GUB_FlightNo:MH370

Md. Hosain Rohman Noyon

Ismatul Islam Pranto

Samim Reza

Green University of Bangladesh

Table of Contents

Template

```
#include<bits/stdc++.h>
#define ff first
#define ss second
using namespace std;
typedef long long ll;
typedef long double ld;
typedef unsigned int uint;
typedef unsigned long long
ull;
using pll = pair<ll, ll>;
typedef vector<int> vi;
typedef vector<ll> vl;
typedef vector<pll> vpll;
typedef vector<vl> matrix;
typedef vector<bool> vb;
#define PI acos(-1)
#define endl "\n"
#define pb push_back
#define ppb pop_back
#define lb lower_bound
#define ub upper_bound
#define for0(n) for(int
i=0;i<n;i++)
#define for1(n) for(int
i=1;i<=n;i++)
#define for0j(n) for(int
j=0;j<n;j++)
#define for1j(n) for(int
j=1;j<=n;j++)
#define each(v) for (auto
&it : v)
#define all(x) (x).begin(),
(x).end()
#define rall(x)
(x).rbegin(), (x).rend()
#define MOD 1000000007
#define MAX LLONG_MAX
#define MIN LLONG_MIN
#define mp make_pair
#define MEM(a,x)
memset(a,x,sizeof(a))
//Debugging Functions starts
template <class T> void
print(T x) {cerr << x;}
template <class T, class V>
void print(pair<T, V> x){
print(x.ff); cerr << ':';
print(x.ss);}
template <class T> void
print(vector<T>
&a){cerr<<'['<<' '; each(a)
{ print(it); cerr << '
';}cerr << ']';}
template <class T> void
print(stack<T> x) {cerr<<"[
";while(!x.empty()){cerr<<x.
top()<<", ";
x.pop();}cerr<<"]";}
template <class T> void
print(queue<T> x) {cerr<<"[
";while(!x.empty()){cerr<<x.
front()<<", ";
x.pop();}cerr<<"]";}
template <class T> void
print(set<T>
&a){cerr<<'['<<' ';each(a){
print(it); cerr << ' ';
}cerr << ']';}
template <class T> void
print(set<T, greater<T>>
&a){cerr<<'['<<' '; each(a)
{print(it);cerr<<'
';}cerr<<']';}
template <class T> void
print(multiset<T> &a)
{cerr<<'['<<' ';
each(a){print(it);cerr<<'
';} cerr<<']';}
template <class T> void
print(multiset<T,
greater<T>> &a){cerr<<'['<<'
'; each(a){print(it);cerr<<'
';}cerr<<']';}
template <class T> void
print(unordered_set<T>
&a){cerr << '[' << ' ';
each(a){print(it);cerr<<'
';}cerr<<']';}
template <class T, class V>
void print(vector<pair<T,
V>> &a){cerr<<'['<<' ';
each(a){print(it.ff);cerr<<"
:";print(it.ss);cerr<<'
';}cerr<<']';}
template <class T, class V>
void print(map<T, V>
&a){cerr<<"[ ";
each(a){print(it);cerr<<"
";}cerr<<"]";}
template <class T, class V>
void print(unordered_map<T,
V> &a){cerr<<"[ ";
each(a){print(it);cerr<<"
";}cerr<<"]";}
template <class T> void
print(vector<vector<T>>
&a){cerr<<"[ ";
each(a){print(it);cerr<<"
";}cerr<<"]";}
//Debugging Functions ends
```

FastIO

```
void speed(){
ios_base::sync_with_stdio(fa
lse);  cin.tie(NULL);
cout.tie(NULL);}
```

File I/O

```
void file(){
#ifndef ONLINE_JUDGE
freopen("input.txt", "r",
stdin);
freopen("output.txt", "w",
stdout);
freopen("error.txt", "w",
stderr);
#define dbg(x) cerr << #x <<
" "; print(x); cerr << '\n';
#else
        #define dbg(x)
#endif
}
```

Graph Grid Possible moves

```
bool ifExist(ll x, ll y){
  if(x>=0 && y>=0 &&
x<=Right && y<=Right) return
true;
```

```
    return false;
}
ll x[]={1,-1,0,0,1,-1,1,-1};
ll y[]={0,0,1,-1,1,-1,-1,1};
[To run at terminal]
win/g++ main.cpp && ./a.exe
lin/g++ main.cpp && ./a.out
```

Bitwise Operation

```
bitset<n> x;
bitset<n> x(intVal);
x[1] = 1; x[2] = 0;
/Note x[0-4] valid
cout << x << endl;
Output: 00010
Getbit: n&(1LL<<i)
Setbit0:n&(~(1LL<<i))
Setbit1: n|(1LL<<i)
Clearbit:n&(~(1ll<<i))
Togglebit: n^(1LL<<i)
RightLast1: n&(-n)
Swap two numbers a, b:
    a = a^b;
    b = a^b;
    a = a^b;
Check if N is power of 2:
 if(n&(n-1)==0)
   cout << "YES"<< endl;
Uppercase to lowercase:
   char f = ch | 32;
Lowercase to uppercase:
   char g = ch & ~32;
Toggle case:
   char e = 'a' ^ (1<<5);
   char h = 'a' ^ 32;
Clear LSB till nth bit:
    ll b = n &
(~((1<<(i+1))-1));
Clear MSB till nth bit
 ll c=n & ((1<<(i+1))-1);
Multiply by 2:
cout << (a<<1) << endl;
Divide by 2:
cout << (a>>1) << endl;
isPalindrome(ll n){
   ll reverse = 0;
   ll temp = n;
  while (temp > 0){
    reverse = (reverse <<
1) | (temp & 1);
    temp >>= 1;
  }
  return reverse == n;}
void printBinary(ll num){
for(ll i=63; i>=0; i--)
  cout << ((num>>i)&1);
cout << endl;
}
```

Built in Function

```
No. of leftside zero:
__builtin_clz(x);
__builtin_clzll(x);
No of rightside zero:
__builtin_ctz(x);
__builtin_ctzll(x);
No. of 1 bit:
__builtin_popcount(x);
__builtin_popcountll(x);
Parity (if odd no of
1bits or not)
__builtin_parity(x);
Infinity
__builtin_inf();/double
(int)__builtin_infd32();
(ll int)__builtin_infd64();
```

STL

Pair

```
pair<int, char> p1;
p1.first, p1.second;
p1= make_pair(1,'a');
p1= {1,'a'};
```

Vector

```
vector<ll> v;
vector<ll> v(size);
v.insert(v.begin()+ind,
val);
v.push_back(n);
v.pop_back();
v.front();   v.back();
v.size();   v.empty();
v.erase(iterator);
v.begin(), v.end();
sort(v.begin(),v.end());
sort(v.begin(),v.end(),
greater<ll>());
auto it =
find(v.begin(),
v.end(), num);
for(auto n: v) /loop
vector<pair<ll,ll>> v;
v.push_back({x,y});
v[ind].first;
v[ind].second;
*max_element(all(v));
```

Set(unique value)

```
/O(log n) -> BST
set<ll>s;
set<ll,greater<ll>>s =
{3,2,1,4}
/output : 4 3 2 1

s.insert(n); s.clear();
s.size();    s.empty();
s.count(n);  s.find(n);
s.erase(x); /delete all
same value
s.erase(s.find(x));
/delete 1 same value
for(auto n: s) /loop
s.begin(), s.end();
s1.swap(s2);
s2.insert(s1.begin(),s1
.end());
*st.begin(); /minimum n
*st.rbegin();/maximum n
s.erase(s.begin())
/first n delete
s.erase(--s.end())
/last n delete
```

UnOrder Set(Not sorted)

```
unordered_set<ll> s;
```

Multi Set(Duplicate value with sorted)

```
multiset<int> ms1;
multiset<int,
greater<int>> ms1;
auto it= ms1.begin();
it++;
```

Map(Sorted & No Duplicate key)

```
O(log n) -> BST
map<ll,ll> m;
map<int, int> mp3 =
{{1, 2}, {3, 4}, {5,
6}};
m.insert({x,y});
/{key,n}
print->m[key];
m.find(x);
for(auto it: m) /loop
it.first, it.second;
it->first,it->second;
/for outside loop
it = m.lower_bound(x)
it->first;
m.upper_bound(x)->first;
```

Pair Map

```
map<int, pair<int,
int>> mp1;
mp1[a].first,
mp1[a].second;

/if value for key is
not assigned Output:
/for int 0,
/for pair<int,int> it
is {0, 0}
map<pair<int, int>,
int> mp2;
mp2[{1, 3}] = -1;
cout<<mp2[{1,3}]; /-1

/Pair,int map loop
for(pair<pair<int,
int>, int> p: mp4)
cout<<p.first.first<< "
"<<p.first.second<<" "
<<p.second;

cout << mp1.size();
/map key finding:
if(mp.find(0) ==
mp.end()) /Not found
else /found
```

Upper bound & lower bound:

```
>For vector:
index =
lower_bound(v.begin(),
v.end(), num)- v.begin();
index =
upper_bound(v.begin(),
v.end(),num)-v.begin();

>For array:
index = lower_bound(A,
A+n, num)-A;
index = upper _bound(A,
A+n, num)-A;
```

Compare for sort()

```
/Descending

Sort(v.begin(),v.end(),
greater<ll>()):

/PairSort 2nd element

bool
pairSort(pair<ll,ll>
&a, pair<ll,ll> &b){

if(a.second==b.second)
return (a.first <
b.first);

return (a.second <
b.second);}

/Vector Row sort

vector<vector<ll>> v(sz);
bool sortcol(const
vector<ll>& v1, const
vector<ll>& v2){
  return v1[0] < v2[0];
}
/Call sort function
sort(all(v),funName);
```

Stack:

```
stack<int> st;
st.push(x);
while(!st.empty()) {
cout<< st.top();
st.pop(); }
```

Queue:

```
queue<int> q;
q.push(x);    q.pop();
q.front();    q.back();
```

Priority Queue:

```
Descending order:
priority_queue<int>
pq1;
Ascending order:
priority_queue<int,
vector <int>, greater
<int> > pq2;
pq2.push(x);
while(!pq.empty()){
   cout<<pq.top();
   pq.pop();
}
```

DeQue:

```
deque<int> dq;
   dq.push_back(10);
   dq.push_front(20);
   dq.pop_front();
   dq.pop_back()
dq.insert(it, val);
dq.at(ind); dq.size();
dq.front(); dq.back();
dq.clear();
dq.erase(it);
for(auto it=dq.begin();
it!=dq.end();++it)
   cout<< *it<<" ";
```

Algorithms:

```
>sort(vec.begin(), vec.end(), [](int x, int y){return x < y;});
>copy_if(foo.begin(), foo.end(), bar.begin(), [](int i){return !(i<0);}) /copy positive val from foo to bar vec
>find_if(all(v), IsOdd)
bool IsOdd (int i) {
  return ((i%2)==1);}
/get index where predicate first true
>binary_search(all(v), item) /returns true if found
>is_sorted(all(foo));
>reverse(all(vec));
>is_permutation (all(foo), bar.begin())
>*min_element(all(vec))
>*max_element(all(vec))
> do{
perms.push_back(arr);
      } while
(next_permutation(all(arr)));
>unique_copy (all(v1),v2.begin());
Function pointer --> int (*funcPtr)()
#undef: Un-defines a text macro
#ifdef: Same as #if defined(...)
#ifndef: Same as #if !defined(...)
#endif: Used to end an #if, #ifdef, or #ifndef
```

String

```
Transformation:
transform(S.begin(), S.end(), S.begin(), ::toupper);
transform(S.begin(), S.end(), S.begin(), ::tolower);
strcpy(destStr,srcStr);
int n = stoi(str);
double n = stod(str);
string s= to_string(n);
s = s1+s2; /merge
string s1 = "Hello World";
/get substr
subS = s1.substr(7,5);
/Output: World
subS = s1.substr(7)
/Output: World
/replace substr
s1.replace(7,5,"Uni");
s1.replace(s1.find(s2), s2.length(),newString);
s.resize(oldSize+Value),'+');
s.empty();
s.at(index);
s.find(subS);
s.rfind(subS);
swap(s1,s2);
str.size();
s1.compare(s2);s1>s2=1, s1<s2=-1,  s1==s2=0

/If we want to take input with space:
char c;
cin >> c;
getline(cin, s)
s = c + s;
sort(s.rbegin(), s.rend()) /sort in non-increasing order
int n = unique(s.begin(),s.end()) - sbegin(); /Getting unique char in string
```

```
max / min char:
*max_element(s.begin(), s.end());
*min_element(s.begin(), s.end());

/delete substr from string:
s.erase(s.begin() + 3, s.end()+7)  /remove loWo from helloworld
/push any substr
string Tmp = "hello alam hello";
string S = "tasdid";
copy(tmp.begin() + 6, tmp.begin()+10, back_inserter(s)) ;
/Output: tasdid alam
/remove all specific char
s.erase(remove( s.begin(), s.end(), 'a' ), s.end());
/check given string is substring or not
O(n*m) complexity
if(s.find("hello") != -1 )item found
else not found
```

Formulas

Triangle

Perimeter 2s: a+b+c

s is Half of Perimeter

Area:

$$\sqrt{s(s-a)(s-b)(s-c)}$$

/Any two sides a,b and the angle between them:

Area = ½ a*b*sin**θ**

Right angle:

Area: (b*h)/2

Pythagoras: $c^2 = a^2 + b^2$

Equilateral:

Height: $\frac{\sqrt{3}}{2}a^2$

Area: $\frac{\sqrt{3}}{4}a^2$

Isosceles:

a = len of equal sides,

b = base,

Height: sqrt($4a^2$-b^2)/2

Area: (b/4)*sqrt($4a^2$-b^2)

Quadrilateral

Rectangle :

Perimeter: 2(l+w)

Length: P/2 – w

Width: P/2 - l

Area: l*w

Parallelogram :

Perimeter: 2(len+base)

Area: base * height

Square :

Perimeter: 4*a

Area: a*a

diagonal: $\sqrt{2}a$

Rhombus :

Perimeter: 4*a

Area: (d_1 * d_2) / 2

d = diagonal

Trapezium :

Perimeter: a+b+c+d

Area: (a + b) × h/2

For all:

Area: 2 * TriangleArea

Circle

Circumference= 2*pi*r

Area = pi * r^2

Part Area= (θ/360)pi*r^2

Arc len, s = $\frac{pi * r * \theta}{180}$

Ellipse

Perimeter = pi*(a+b)

Area = pi * a * b

Hexagon

Perimeter= 6*a

Area = na^2cot(180/n)/4n

Area = 6 * TriangleArea

{triangle is Equilateral}

3D Shape

CS = Curved suface

LS = Curved/lateral s

TS = Total surface

Cuboid-Rectangle:

Diagonal: sqrt(a^2+b^2+c^2)

TS Area:2(ab+bc+ca)

LS area: 2h(l+b)

Volume: abc

Cube-Square:

Volume: a^3

LS area: 4a^2

TS area: 6a^2

Sphere-Circle:

Volume: (4/3)*pi*r^3

LS area: 4*pi*r^2

TS area: 4*pi*r^2

Hemisphere-Half_circle:

Volume: (2/3)*pi*r^3

LS area: 2*pi*r^2

TS area: 3*pi*r^2

Cylinder:

Volume: pi*r^2*h

LS area: 2*pi*r*h

TS area: 2*pi*r(r+h)

Cone:

Volume: (1/3)*pi*r^2*h

LS area: pi*r*l

TS area: pi(r+l)

Prism

Volume of a triangular prism: area of triangle × Height = (1/2 base × height) × Height;

base: length of the base of the triangle

height: height of the triangle

Height: height of the triangular prism

Lines:

Straight line eqn,

y = mx + c;

IntersectionRatio,m1:m2

X = (m1x2+m2x1)/(m1+m2)

y = (myx2+m2y1)/(m1+m2)

slope,m=(y2-y1)/(x2-x1)

slope,m = tan(theta)

Perpendicular lines,

m1*m2 = -1

Vertical line,m= inf

horizontal line,m=0

/Given a straight line, find the parallel and perpendicular line:

>Parallel:change C to K

>Perpendicular: swap a and b with sign (bx-ay+k=0)

>Solution of x,

x=(-b+-sqrt(b^2-4ac))/2a

Combinatorics:

Permutation(Order matters):

>Repetition not allowed

P(n,r):n!/(n-r)!

>Repetition allowed ,

P(n,r): n^r

Combinations:

>Number of arrangements

C(n) = n!

>With r elements,

Without repetition

C(n,r) = n!/(n-r)!r!,

>With repetition:

C(n,r)= (n+r-1)!/(n-1)!r!

Example: Flavors are chocolate, vanilla, and pineapple. If the person can select two scoops at a time, then he can have one flavor two times

Circular permutations,

>when clockwise and anticlockwise orders are same: (n-1)!/2

when clockwise and anticlockwise orders are different: (n-1)!

From n points:

>Straight line:

nC2 = n(n-1)/2

>Triange = nC3

>Ractacangle = nc4

>Diagonal =

nC2-n = (½)n(n-3)

>To invite 1 or more from N friends= $(2^{n})-1$

Sequence and Series Formula

>Nth term:

a+(n-1)d

$ar^{(n-1)}$

>Sum of Nth term:

n/2(2a + (n-1)d)

$a(1-r^{n})/(1-r)$ /if r<1

$a(r^{n} -1)/(r-1)$ /if r>1

>Natural number

N*(N+1)/2

>Square of Natural n

[n(n+1)(2n+1)] / 6

>Sum of squares of first n even numbers

[2n(n + 1)(2n + 1)]/3

>Sum of squares of first n odd numbers

[n(2n+1)(2n-1)] / 3

MATH

* if x and m coprime, x ^ phi(m) = 1 (mod m)

* if x and m coprime, x ^ n = x ^ (n mod phi(m)) (mod m)

ALGORITHM

MOD

>(a+b)%mod = ((a%mod)+(b%mod)) %mod;

>(a-b)%mod = ((a%mod)-(b%mod)+mod) %mod;

>(a*b)%mod = ((a%mod)*(b%mod)) %mod;

>(a/b)% mod =

$((a\%mod)*(b^{-1}\%mod))$ %mod; [NB:exist if __gcd (b, mod) == 1]

$>b^{-1}\%mod = x$

or, 1 = (b*x)%mode

$>B^{-1}$ % mod = $(B^{mod\ -\ 2}$ % mod)

> (a/b)%mod = (a%mod)(binPow(b,mod-2,mod))

/GCD = __gcd(a,b);

/LCM =(a*b)/gcd(a,b);

BIG MOD

```
ll binPow(ll a, ll b,
ll m){
if(b==0) return 1;
if(b==1) return a% m;
ll ans = binPow(a, b/2,
m);
ans = (ans * ans)% m;
if(b%2==1) ans = (ans *
a) % m;
return ans ;}
```

Inverse mod

```
ll inv(ll a,ll m) {
return (binPow(a, m-2,
m)%m);}
```

Extended Euclidean Algorithm

Question: Given, a,b,c for eq ax + by = c; x,y ?

Ans: ax + by = gcd(a, b);

gcd(a, b) = gcd(b, a%b);

gcd(b, a%b) = bx1 + (a%b)y1;

a%b = a - (a/b) * b;

From the above equations we get,

ax + by = bx1+ (a%b)y1;

ax + by = bx1 + (a - (a/b) * b)y1;

ax + by = ay1 + b(x1 - (a/b) * y1);

Comparing the coefficients of a and b, we get x = y1;

y = x1 - (a/b) * y1;

CODES

```
int gcdExtended(int a,
int b, int* x, int* y){
  if (a == 0){
    *x = 0, *y = 1;
    return b;
  }
  int x1, y1;
  int gcd =
gcdExtended(b % a, a,
&x1, &y1);
*x = y1 - (b / a) * x1;
*y = x1;
return gcd; }
int main() {
  int x,y,a=35,b=15;
  int g =
gcdExtended(a,b,&x,&y);
cout<<"GCD("<<a<<","<<b
<<")="<<g<<endl;
return 0; }
/Input: a = 35, b = 15
/Output:gcd=5, x=1,y=-2
Note: 35*1+15*(-2) = 5
```

Check Prime or not

```
bool checkPrime(ll n){
if(n==2) return true;
if(n==1)return false;
for
(int i=2;i*i<=n;i++){
 if(n%i==0)
   return false;}
return true;}
```

Sieve Algorithm

/To get primes in O(nloglogn)

```
const int N=1e7+10;/1e6

vector<ll>
primeFactor[N+1];

vector<bool>
isPrime(N+1,true);

vector<int>lp(N,0),hp(N,0
);

void sieve() {

isPrime[0] = false ;

isPrime[1] = false;

for (int p = 2; p * p <=
N; p++) {

  if (isPrime[p] == true)

{ lp[p] = hp[p] = p;

primeFactor[p].pb(p);

  for (int i = p * 2; i <
N; i += p) {

primeFactor[i].pb(p);
isPrime[i] = false;

hp[i] = p;

if(lp[i]==0) lp[i] = p;

      }} } }
```

Prime Factors

```
/Find hp using Sieve algo
map<ll,ll> primeFactor;
ll num;
cin>>num;
while(num>1){
  int primeFac = hp[num];
  while(num%primeFac==0){
     num /= primeFac;
primeFactor[primeFac]++;
  }
}
Input: num = 30
Output: 2 ase 3 bar,
        3 ase 1 bar
```

Find Divisors & Count till N

```
/Alternate of Sieve
```

```
const int Max = 1e5+10;
ll divcnt[Max];
vector<ll> divs[Max];
void DivisorCount(ll
n){
  for(int i = 1; i <=
n; i++){
    for(int j = i; j <=
n; j += i){
      divcnt[j]++;
      divs[j].pb(i);
    }}}
```

Count & SumOfDivisors of N

/Using Sieve&PrimeFactors

if num = $P_1^{n1}*P_2^{n2}*P_3^{n3}$..

count= $(n_1+1)(n_2+1)(n_3+1)\dots(n_k+1)$

sum $=(1+P_1^1+P_1^2+\dots+P_1^{n1})*(1+P_2^1+P_2^2+\dots+P_2^{n2})*(1+P_3^1+P_3^2+\dots+P_3^{n3})\dots$

$=[(P_1^{n1+1}-1)/(P_1-1)]*[(P_2^{n2+1}-1)/(P_2-1)]*[(P_3^{n3+1}-1)/(P_3-1)]\dots$

Input: 36

Output: 2^2*3^2

$=(1+2+4)*(1+3+9)$

$=[(2^{2+1}-1)/(2-1)]*[(3^{2+1}-1)/(3-1)]$

$=91$

SumOfDivisor of till N

```
/like N=2, sum = 4
/cz div[1]=1,div[2]=1,2
ll sumOfDiv(ll n){
  ll sum = 0;
  for(int i = 1; i <= n;
i++){
    ll div = n/i;
    sum += div*i;
    }
  return sum;
}
```

Power of Prime Number in a Factorial n!

```
ll largestPower(ll n, ll
p){
  ll x = 0;
  while(n){
    n /= p;
    x += n;
  }
return x; }
\Input: n = 100 and p1 = 5
\Output: e1 = [100/5] +
[100/25] + [100/125] …
 e1 = 20 + 4 + 0
 e1 = 24
The power of 5 in 100! is
24.
```

Binary Search

```
ll binarySearch(ll arr[],
ll l, ll r, ll x){
 while (l <= r) {
  ll m= l + (r - l)/2;
   if (arr[m] == x)
         return m;
   if (arr[m] < x)
        l = m + 1;
   else
    r = m - 1;
 }
return -1; }
```

BFS

```
bool bfs(src) {

deque<ll>q;

vis[src]= true;

q.pb(src);

d[src] = 0;

while(!q.empty()){

  ll src=q.front();

  q.pop_front();

each(graph[src]){

    adj=graph[src][i];

    if(vis[adj]==0){

      vis[adj]=true;

      q.pb(adj);
```

```
     prev[adj] =src;
      d[adj] = d[src]+1;
      }
}}}
return 0;}
```

DFS

```
bool vis[N];
ll prev[N];
ll stime[N],endTime[N];
vector<int>graph[10002];
prev[src]=-1,
stime[src] = 0;
time = 0;
void dfs(int node){
     time++;
    stime[node] = time;
    vis[node]=true;
 for(auto x:graph[node]){
   if(!vis[x]){
     prev[x] = node;
     dfs(x); }
  }
time++;
endTime[node]= time;}
```

Cycle Detection (DFS)

```
If(prev[u]!=v &&
visited[v] ==true) cycle
found. /inside loop
```

Path Print(BFS)

```
Print(G,src, cur){
If(cur==src) print(src)
Else if(Prev[cur]==-1)
    Print(no path);
Else{
   print(G,src,prev[cur]
   Print(cur); }
```

Dijkstra

```
struct Node{
  int at, cost;
  Node(int _at, int
_cost){
    at = _at;
    cost = _cost;
  }};
bool operator<(Node a,
Node b){
  return a.cost > b.cost;
}
struct Edge{
  int v, w;
  Edge(int _v, int _w){
    v = _v;
    w = _w;
  }
};
vector <Edge> G[10001];
priority_queue <Node> pq;
int dist[10001];
int n, m, s;
void dijsktra(int src){
  for(int i = 1; i <= n;
i++){
    dist[i] = 1e9;
  }
  dist[src] = 0;
  pq.push(Node(src, 0));
  while(!pq.empty()){
    Node u = pq.top();
    pq.pop();
  if(u.cost!=dist[u.at]){
      continue;
    }
   for(int i = 0; i <
G[u.at].size(); i++){
    Edge e = G[u.at][i];
       if(dist[e.v] >
u.cost + e.w){
      dist[e.v]=u.cost+e.w;
       pq.push(Node(e.v,
dist[e.v]));
       }
     }}
}
```

White-Black (two-color) - Balanced Subtrees(dfs)

```
void dfs(int s){
  for(int i=0;
i<nod[s].size(); i++){
    dfs(nod[s][i]);
b[s-1]+=b[nod[s][i]-1];
w[s-1]+=w[nod[s][i]-1];
  }
}
int main(){
  int t;
  cin>>t;
  while(t--){
    int n, q;
    cin>>n;
    w[0]=b[0]=0;
  for(int i=0; i<=n;
i++) nod[i].clear();
  for(int i=1; i<n;
i++){
      int x;
      cin>>x;
 nod[x].push_back(i+1);
      w[i]=b[i]=0;
    }
    cin>>st;
   for(int i=0; i<n;
i++){
   if(st[i]=='W'){
        w[i]=1;
        b[i]=0;
    }
   else{
        w[i]=0;
        b[i]=1;
      }
    }
    dfs(1);
    int cnt=0;
    for(int i=0; i<n;
i++){
      if(b[i]==w[i])
cnt++;
    }
   cout<<cnt<<endl;
  }
  return 0;}
```

BFS on GRID

```
#define ROW 4
#define COL 4
int dRow[] = {-1,0,1,0};
int dCol[] = {0,1,0,-1};
bool isValid(bool
vis[][COL],int row, int
col){
if(row<0 || col<0 ||
row>=ROW || col>=COL)
     return false;
if (vis[row][col])
          return false;
return true;
}
void BFS(int grid[][COL],
bool vis[][COL], int row,
int col){
queue<pair<int, int>>q;
q.push({ row, col });
vis[row][col] = true;
while (!q.empty()){
pair<int,int> cell =
q.front();
int x = cell.first;
int y = cell.second;
cout<<grid[x][y]<<" ";
q.pop();
for (int i = 0; i < 4;
i++) {
int adjx = x + dRow[i];
int adjy = y + dCol[i];
if (isValid(vis, adjx,
adjy)) {
q.push({ adjx, adjy });
vis[adjx][adjy] = true;
           }
        }
   }
}
int main(){
int grid[ROW][COL]={{
1, 2, 3, 4}, { 5, 6, 7, 8
}, { 9, 10, 11, 12 }, {
13, 14, 15, 16 }};
bool vis[ROW][COL];
memset(vis, false, sizeof
vis);
BFS(grid, vis, 0, 0);
return 0;}
```

Kruskal for finding MST

```
const int Max = 15e3+10;
struct Node{
  int u, v, w;
} g[Max];
bool less(Node a, Node
b){
  return a.w < b.w;
}
bool more(Node a, Node
b){
  return a.w > b.w;
}
int node, edge,
parent[Max];
int Find_parent(int n){
//cout<<": "<<n<<"
"<<parent[n]<<endl;
if(parent[n] == n)
     return n;
return
  Find_parent(parent[n]);
}
void graph(){
 for(int i = 1; ; i++){
   cin >> g[i].u >>
g[i].v >> g[i].w;
     edge++;
   }
}
int kruskal(){
  int sum = 0;
  for(int i = 0; i <=
node; i++)
     parent[i] = i;
  for(int i = 1; i <=
edge; i++)
   {
int u =
Find_parent(g[i].u), v =
Find_parent(g[i].v);
//cout <<u<<" "<< v << endl;
 if(u != v){//printf("The
cost from %d to %d is :
%d\n", s1[i], s2[i], w[i]);
     parent[u] = v;
     sum += g[i].w; }}
  return sum;}
int main(){
  int t;
  cin >> t;
  for(int tc = 1; tc <=
t; tc++){
    cin >> node;
    edge = 0;
    graph();
// Minimus
sort(g+1,g+edge+1,less);
cout<<kruskal()<<endl;
// Maximum
sort(g+1,g+edge+1,more);
cout<<kruskal()<<endl;}
 return 0;}
```

Prim's for finding MST

```
const int Max = 1e5+10;
bool vist[Max];
vector <pll> G[Max];
ll prim(int src){
 priority_queue<pll,
vector<pll>,greater<pll>>
q;
  ll mn = 0;
  q.push(make_pair(0,
src));
  while(!q.empty()){
    pll p = q.top();
    q.pop();
    int u = p.second;
   if(vist[u] == true){
      continue;
    }
    mn += p.first;
    vist[u] = true;
    for(pll v : G[u]){
      if(vist[v.second]
== false){
        q.push(v);
      }
    }
  }
  return mn;
}
int main(){
  int n, m, u, v;
  ll w, mn;
  cin >> n >> m;
  for(int i = 1; i <= m;
i++){
    cin >> u >> v >> w;
G[u].push_back(make_pair(
w, v));
G[v].push_back(make_pair(
w, u));}
  mn = prim(1);
  cout << mn << endl;
  return 0;
}
```

KMP string matching

```
void
computeLPSArray(string
pat, int M, int* lps){
     int len = 0;
     lps[0] = 0;
     int i = 1;
while (i < M) {
 if(pat[i]==pat[len]){
     len++;
     lps[i] = len;
     i++;
 }
 else{
   if (len != 0){
    len = lps[len - 1];
   }
   else{
     lps[i] = 0;
     i++;
    }}}
}
void KMPSearch(string
pat, string txt){
int M = pat.size();
int N = txt.size();
int lps[M];
computeLPSArray(pat, M,
lps);
int i = 0, j = 0;
while ((N - i) >= (M -
j)) {
 if (pat[j]==txt[i]){
     j++;
     i++;
  }
 if (j == M) {
  printf("Found pattern
at index %d ", i - j);
   j = lps[j - 1];
  }
 else if (i < N && pat[j]
!= txt[i]) {
 if (j != 0)
    j = lps[j - 1];
 else
    i = i + 1;
 }}
}
int main(){
string txt =
"ABABDABACDABABCABAB";
string pat = "ABABCABAB";
KMPSearch(pat, txt);
 return 0;
}
```

Longest increasing subsequence (LIS)

```
void solve(){
  ll n;
  cin>>n;
  ll arr[n];
for(int i=0;i<n;i++)
        cin>>arr[i];
vector<ll>v;
v.push_back(arr[0]);
for(int i=1;i<n;i++){
 if(arr[i]>v.back())
   v.push_back(arr[i]);
 else{
   int ind =
lower_bound(all(v),arr[i]
) - v.begin();
v[ind]=arr[i];
 }}
cout<<v.size()<<endl;}
```

Longest common subsequence (LCS)

```
int lcs(string X, string
Y, int m, int n){
   int L[m + 1][n + 1];
   int i, j;
for (i = 0; i <= m; i++){
   for (j = 0; j <= n;
j++) {
  if (i==0 || j==0)
     L[i][j] = 0;
  else if (X[i - 1] ==
Y[j - 1])
  L[i][j]=L[i - 1][j - 1]
+ 1;
  else
L[i][j] = max(L[i -
1][j], L[i][j - 1]);
      }
    }
return L[m][n];
}
int main() {
    string X = "AGGTAB";
    string Y = "GXTXAYB";
    int m = X.size();
    int n = Y.size();
printf("Length of LCS is
%d\n", lcs(X, Y, m, n));
  return 0;
}
```

Longest Unique Subarray

```
ll n;
cin >> n;
vector<int> v;
map<int, bool> mp;
int mx=0;
int currans=0;
int j=0;
for(int i=0; i<n; i++){
  int k;
  cin>>k;
  v.push_back(k);
if(mp.find(k)==mp.end()||
mp[k]==false) {
    mp[k]=true;
    currans++;
    mx=max(currans,mx);
  }
  else{
    while(v[j]!=k){
      mp[v[j]]=false;
      j++;
    }
    currans=(i-j);
    j++;
  }
}
cout<<mx<<endl;
```

Coin Change

In a strange shop there are n types of coins of value A1, A2,... An. You have to find the number of ways you can make K using the coins. You can use any coin at most K times.
For example, suppose there are three coins 1, 2, 5. Then if K = 5 the possible ways are:
    11111, 1112, 122, 5
So, 5 can be made in 4 ways.
Solution:

```
ll ar[Max];
int main(){
ll t, n, k;
cin>>t;
ll tc =1;
 while(t--){
   cin>>n>>k;
   ll dp[k + 1];
memset(dp,0,sizeof dp);
   dp[0] = 1;
   for(int i = 1; i <= n;
i++) cin>>ar[i];
  for(int i = 1; i <= n;
i++){
  for(int j = 1; j <= k;
j++){
   if(ar[i] <= j){
    dp[j] = dp[j] % Mod +
dp[j-ar[i]] % Mod;
    dp[j] %= Mod;
        }
      }
    }
printf("Case %d: %lld\n",
tc, dp[k]);
    tc++;
 }
 return 0;
}
```

least Coin needed

```
int count(vector<int>&
coins, int n, int sum){
  vector<vector<int>>
dp(n + 1, vector<int>(sum
+ 1, 0));
    dp[0][0] = 1;
for(int i=1;i <= n;i++){
 for(int j=0;j<=sum;j++){
  dp[i][j] += dp[i-1][j];
  if((j-coins[i-1])>= 0){
    dp[i][j] += dp[i][j -
coins[i - 1]];
              }
         }
    }
    return dp[n][sum];
}
```

Knapsack profit

```
// Function to find the
maximum profit
int knapSack(int W, int
wt[], int val[], int n)
{
    int dp[W + 1];
    memset(dp, 0,
sizeof(dp));
 for (int i = 1; i < n +
1; i++) {
  for (int w = W; w >= 0;
w--) {
   if (wt[i - 1] <= w)
     dp[w] = max(dp[w],
dp[w-wt[i-1]] +val[i-1]);
     }
  }
     return dp[W];
}

int main() {
  int profit[] = { 60,
100, 120 };
   int weight[] = { 10,
20, 30 };
      int W = 50;
      int n =
sizeof(profit) /
sizeof(profit[0]);
cout << knapSack(W,
weight, profit, n);
      return 0;
}
```

SOS DP

```
Problem: Given a fixed
array A of 2N integers,
we need to calculate ∀ x
function F(x) = Sum of
all A[i] such that x&i =
i, i.e., i is a subset of
x.

for(int i = 0; i < (1 <<
N); ++i)F[i] = A[i];
for(int i = 0; i < N;
++i){
  for(int mask = 0;
mask<(1 << N);++mask){
   if(mask & (1 << i)){
      F[mask] += F[mask ^
(1 << i)];
    }}}
```

Remove k digit build lowest number

```
string num;
int k;
cin>>num>>k;
stack<char>stk;
int sz=num.size();
for(int i=0;i<sz; i++){
  while(k>0 &&
!stk.empty() &&
stk.top()>num[i]){
    stk.pop();
    k--;
  }
 if(num[i]!='0')
   stk.push(num[i]);
 else if(!stk.empty())
   stk.push(num[i]);
}
while(!stk.empty() &&
k--){
  stk.pop();
}
vector<char> ans;
while(!stk.empty()){
ans.push_back(stk.top());
 stk.pop();
}
reverse(all(ans));
if(ans.empty())cout<<0;
for(auto i:ans)cout<<i;
cout<<endl;
```

Subarray Sum count number of subarrays have same sum

```
cin>>n>>x;
map<ll, ll>mp;
ll A[n+3], cnt=0, sum=0;
for(int i=0; i<n; i++){
  cin>>k;
  sum+=k;
  A[i]=sum;
}
for(int i=0; i<n; i++){
  if(A[i]==x) cnt++;
  if(mp[A[i]-x]>=1)
cnt+=mp[A[i]-x];
  mp[A[i]]++;
}
cout<<cnt<<endl;
```

Minimum Lexicographical Rotation

```
int
minimumExpression(strin
g s){
  s = s + s;
  int i = 0, j = 1, k =
0, len = s.size();
while(i + k < len && j
+ k < len){
  if(s[i + k]==s[j+k])
      k++;
else if(s[i+k]<s[j+k]){
  j=max(j+k+1, i+1);
   k = 0;
  }
else{
   i = max(i + k + 1, j
+ 1);
   k = 0;
    }
  }
  return min(i, j);
}
```

Merge Sort

```
void merge(int A[ ], int
start, int mid, int end){
  int p = start,q =
mid+1;
  int Arr[end-start+1],
k=0;
 for(int i = start ; i <=
end ; i++) {
  if(p > mid)
    Arr[ k++ ] = A[q++] ;
  else if ( q > end)
    Arr[k++]= A[p++];
  else if(A[p]<A[q])
    Arr[ k++ ] = A[p++];
 else
     Arr[k++]=A[q++];
 }
 for(int p=0; p< k ; p++)
    A[start++]=Arr[p];
}
void merge_sort (int A[
], int start, int end){
  if( start < end ){
    int mid = (start +
end ) / 2 ;
merge_sort (A,start,mid);
merge_sort(A,mid+1, end);
merge(A,start,mid,end);
  }
}
```

Counting Sort

```
void counting_sort(int
A[], int Aux[], int
sortedA[], int N){
  // First, find the
maximum value in A[]
  int K = 0;
  for(int i=0; i<N;
i++){
     K = max(K, A[i]);
  }
// Initialize the
elements of Aux[] with
0
 for(int i=0 ; i<=K;
i++){
     Aux[i] = 0;
  }
// Store the
frequencies of each
distinct element of
A[],
  // by mapping its
value as the index of
Aux[] array
  for(int i=0; i<N;
i++){
     Aux[A[i]]++;
  }
  int j = 0;
  for(int i=0; i<=K;
i++){
    int tmp = Aux[i];
    // Aux stores which
element occurs how many
times,
    // Add i in
sortedA[] according to
the number of times i
occured in A[]
    while(tmp--){
      //cout << Aux[i]
<< endl;
      sortedA[j] = i;
      j++;
    }
  }
}
```

Next Greater Element // O(n)

```
vector<int>
nextGreaterElement(vect
or<int> &arr){
  int n = arr.size();
  stack<int> s;
  vector<int> ret(n +
1, n);
  for(int i = n - 1; i
>= 0; i--) {
   while(!s.empty() &&
arr[s.top()] <=
arr[i]){
      s.pop();
    }
   if(!s.empty()){
      ret[i] = s.top();
    }
    s.push(i);
  }
  return ret;
}
```

String OP

Mod of a string:

```
int mod = 0;
mod = (mod*10 + s[i]-'0')
% num;
```

String Multiplication:

```
string multiply(string
num1, string num2){
int len1 = num1.size();
int len2 = num2.size();
if(len1==0 || len2==0)
   return "0";
vector<int> result(len1 +
len2, 0);

 int i_n1 = 0;
 int i_n2 = 0;

for(int
i=len1-1;i>=0;i--){
    int carry = 0;
    int n1 = num1[i] -
'0';

i_n2 = 0;
for (int j=len2-1; j>=0;
j--){
  int n2 = num2[j] - '0';
  int sum = n1*n2 +
result[i_n1 + i_n2] +
carry;
   carry = sum/10;
 result[i_n1 + i_n2] =
sum % 10;
    i_n2++;
 }
if (carry > 0)
      result[i_n1 + i_n2]
+= carry;
    i_n1++;
  }
int i =result.size()-1;
while (i>=0 && result[i]
== 0)
    i--;

  if (i == -1)
    return "0";

  string s = "";

  while (i >= 0)
    s +=
std::to_string(result[i--
]);

  return s;
}

int main(){
  string str1;
  string str2;

  if((str1.at(0) == '-'
|| str2.at(0) == '-') &&
      (str1.at(0) != '-'
|| str2.at(0) != '-' ))
    cout<<"-";


  if(str1.at(0) == '-')
    str1 =
str1.substr(1);

  if(str2.at(0) == '-')
    str2 =
str2.substr(1);

  cout << multiply(str1,
str2);
  return 0;
}
```

String Addition:

```
string findSum(string
str1, string str2){
  if (str1.length() >
str2.length())
    swap(str1, str2);
   string str = "";

int n1 = str1.length(),
n2 = str2.length();
reverse(str1.begin(),
str1.end());
reverse(str2.begin(),
str2.end());

 int carry = 0;
for (ll i=0; i<n1;
i++){
  int sum =
((str1[i]-'0')+(str2[i]
-'0')+carry);

str.push_back(sum%10 +
'0');
   carry = sum/10;
  }
for(ll i=n1;i<n2; i++){
   int sum =
((str2[i]-'0')+carry);

str.push_back(sum%10 +
'0');
    carry = sum/10;
  }
if (carry)
str.push_back(carry+'0'
);

  reverse(str.begin(),
str.end());
  return str;
}
```

Large Division

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
int main(){
    int test, cs = 1;
    cin>>test;
    while(test--){
        string s;
        ll div;
        cin>>s>>div;
        ll temp = 0;
        if(div < 0)
    div = -1*div;
 for(int i = 0; i <
s.size(); i++){
  if(s[i] == '-')
     continue;
  temp =
temp*10+(s[i]-'0');
 if(temp >= div){
    temp %= div;
        }
        }
   cout<<"Case
"<<cs++<<": ";
 if(temp == 0)
cout<<"divisible"<<endl
;
 else
   cout<<"not
divisible"<<endl;
     }
}
```

Catalan Number

```
ll catalon_number(ll
n){
    vll catalon(n+5);
    catalon[0]=1;
    FOR(1, n+1){
     catalon[i] =
(catalon[i-1]*(4*i-2))/
(i+1);
    }
   return catalon[n];
}
```

The Catalan numbers: 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, 208012, 742900, 2674440, 9694845

Distance:

1. 2 point :
2. point - line: (ax1+by1+c)/sqrt(a^2+b^2)
3. Line - line: (c1-c2)/sqrt(a^2+b^2)

Ordered Set:

```
#include<bits/stdc++.h>
#include
<ext/pb_ds/assoc_contai
ner.hpp>
#include
<ext/pb_ds/tree_policy.
hpp>
using namespace std;
using namespace
__gnu_pbds;
template <typename T>
using Set = tree<T,
null_type,

less<T>, rb_tree_tag,

tree_order_statistics_n
ode_update>;
Set <int> st;
int main(){
    st.insert(5);
//Insert
    st.erase(5);
//Delete
    st.insert(1);
    st.insert(2);
    st.insert(9);
    cout <<
*st.find_by_order(0) <<
endl; //Find value by
rank
    cout <<
st.order_of_key(9) <<
endl; //Find value's
rank
    /* For multiple
same element, use pair,
store index in second
of pair */
    return 0;
}
```

Hashing

```
const int base = 331;
const int Max = 2e6+10;
const int Mod = 1e9+ 7;
const ll Inf = 1LL<<62;
ll pw[Max];
ll Hash[Max];
void pre_power(){
   pw[0] = 1;
for(ll i = 1;i<Max;i++)
{
  pw[i] = (pw[i - 1] *
base) % Mod;
    }
}
void Hashing(string
str, int len){
   ll hash_val = 0;
   for(int i = 0; i <
len; i++){
     hash_val =
(hash_val * base +
str[i]) % Mod;
 Hash[i + 1] =hash_val;
    }
}
ll SubstringHash(int l,
int r){
    return (Hash[r] -
(Hash[l - 1] * pw[r - l
+ 1]) % Mod + Mod) %
Mod;
}
```

Mobius:

```
int N = 15;

int mu[N+1];
memset(mu,0,sizeof(mu))
;
mu[1] = 1;

for(ll i = 1;i<=N; ++i)
 for(ll j = 2*i; j<=N;
j += i)
    mu[j] -= mu[i];
for(int i = 1; i<=N;
++i)
    printf("%d :
%d\n",i,mu[i]);
```

Binary Indexed Tree

```
const int Max = 1e5 +
10;
int ar[Max], n;
ll BIT[Max];

void update(int idx,
int val){
  while(idx <= n){
     BIT[idx] += val;
    idx += idx & -idx;
    }
}

ll query(int idx){
     ll ret = 0;
     while(idx > 0){
     ret += BIT[idx];
     idx -= idx & -idx;
     }
     return ret;
}

ll query(int l, int r){
     return query(r) - 
query(l - 1);
}
void build(){
    for(int i = 1; i <= 
n; i++){
          update(i, 
ar[i]);
     }
}

int main(){
  int q, l, r;
  scanf("%d %d", &n, 
&q);
 for(int i = 1; i <= n; 
i++){
    scanf("%d", &ar[i]);
   }
  build();
  while(q--){
     cin>>l>>r;
printf("%lld\n", 
query(l, r));
    }
    return 0;
}
```

Segment tree

Sum/Min/Max/kth '1'/first ind>x

```
const ll N = 1e5+2;

ll tree[4*N], a[N];

void build(ll node, ll 
st, ll en){

 if(st==en){

   tree[node] = a[st];

     return;

  }

 ll mid = (st+en)/2;

build(2*node, st, mid);

build(2*node+1,mid+1,en
);

/For sum/kth '1'

tree[node] = 
tree[2*node] + 
tree[2*node+1];

/For max/first ind>x

tree[node] = 
max(tree[2*node],tree[2
*node+1]);

}


ll query(ll node, ll 
st, ll en, ll l, ll r){

/For kth '1' replace 
'l,r' with 'k'

 if(st>r || en<l)

    return 0; /INT_MIN

 if(l<=st && en<=r)

    return tree[node];

/For kth '1'

if(st==en) return st;

if(k<tree[2*node])

    return 
query(2*node,st,mid,k);

else

    return 
query(2*node+1,mid+1,en
,k-tree[2*node]);


ll mid = (st+en)/2;

ll q1 = query(2*node, 
st, mid, l, r);

ll q2 = query(2*node+1, 
mid+1, en, l, r);

/For sum

return q1 + q2;

/For max/first ind > x

return max(q1, q2);

}


void update(ll node, ll 
st, ll en, ll ind, ll 
val){

/For kth '1' remove 
'val'

  if(st==en) {

     a[st] = val;

    tree[node] = val;

/For kth '1'

  a[st] ^= 1;

  tree[node] ^= 1;

    return;  }

 ll mid = (st+en)/2;

/For kth '1' parameters 
of calling func might 
be changed.

 if(ind<=mid) 
update(2*node,st,mid,in
d,val);

 else
```

```
   update(2*node+1,
mid+1, en,ind,val);
/For sum or, kth '1'
tree[node]=tree[2*node]
+ tree[2*node+1];
/For max /first ind>x
tree[node]=max(tree[2*n
ode] , tree[2*node+1]);
}
ind main(){
 int n,q;
 cin >> n >> q;
 for(int i=0; i<n;i++){
    cin >> a[i];
 }
 build(1,0,n-1);
 while(q--){
     ll type;
     cin>>type;
if(type==1){
    ll ind, x;
   cin >> ind >> x;
 update(1,0,n-1,ind,x);
}
else {
   ll l, r;
  cin>>l>>r;
  ll ans =
query(1,0,n-1, l, r);
   cout<<ans<<endl;
/For first ind > x
 ll x;
cin>>x;
ll lo = 0, hi = n-1;
ll ans = n;
while(lo<=hi){
  ll mid = (lo+hi)/2;
  if(query(1,0,n-1,lo,
mid) < x)
  lo = mid + 1;
else{
hi = mid - 1;
ans = min(ans, mid);
  }
}
if(ans==n)
   cout<<-1<<endl;
else
   cout<<ans<<endl;
 }
 }
 return 0;
}
```

Min/Max and number of same element

```
const ll N = 1e5+2;
pll tree[4*N];
ll a[N];
void build(ll node, ll
st, ll en){
 if(st==en){
tree[node].first=a[st];
tree[node].second = 1;
       return;
  }
 ll mid = (st+en)/2;
build(2*node, st, mid);
build(2*node+1,mid+1,en
);
if(tree[2*node].ff ==
tree[2*node+1].ff){
   tree[node].ff =
tree[2*node].ff;
  tree[node].ss =
tree[2*node].ss +
tree[2*node+1].ss;
}
else if(tree[2*node].ff
< tree[2*node+1].ff){
     tree[node].ff =
tree[2*node].ff;
     tree[node].ss =
tree[2*node].ss;
}
else if(tree[2*node].ff
> tree[2*node+1].ff){
     tree[node].ff =
tree[2*node+1].ff;
     tree[node].ss =
tree[2*node+1].ss;
   }
}
pll query(ll node, ll
st, ll en, ll l, ll r){
  if(st>r || en<l)
      return {MAX,0};
 if(l<=st && en<=r)
   return tree[node];

ll mid = (st+en)/2;
pll q1 = query(2*node,
st, mid, l, r);

pll q2 =
query(2*node+1, mid+1,
en, l, r);


if(q1.ff == q2.ff){

q1.ss = q1.ss+q2.ss;

  return q1;

}

else if(q1.ff>q2.ff)

     return q2;

else

     return q1;

}

void update(ll node, ll
st, ll en, ll ind, ll
val){

 if(st==en) {

     a[st] = val;

  tree[node].ff = val;

  return;

 }

ll mid = (st+en)/2;


if(ind<=mid)

  update(2*node,st,
mid, ind,val);

else

  update(2*node+1,
mid+1, en,ind,val);


if(tree[2*node].ff ==
tree[2*node+1].ff){

   tree[node].ff =
tree[2*node].ff;

   tree[node].ss =
tree[2*node].ss +
tree[2*node+1].ss;

  }

else if(tree[2*node].ff
< tree[2*node+1].ff){

  tree[node].ff =
tree[2*node].ff;

   tree[node].ss =
tree[2*node].ss;

     }

 else
if(tree[2*node].ff >
tree[2*node+1].ff){

   tree[node].ff =
tree[2*node+1].ff;

   tree[node].ss =
tree[2*node+1].ss;

    }

}


int main(){
   ll n, q;
   cin>>n>>q;
for(ll i=0; i<n;i++){
   cin >> a[i];
      }
 build(1,0,n-1);
  while(q--){
    ll type;
    cin>>type;
if(type==1){
   ll ind,x;
  cin >> ind >> x;
update(1,0,n-1,ind,x);
    }
else {
    ll l, r;
   cin>>l>>r;

  pll ans =
query(1,0,n-1, l, r-1);
 cout<<ans.ff<<"
"<<ans.ss<<endl;  }}}
```

Max Sum of Segment

```
—-----------------------------------------------
sum = Total sum of
segment
pref = max sum of pref
segment
suff = max sum of suff
segment
ans= max sum of the
segment
————————————————————————
const ll N = 1e5+2;
struct grp{
 ll sum,pref,suff,ans;
};
grp tree[4*N];

ll a[N];

void build(ll node, ll
st, ll en){

 if(st == en){

    if(a[st]<=0){

tree[node].sum = a[st];
tree[node].pref =
tree[node].suff =
tree[node].ans = 0;   }

 else{
tree[node].sum =
tree[node].pref =
tree[node].suff =
tree[node].ans = a[st];

   }

   return;  }

ll mid = (st + en)/2;

build(2*node, st, mid);

build(2*node+1, mid+1,
en);
tree[node].sum =
tree[2*node].sum +
tree[2*node+1].sum;

tree[node].pref =
max(tree[2*node].pref,
tree[2*node].sum +
tree[2*node+1].pref);

tree[node].suff =
max(tree[2*node+1].suff
, tree[2*node+1].sum +
tree[2*node].suff);

tree[node].ans =
max(tree[2*node].suff+t
ree[2*node+1].pref,
max(tree[2*node].ans,
tree[2*node+1].ans));

}

void update(ll node, ll
st, ll en, ll idx, ll
val){

  if(st == en){

     a[st] = val;

    if(a[st]<=0){
tree[node].sum = a[st];
tree[node].pref =
tree[node].suff =
tree[node].ans = 0;

        }

  else{
tree[node].sum =
tree[node].pref =
tree[node].suff =
tree[node].ans = a[st];

        }

     return; }

  ll mid = (st+en)/2;

 if(idx <= mid){

    update(2*node, st,
mid, idx, val);
 }

 else{

update(2*node+1, mid+1,
en, idx, val);

   }

  tree[node].sum =
tree[2*node].sum +
tree[2*node+1].sum;

  tree[node].pref =
max(tree[2*node].pref,
tree[2*node].sum +
tree[2*node+1].pref);

   tree[node].suff =
max(tree[2*node+1].suff
, tree[2*node+1].sum +
tree[2*node].suff);

   tree[node].ans =
max(tree[2*node].suff+t
ree[2*node+1].pref,
max(tree[2*node].ans,
tree[2*node+1].ans));

}

int main(){
    ll n, q;
   cin>>n>>q;
for(ll i=0; i<n;i++){
    cin >> a[i];
   }
build(1,0,n-1);
cout<<tree[1].ans<<endl
;
while(q--){
     ll ind,x;
   cin >> ind >> x;
update(1,0,n-1,ind,x);
cout<<tree[1].ans<<endl
;
   }
}
```

Segment Tree - Lazy Propagation

```
const int mx = 1e5 +
10;
ll a[mx];
struct Node
{
  ll sm, prop;
} seg[4*mx];

void build(ll nod, ll
lo, ll hi)
{
  if(lo==hi)
  {
  seg[nod].sm=a[lo];
    return;
  }
  ll mid = (lo+hi)>>1;

 build(nod*2, lo, mid);
  build(nod*2+1, mid+1,
hi);
  seg[nod].sm =
seg[nod*2].sm +
seg[nod*2+1].sm;
  seg[nod].prop = 0;
}
ll query(ll nod, ll lo,
ll hi, ll l, ll r, ll
cary)
{
  if(lo>r || hi<l)
return 0;
  if(lo>=l && hi<=r)
 {
    return seg[nod].sm
+ cary*(r-l+1);
  }
  ll mid=(lo+hi)>>1;
  ll x = query(nod*2,
lo, mid, l, r, cary +
seg[nod].prop);
  ll y = query(nod*2+1,
mid+1, hi, l, r, cary +
seg[nod].prop);

  return x+y;
}

void update(ll nod, ll
lo, ll hi, ll l, ll r,
ll val)
{
  if(lo>r || hi<l)
return;
  if(lo>=l && hi<=r)
  {
    seg[nod].sm +=
((r-l+1) * val);
    seg[nod].prop +=
val;
    return;
  }
  ll mid=(lo+hi)>>1;
  update(nod*2, lo,
mid, l, r, val);
  update(nod*2+1,
mid+1, hi, l, r, val);
  seg[nod].sm =
seg[nod*2].sm +
seg[nod*2+1].sm +
(r-l+1) *
seg[nod].prop;
}

int main(){
  ll n, q;
  cin>>n>>q;
for(ll i=0; i<n; i++)
     cin>>a[i];

  build(1, 0, n-1);

for(ll i=0; i<q; i++){
    int x;
    cin>>x;
    if(x==2){
      ll l, r;
      cin>>l>>r;
      cout<<query(1, 0,
n-1, l, r, 0)<<endl;
    }
    else{
      ll val, pos;
      cin>>pos>>val;
      update(1, 0, n-1,
pos-1, pos-1, val);
    }
  }
}
```

Segmented tree Tri

```
#define FOR(i,a,b)
for(int i=a;i<=b;i++)

#define ROF(i,a,b)
for(int i=a;i>=b;i--)

#define REP(i,b)
for(int i=0;i<b;i++)


int tri[1000005][26];
//Total char in input
file,Number of distinct
char

bool flag[1000005];
//Indicate where string
finishes

int id=1;


int main(){
    string str;
    cin >> str;
    int r=1;
   REP(i,str.size()){
  int x=str[i]-'a'; //
It maybe '0'/'A'/both
  if(!tri[r][x]){
    tri[r][x]=++id;
   }
    r=tri[r][x];
  }
  flag[r]=true;
  return 0;
}
```

Matrix expo

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

using namespace std;

#define ull unsigned
long long

#define ll long long

ll a,b,n,x;


void matmul(ll a[2][2],
ll b[2][2]){

    ll mul[2][2];

    for(int i=0; i<2;
i++){

        for(int j=0;
j<2; j++){


mul[i][j]=0;

            for(int
k=0; k<2; k++){


mul[i][j]=(mul[i][j]+(a
[i][k]*b[k][j])%x)%x;

            }

        }

    }

for(int i=0; i<2; i++){

for(int j=0; j<2; j++){

   b[i][j]=mul[i][j];

  //cout<<a[i][j]<<" ";

    }

   //cout<<endl;

  }

 return;

}

void find_ans(ll
m[2][2],ll b[2][2], ll
p){

  while(p){
    if(p&1)
       matmul(m,b);
       matmul(m,m);
       p/=2;
   }
  return;
}
void solve(){
   cin>>a>>b>>n>>x;
   x=pow(10,x);
  ll base[2][2]=
{{b,0},{a,0}};
  ll mat[2][2]=
{{1,1},{1,0}};
  ll ans;
  if(n==1) ans=b;
  else if(n==0) ans=a;
   else{
find_ans(mat,base,n-1);
ans=base[0][0]+base[0][
1];
   }
  cout<<ans<<endl;
    return;
}

int main(){
    int t, tc=1;
    cin>>t;
    while(t--){
        cout<<"Case
"<<tc++<<": ";
        solve();
    }
    //solve();
    return 0;
}
```

Mo Algorithm

problem link-
https://www.hackerrank.com/contests/gub-idpc-2022/challenges/frequency-xor/copy-from/1347288030

```
#define ull unsigned
long long
#define ll long long
#define pii
pair<int,int>
#define MAX 1000005
int arr[MAX],cnt[MAX],
ans[MAX];
int
n,q,block_size,ansr=0;
pair<pii,int>qry[MAX];
bool
cmp(pair<pii,int>x,pair
<pii,int>y){
    int
xx=x.first.first/block_
size;
    int
yy=y.first.first/block_
size;
    if(xx!=yy) return
xx<yy;
    return
x.first.second<y.first.
second;
}

void add(int x){
    if(cnt[x]==0){
        ansr^=1;
        cnt[x]++;
    }
    else{
        ansr^=cnt[x];
        cnt[x]++;
        ansr^=cnt[x];
    }
}
void Remove(int x){
    ansr^=cnt[x];
    cnt[x]--;
    ansr^=cnt[x];
}

void solve2(){
    int l=0, r=-1;
    set<int>s;
    for(int i=0; i<q;
i++){
        int
lp=qry[i].first.first;
        int
rp=qry[i].first.second;

        while(r<rp){
            r++;

add(arr[r]);
        }
        while(r>rp){

Remove(arr[r]);
            r--;
        }
        while(l<lp){

Remove(arr[l]);
```

```
            l++;

        }

        while(l>lp){

            l--;

add(arr[l]);

        }

ans[qry[i].second]=ansr
;

    }


    for(int i=0; i<q;
i++)

cout<<ans[i]<<endl;

}


void solve(){
    cin>>n;
    for(int i=1; i<=n;
i++) cin>>arr[i];
    cin>>q;
    for(int i=0; i<q;
i++){
        int x,y;
cin>>x>>y;

qry[i].first.first=x;
qry[i].first.second=y;
qry[i].second=i;
    }
 block_size=sqrt(n);
sort(qry,qry+q,cmp);
    solve2();
    return;
}
```

Pollard rho

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

#define ull unsigned
long long

#define ll long long

#define SZ 1000005


int MARK[SZ+1];

vector<int>PRIME;


void sieve(){

    MARK[1]=1;

    int root=sqrt(SZ);

    for(int i=3;
i<=root; i+=2){

        if(!MARK[i]){

            for(int
j=i*i; j<=SZ;
j+=(i*2)){

MARK[j]=1;

            }

        }

    }

PRIME.push_back(2);

    for(int i=3; i<=SZ;
i+=2){

        if(!MARK[i])
PRIME.push_back(i);

    }

    return;

}

//ll Mul(ll a, ll p, ll
m){//If we use
recursive function for
this part then CPU
limit will be exit

//    if(p==0) return
0;

//    ll
ret=Mul(a,p/2,m);

//
ret=((ret%m)+(ret%m))%m
;

//    if(p&1)
ret=((ret%m)+(a%m))%m;

//

//    return ret;

//}

ll Mul(ll a, ll b, ll
m){

    ll ret=0, c=a;

    while(b){

        if(b&1)
ret=(ret+c)%m;

        b>>=1;
c=(c+c)%m;

    }

    return ret;

}

//ll bigmod(ll a, ll p,
ll m){
//    if(p==0) return
1;
//    ll
ret=bigmod(a,p/2,m);
//    ret*=ret;
//    if(p&1) ret*=a;
//
//    return ret;
//}

ll bigmod(ll a, ll n,
ll m){
 ll ret=1,c=a;
 while(n){
 if(n&1)
   ret=Mul(ret,c,m);
      n>>=1;
    c=Mul(c,c,m);
  }
  return ret;
}


bool isprime(ll n){

   if(n==2) return 1;

   if(n%2==0) return 0;

   ll d=n-1;

  while(d%2==0) d>>=1;

  int test[]=
{2,3,5,7,11,13,17,19,23
};

for(int i=0; i<9; i++){

 ll x=test[i]%(n-2),
temp=d;

 if(x<2) x+=2;

 ll a=bigmod(x,d,n);

 while(temp!=n-1 &&
a!=1 && a!=n-1){

   a=Mul(a,a,n);

   temp<<=1;

 }

 if(a!=n-1 &&
(temp&1)==0) return 0;

  }

  return 1;

}


ll pollard_rho(ll n, ll
c){
  ll x=2, y=2, i=1,
k=2, d;
    while(true){
x=(Mul(x,x,n)+c);
      if(x>=n) x-=n;
d=__gcd(abs(x-y),n);
     if(d>1) return d;
        if(++i==k){
            y=x, k<<=1;
        }
    }
    return n;
}


void llfactorize(ll n,
vector<ll> &f){
    if(n==1) return;
    if(n < 1e9){
    for(int i=0;
PRIME[i]*PRIME[i] <= n;
i++){
if(n%PRIME[i]==0){
while(n%PRIME[i]==0){
f.push_back(PRIME[i]);
   n/=PRIME[i];
   }
   }
  }
if(n!=1)f.push_back(n);
    return;
 }
 if(isprime(n)){
    f.push_back(n);
    return;
  }
    ll d=n;
for(ll i=2; d==n;i++){
d=pollard_rho(n,i);
   }
  llfactorize(d,f);
  llfactorize(n/d,f);
}
void factorize(ll n,
vector<pair<ll,ll>>
&ans){
   vector<ll>v;
  llfactorize(n,v);
if(v.size()==0) return;
sort(v.begin(),
v.end());
    ll a=v[0], b=1;
for(ll i=1; i<v.size();
i++){
if(v[i]==v[i-1]) b++;
else{
ans.push_back({a,b});
       a=v[i];
       b=1;
     }
   }
ans.push_back({a,b});
}
ll phi(ll n,
vector<pair<ll,ll>>
&ans){
    ll ph=n;
 for(auto i:ans){
    ph/=i.first;
   ph*=(i.first-1);
    }
  return ph;
}

void solve(){
    ll n,ans;
    cin>>n;
vector<pair<ll,ll>>v;
  factorize(n,v);
  ll phi_n=phi(n,v);
  ll b=n+1;
while(1){
vector<pair<ll,ll>>vv;
factorize(b,vv);
ll phi_b= phi(b,vv);
if(phi_b>phi_n){
  cout<<b<<endl;
  break;
  }
  b++;
  }
}
```