that.

```

    1
   /
  (a)

```

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

```
using namespace std;
```

```
void dfs(int node,int parent,string &s,vector<vector<int>>&subroot,vector<vector<int>>& v1)
```

```
{
```

```

        //visited[node]=1;

        subroot[node][s[node-1]-'a']++;

        //intime[node]=t;

        //t++;

        //z.push_back(node);

        for( auto it:v1[node])
        {
            if(it!=parent)
            {
                dfs(it,node,s,subroot,v1);

                for(int i=0;i<26;i++)
                subroot[node][i]+=subroot[it][i];

            }

        }

        //outtime[node]=t;

        //t++;

    }

int main()

{

    int N,i, Q;

    string S;

    cin >> N >> Q;

    cin >> S;

    vector<vector<int>>>v1(N+1);

    for(i=0;i<N-1;i++)

    {

        int u, v;

        cin >> u >> v;

        v1[u].push_back(v);

        v1[v].push_back(u);

    }

    vector<vector<int>>>subroot(N+1,vector<int>(26,0));

    dfs(1,0,S,subroot,v1);

    while(Q--)
```

```

{

    int u;

    char c;

    cin >> u >> c;

    cout<<subroot[u][c-'a']<<"\n";

    //cout<<cnt<<endl;

}

return 0;

}

```

The screenshot shows a web browser window with the URL `care.srmup.in/srmncretelab/#/srmncretelab/student/home`. A green notification bar at the top states: "You have already solved this challenge! Though you can run the code with different logic!". The page content is organized into tabs: Course, DS, Session, Tree 2, Question Information, and Level 1 • Challenge 77. The "Question Information" tab is active, showing the following details:

Problem Description
You are given a weighted graph with N vertices and M edges. Find the total weight of its maximum spanning tree.

Constraints

- $1 \leq T \leq 20$
- $1 \leq N \leq 5000$
- $1 \leq M \leq 100\,000$
- $1 \leq c \leq 10\,000$

Input
The first line contains one integer T denoting the number of test cases. Each test case starts with a line containing 2 space-separated integers: N and M. Each of the following M lines contain description of one edge: three different space-separated integers: a, b and c. a and b are different and from 1 to N each and denote numbers of vertices that are connected by this edge. c denotes weight of this edge.

Output
For each test case output one integer - the total weight of its maximum spanning tree.

Logical Test Cases

Test Case 1	Test Case 2
INPUT (STDIN)	INPUT (STDIN)
1	1
3 3	3 3
1 2 2	1 2 3
2 3 3	2 3 4
1 3 4	1 3 5

```

#include<bits/stdc++.h>

typedef long long ll;

using namespace std;

struct edge
{

    int u;

    int v;

    int w;

};

edge a[100005];

int parent[100005];

```

```

bool comp (edge a , edge b)

{

    return a.w>b.w;

}

int find_parent(int u) ///DSU find

{

    /* return (parent[u]==u) ? u: find_parent(parent[u]);*/

    if(parent[u]==u)

        return u;

    else

        return parent[u]=find_parent(parent[u]);

}

void merge(int u, int v) /// DSU union

{

    parent[u]=v;

}

int main()

{

    int t;

    cin>>t;

    while(t--){

        int n,m;

        cin>>n>>m;

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

            parent[i]=i;

        for(int i=0;i<m;i++) {

            cin>>a[i].u>> a[i].v >> a[i].w;

        }

        sort(a,a+m,comp);

        ll ans=0;

        for(int i=0;i<m;i++) {

            int x=find_parent(a[i].u);

            int y=find_parent(a[i].v);

            if(x!=y)

            {

                merge(x,y);

                ans+=a[i].w;

            }

        }

    }

```

```

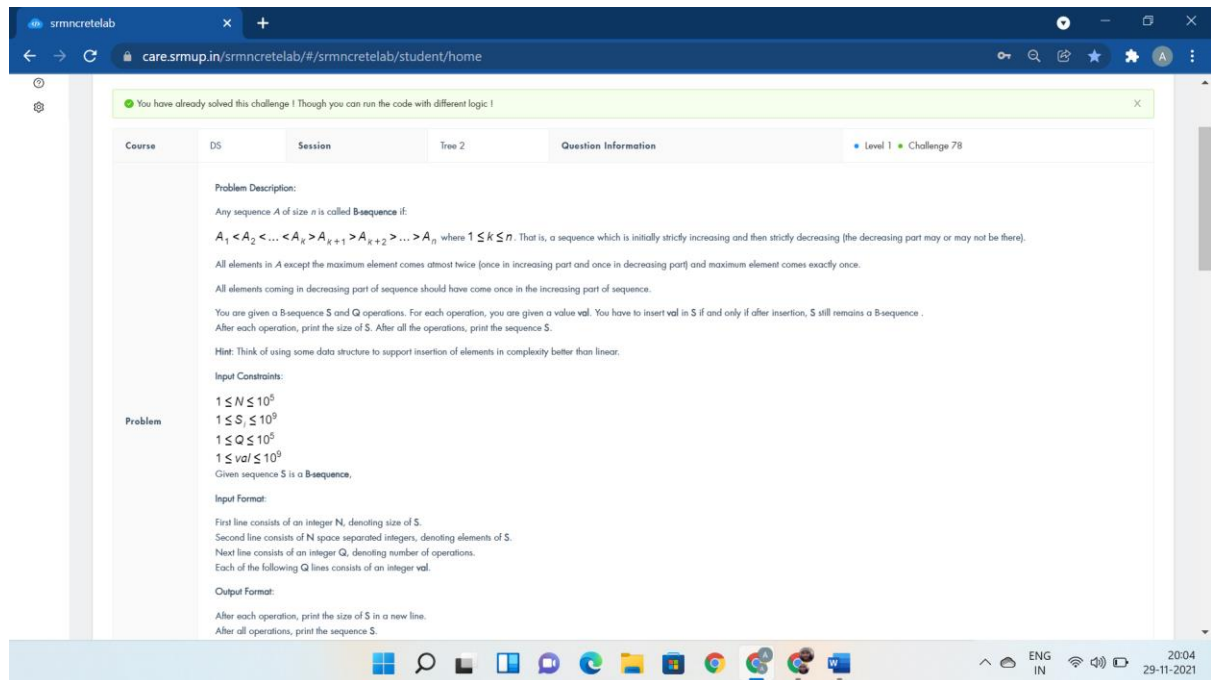
    }

    cout<<ans<<endl;
}

return 0;

cout<<"int printheap(int N)";
}

```



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

```
#include<map>
```

```
using namespace std;
```

```
int main() {
```

```
    int N,i,maximum=INT_MIN;
```

```
    scanf("%d", &N);
```

```
    int S[N];
```

```
    map<int,int> map;
```

```
for(i=0;i<N;i++) {

    scanf("%d", &S[i]);

    maximum=max(maximum,S[i]);

    map[S[i]]++;

}

int temp,Q;

cin>>Q;

for(i=0;i<Q;i++) {

    scanf("%d", &temp);

    if(temp==maximum) printf("%d\n",N);

    else {

        if(map[temp]>=2) printf("%d\n",N);

        else {

            map[temp]++;

            N++;

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

            maximum=max(maximum,temp);

        }

    }

}
```

```
}
```

```
for(auto it=map.begin();it!=map.end();it++) printf("%d ",it->first);
```

```
for(auto it=map.rbegin();it!=map.rend();it++) {
```

```
    if(it->second>1) printf("%d ",it->first);
```

```
}
```

```
}
```

The screenshot shows a web browser window with the address bar displaying 'care.srmup.in/srmncretelab/#/srmncretelab/student/home'. The page content includes a notification at the top: 'You have already solved this challenge! Though you can run the code with different logic!'. Below this, the page is titled 'Level 1 Challenge 79'. The main content area is divided into sections: 'Question Description', 'Input', 'Output', and 'Constraints'. The 'Question Description' section states: 'In a movie festival, n movies will be shown. You know the starting and ending time of each movie. Your task is to process q queries of the form: if you arrive and leave the festival at specific times, what is the maximum number of movies you can watch? You can watch two movies if the first movie ends before or exactly when the second movie starts. You can start the first movie exactly when you arrive and leave exactly when the last movie ends.' The 'Input' section states: 'The first input line has two integers n and q: the number of movies and queries. After this, there are n lines describing the movies. Each line has two integers a and b: the starting and ending time of a movie. Finally, there are q lines describing the queries. Each line has two integers a and b: your arrival and leaving time.' The 'Output' section states: 'Print the maximum number of movies for each query.' The 'Constraints' section lists: '1 ≤ n, q ≤ 2 · 10^5' and '1 ≤ a ≤ b ≤ 10^6'. Below the constraints, there are two tabs for 'Logical Test Cases': 'Test Case 1' and 'Test Case 2'. The browser's taskbar at the bottom shows various application icons and the system clock indicating 20:05 on 29-11-2021.

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

```
using namespace std;
```

```
int dp[1000006][25];
```

```
void solve(){}
```

```
int main(){
```

```
    solve();
```

```
    int n, q; cin>>n>>q;
```

```
    for (int i = 0; i < n; i++) {
```

```
        int x, y; cin>>x>>y;
```

```

        dp[y][0] = max(dp[y][0], x);
    }

    for (int i = 1; i <= 1000000; i++)

        dp[i][0] = max(dp[i][0], dp[i-1][0]);


    for (int k = 1; k <= 20; k++)

        for (int i = 1; i <= 1000000; i++)

            dp[i][k] = dp[dp[i][k-1]][k-1];


    while(q--){

        int x,y; cin>>x>>y;

        int ans = 0;

        while(y>0){

            int z = 0;

            for (int i = 0; i <= 20; i++){

                if (dp[y][i] < x){

                    z = i;

                    break;

                }

            }

            if (z == 0)

                break;

            ans += (1<<(z-1));

            y = dp[y][z-1];

        }

        cout<<ans<<endl;

    }

}

```


srmmcretelab

care.srmup.in/srmmcretelab/#/srmmcretelab/student/home

You have already solved this challenge! Though you can run the code with different logic!

Course	DS	Session	Tree 2	Question Information	Level 1	Challenge 80
<p>Problem Description</p> <p>M is alone and he has an array a_1, a_2, \dots, a_n. M wants to choose two integers i, j such that $i \neq j, 1 \leq i, j \leq n$ and the value $a_i \& a_j$ (bitwise AND) is maximum.</p> <p>What is the maximum value M can get?</p> <p>Input</p> <p>First line contains only n, length of array.</p> <p>Second line contains the array elements a_1, a_2, \dots, a_n separated by space.</p> <p>$1 \leq n \leq 3 \times 10^5$</p> <p>$1 \leq a_i \leq 10^9$</p> <p>Output</p> <p>The only line of output contains an integer, maximum value value that M can get.</p>						
<p>Logical Test Cases</p> <div> <p>Test Case 1</p> <p>INPUT (STDIN)</p> <p>4</p> <p>3 4 2 3</p> <p>EXPECTED OUTPUT</p> <p>3</p> </div> <div> <p>Test Case 2</p> <p>INPUT (STDIN)</p> <p>5</p> <p>1 2 3 2 5</p> <p>EXPECTED OUTPUT</p> <p>2</p> </div>						

20:05 29-11-2021

```
#include<stdio.h>

#include<stdlib.h>

#include<math.h>

void input(long *,int);

int main()

{

    int n;


    scanf("%d",&n);

    long *ptr = (long*)malloc(n*sizeof(long));

    input(ptr,n);

    return 0;

}

void input(long *ptr, int n)

{

    int i, j;

    int m;

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

    {

        scanf("%ld", ptr+i);

    }

}
```

```

for(i = 0; i < n; i++)
{
    if (*(ptr + i) <= m)
    {
        continue;
    }

    for (j = i + 1; j < n; j++)
    {
        int temp = *(ptr + i) & *(ptr + j);

        if(temp > m)
        {
            m = temp;
        }
    }
}

printf("%d", m);
}

```

GRAPH:-

The screenshot shows a web browser window with the URL `care.srmup.in/srmncretelab/#/srmncretelab/student/home`. A notification at the top states: "You have already solved this challenge! Though you can run the code with different logic!". The page content is organized into a table with columns: Course, DS, Session, Graph, Question Information, and Level 1 Challenge 81.

Question description
 Consider a network consisting of n computers and m connections. Each connection specifies how fast a computer can send data to another computer. Kotivalo wants to download some data from a server. What is the maximum speed he can do this, using the connections in the network?

Input
 The first input line has two integers n and m : the number of computers and connections. The computers are numbered $1, 2, \dots, n$. Computer 1 is the server and computer n is Kotivalo's computer.

After this, there are m lines describing the connections. Each line has three integers a , b and c : computer a can send data to computer b at speed c .

Output
 Print one integer: the maximum speed Kotivalo can download data.

Constraints

- $1 \leq n \leq 500$
- $1 \leq m \leq 1000$
- $1 \leq a, b \leq n$
- $1 \leq c \leq 109$

Logical Test Cases

Test Case 1	Test Case 2
INPUT (STDIN)	INPUT (STDIN)
4 5	4 5

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

```
using namespace std;
```