

# Competitive programming

- Elydark

→ Prime sieve

Sieve of erathostenes

Make an array (bool) of size n.  
Mark all true.

T	T	T	T	T	T	T	T	T	T	T	T
0	1	2	3	4	5	6	7	8	9	10	

$A[0] = \text{false}$ ;  $A[1] = \text{false}$

We need to find all primes from 2 to n.

```

for ( $i = 2; i * i \leq n; i++$ )
{
    if ( $A[i] == \text{true}$ )
        for ( $j = i * i; j \leq n; j += i$ )
        {
             $A[j] = \text{false}$ ;
        }
}

```

print those index which are marked true.

Time-complexity =  $n \log \log n$

Finding GCD

Page No.

Euclid's algorithm

$$g(d(a, b)) = \begin{cases} g(d(b, a \mod b)) \\ \text{when } a > b \end{cases}$$

Learn it as "bat"

$$g(d(a, 0)) = a$$

→ Extended - euclid algorithm

$$\gcd(a, b) = ax + by$$

we need to find  $x$  &  $y$ .

$$ax + by = \gcd(a, b) \quad i)$$

$$\text{Also, } \gcd(a, b) = \gcd(b, a \mod b) \quad ii)$$

From i) & ii)

$$ax + by = \gcd(b, a \mod b) \quad iii)$$

$$\Rightarrow bx_1 + (a \mod b)y_1 = \gcd(a, b) = ax + by$$

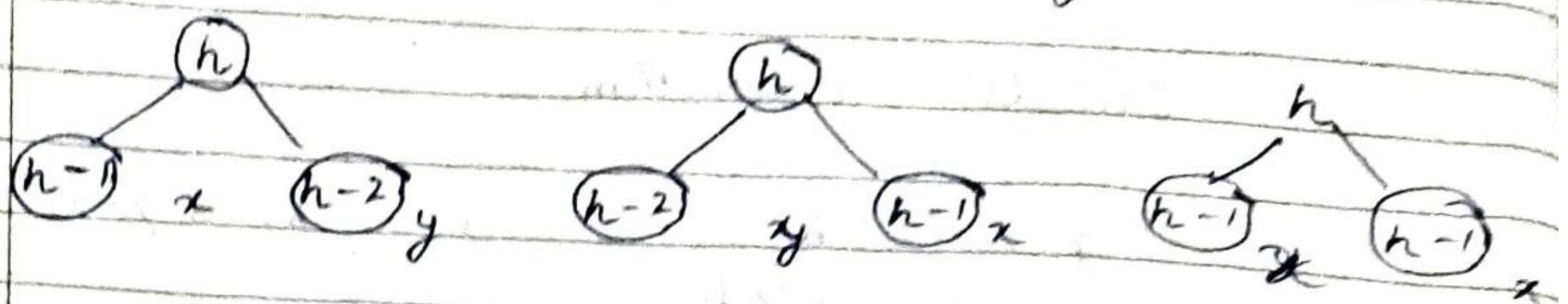
$$\Rightarrow bx_1 + \left\{ a - \left[ \frac{a}{b} \right] \cdot b \right\} y_1 = ax + by$$

⇒ On comparing, we get

$$x = y_1$$

$$y = x_1 - \left[ \frac{a}{b} \right] y_1$$

→ Number of balanced binary tree



$$\text{and} = 2x + y$$

```
int balance BT (int h) {
```

```
if (h == 0 || h == 1) {
```

```
return 1;
```

```
}
```

```
int m = pow(10, q) + 7;
```

```
int x = balance BT (h-1);
```

```
int y = balance BT (h-2);
```

```
long res1 = (long) x * x;
```

```
long res2 = (long) x * y * 2;
```

```
int ans1 = (int) (res1 / m);
```

```
int ans2 = (int) (res2 / m);
```

```
return (ans1 + ans2) / m;
```

```
}
```

$$\text{Balance} = |LH - RH| \leq 1$$

$(h-1) \rightarrow x$

$(h-2) \rightarrow y$

$$\text{and} = x^* x + x^* y + x^* y$$

$$= x^2 + 2xy$$

modulo

inverse

&

$$a \times b = 1$$

$$b = \frac{1}{a}$$

(multiplication

inverse)

Multiplication mod inverse

$$(A \cdot B) \cdot \text{lcm} = 1$$

We have to find B.

$$(A \cdot B) \cdot \text{lcm} = 1$$

$$\Rightarrow ((A \cdot \text{lcm}) \cdot (B \cdot \text{lcm})) \cdot \text{lcm} = 1$$

$$\Rightarrow \therefore (a \cdot b) \cdot \text{lcm} = ((a \cdot \text{lcm}) \cdot (b \cdot \text{lcm})) \cdot \text{lcm}$$

$$\Rightarrow 1 \leq B \leq m-1$$

$$(a \cdot b) \cdot \text{lcm} = 1$$

$$\Rightarrow (a \cdot b) \cdot \text{lcm} - 1 = 0$$

$$a \cdot b \equiv 1$$

$$\Rightarrow a \cdot b - 1 \equiv 0 \text{ (multiple of m)}$$

$$\Rightarrow a \cdot b - 1 \equiv mq$$

$$\Rightarrow a \cdot b + mQ = 1$$

$$\Rightarrow a \cdot b + mQ = 1$$

|

$$\Rightarrow g(d(a, m) = 1)$$

$$(ax + by) = \gcd(a, b)$$

mod inverse (a, m)

$$= b$$

matlab a me kya multiply kre  
aur m se mod kore ki ans = 1.

mod inverse (a, m) = b

$$\Rightarrow \underset{i}{(ax + ?) \% m} = t$$

we have to find this only

→ Hard question

$$ax + by = d$$

a & b item se d banana hai  
Need to find number of pairs  
possible of (x, y) for such

$$ax + by = d$$

$$ax = \underbrace{d - by}$$

This term must be  
divisible by x.

$$0 \leq y \leq \frac{d}{b}$$

$$0 \leq x \leq \frac{d}{a}$$

$$ax = d - by$$

Let us assume  $y_1$  to be minimum value for such condition to be hold.

$d \rightarrow$  fixed

$b \rightarrow$  fixed

$y \rightarrow$  can change.

$$ax = d - by_1$$

$$\text{Next term, } d - b(y_1 + a)$$

$$d - b(y_1 + 2a)$$

$$d - b(y_1 + 3a)$$

⋮

$$d - b(y_1 + na)$$

On comparing with maximum value.  
of  $y$

$$y_1 + na = \frac{d}{b}$$

$$na = \frac{d - y_1}{b}$$

$$n = \frac{d - y_1}{a}$$

Total terms  
 $= 1 + n$

Now we need to find  $y_1$

$$(d - by_1) \mod a = 0$$

$$\Rightarrow d/a - (by_1)/a = 0 \quad \text{--- i)}$$

$$\Rightarrow d/a - [b/a \cdot (y_1/a)] \mod a = 0 \quad \text{--- ii)}$$

$$ax + by = d$$

$$y = \frac{d - ax}{b}$$

For smallest value,

$$y_1/a = \left( \frac{d - ax}{b} \right) \mod a$$

$$= \left( \frac{d}{b} \right) \mod a - \frac{(ax)}{b} \mod a$$

$$y_1/a = \left( \frac{d}{b} \right) \mod a \quad \text{--- iii)}$$

i) & ii)

$$\Rightarrow d/a - [b/a \cdot \left( \frac{d}{b} \right) \mod a] \mod a = 0$$

$$\Rightarrow d/a - \left[ \left( b \cdot \frac{d}{b} \right) \mod a \right] \mod a = 0$$

$$d/a - (d/a) \mod a = 0$$

This holds

$$y_1 = \left( \frac{d}{b} \right) \mod a$$

$$= d^* \text{ mod inverse}(b, a)$$

if ( $d \neq g$ )  $\{$

$\} \quad \text{cout} << d << \text{endl};$

if ( $d == 0$ )  $\{$

$\} \quad \text{cout} << 1 << \text{endl};$

$a' = g;$

$b' = g;$

$d' = g;$

long value =  $((d * a)^* \text{modInverse}(b, a)) / a;$

long firstValue =  $d / b$

if ( $b < y1 * b$ )  $\{$

$\} \quad \text{cout} << 0 << \text{endl};$

else  $\{$

long n =  $(\text{firstValue} - y1) / a;$

$\} \quad \text{cout} << 1 + n << \text{endl};$

$\}$

$$y_1 = \left(\frac{d}{b}\right) \cdot a$$

$$ax + by = d$$

$$\cancel{ax} + \cancel{by} = \cancel{d}$$

$y$

$y$

$y$

$$Ax + By = D$$

→

## Advanced GCD

$$\text{gcd}(a, b) = \text{gcd}(b, a \% b)$$

both must be integers

$$\text{gcd}(10^{240}, 40) = \text{gcd}(40, 10^{240} \% 40)$$

Input as string.

$$\begin{aligned} \text{ex- } & (23567) \% 40 \\ &= (0 * 10 + 2) \% 40 = 2 \end{aligned}$$

$$(2 * 10 + 3) \% 40 = 23$$

$$(23 * 10 + 5) \% 40 =$$

$$2356 \% 40 = (235 * 10 + 6) \% 40$$

$$\begin{aligned} &= ((235 * 10) \% 6 + 6 \% 40) \% 40 \\ &= (\text{from previous} + 6 \% 40) \% 40 \end{aligned}$$

```

ll gcdLarge (ll a, string b) {
    ll mod = 0;
    for (int i=0; i < b.size(); i++) {
        mod = (mod * 10 + b[i] - '0') \% a;
    }
    return mod;
}

```

ICPC question  
train or walk

[Train]  
[Walk]

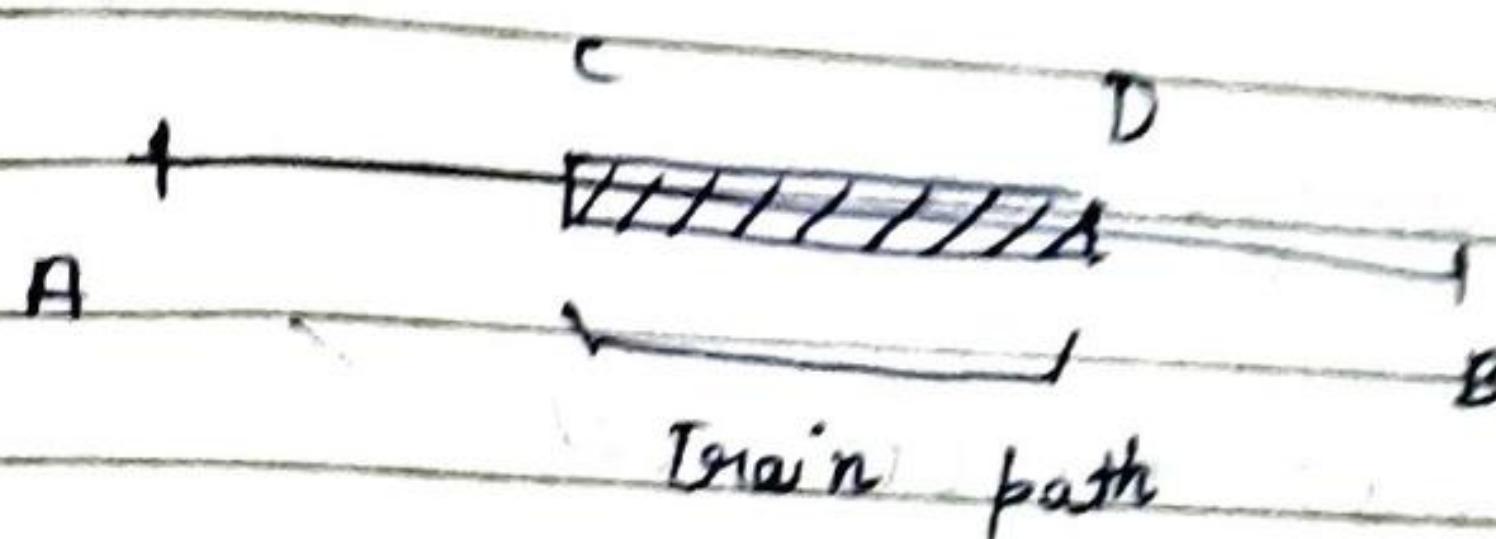


$$v_{\text{walk}} = \left(\frac{1}{b}\right) \text{ m/s}$$

$$v_{\text{train}} = \left(\frac{1}{q}\right) \text{ m/s}$$

train starts  
at  $t = 2$  sec

(From C to D)



Path 1: A to B without train

$$\text{distance} = |x_A - x_B|$$

Path 2: A to D without train

Path 3: A to C to D

Path 4: A to C to B

$$t = \frac{|x_B - x_A|}{\left(\frac{1}{b}\right)} = b(x_B - x_A)$$

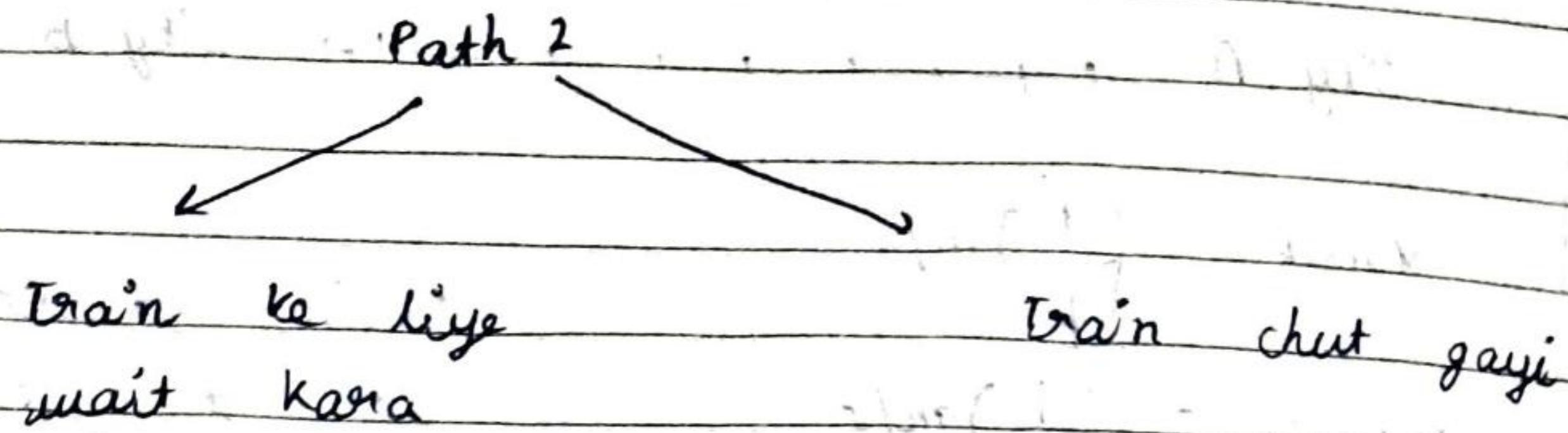
Path 2: A to C and go to D then B.

$$\text{distance} = |x_C - x_A| + |x_D - x_C| + |x_B - x_D|$$

$$t = b|x_C - x_A| + q|x_D - x_C| + b|x_B - x_D|$$

$$ans = \min(t_1, t_2);$$

But train starts at  $t = y$ .



waiting-time

$$= y - p |x_c - x_a|$$

if  $AC(\text{time}) > y$ :  
can't catch train

$$t_2 += \text{waiting\_train}$$

Path 1

$$ans = \min(t_1, t_2)$$

in >> n >> A >> B >> C >> D >> P >> Q >> Y;

int A[n];

for (int i=0; i<n; i++) in >> A[i];

int t1 = p \* (abs(A[a] - A[b]));

int tAC = p \* (abs(A[c] - A[a]));

if (tAC > y {

// do nothing; t2 = INT\_MAX;

}

else {

waiting-time = y - tAC;

t2 += waiting-time;

}

cout << min(t1, t2) << endl;

### Modified GCD - Codeforces

$$a = 24, b = 36$$

$$\text{gcd} = 12$$

1 2 3 4 6 → all divides gcd

Range - low & high which divides both a and b.

Making list of divisors

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

```
    if (gcd % i == 0) {
```

```
        v.push(i);
```

```
        if (i * i != gcd) {
```

```
            v.push(gcd / i);
```

```
}
```

```
}
```

```
res = -1;
```

```
for (d: div) {
```

```
    if (d >= L && d <= U) {
```

```
        res = max(res, d);
```

```
}
```

```
}
```

```
cout << res;
```

Zero remainder array - (odderes)

ex-  $k=5$

$4 \ 4 \ 4 \rightarrow$  same group  
(same remainder)

map <int, int> m;

$m[\text{remainder}] = \text{no. of same}$ ;

$4 \ 4 \ 4 \text{ for } k=5$

$x_1 = 1, x_2 = 6, x_3 = 11$

5                        5

$$x_1 \% 5 = x_2 \% 5 = x_3 \% 5$$

claim - Sum of 2 numbers is divisible by  $k$ , iff sum of their remainder is divisible by  $k$ .

$$g_1 = 4 \% 5 = 4 \quad \} \quad s = 5 \% 5 = 0$$

$$g_2 = 1 \% 5 = 1 \quad \}$$

$$g_1 = 4 \% 5 = 4 \quad \} \quad s = 5 \% 5 = 0$$

$$g_2 = 6 \% 5 = 1 \quad \}$$

1

3

ex-

$k=6$

8 7 1 8 3 7 5 10 8 9

$m[1] = 3, x_1 = 5, 11, 17$

$an = 17 + 1$

$n[2] = 3, x_2 = 4, 10, 16$

$m[3] = 2, x_3 = 3, 9$

$m[4] = 1, x_4 = 2$

$m[5] = 1, x_5 = 1$

## Chinese Remainder Theorem

2 pack of chocolates

1 pack = 5 people (3 rem)

1 pack = 6 people (2 rem)

$$x \equiv 3 \pmod{5}$$

$$x \equiv 2 \pmod{6}$$

$$x = n_5 + 3 \Rightarrow x = 3, \boxed{8}, 13, 18, 23$$

$$x = m_6 + 2 \Rightarrow x = 2, \boxed{8}, 14, 20, 26$$

$$x = 8, 38, 68$$

$$8, \quad 38, \quad 68$$

$$30 \quad 30$$

$$\downarrow ?$$

$$\text{LCM}(5, 6)$$

CRT,

$$a \equiv a_1 \pmod{p_1}$$

$$a \equiv a_2 \pmod{p_2}$$

⋮

$$a \equiv a_k \pmod{p_k}$$

$$x = \sum (a_{0m}[i] * pp[i] + inv[i]) - prod$$

$$pp[i] = \frac{prod}{num[i]}$$

$$inv[i] = \text{modInv}(pp[i], num[i]);$$

Code

```
for (i=0; i < k; i++) {
    int pp = prod / num[i];
    result += aem[i] * inv(pp, num[i]);
    pp =;
}
return result / prod;
```

$$TC = n \log n$$

Train partner - codechef

1	4	9	12
2	5	10	13
3	6	11	14
7	8	15	16

Remainder (% 8)

0

Solution

$n-1$

coach

SL

7

$n+1$

SU

1

$n+3$

LB

4

$n-3$

LB

2

$n+3$

MB

5

$n-3$

MB

3

$n+3$

UB

6

$n-3$

UB

## Matrix exponentiation

$$A = \begin{bmatrix} & \\ & \end{bmatrix}$$

$$B = \begin{bmatrix} & \\ & \end{bmatrix}$$

$$A \cdot B = \sum_{n=1}^k A_{i,n} \times B_{n,j}$$

$$\text{We know, } f(n) = f(n-1) + f(n-2)$$

$$m \times \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}$$

We need to find this  $m$

$$m = 2 \times 2$$

$$\Rightarrow \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}$$

$$a^* f(n) + b^* f(n-1) = f(n+1)$$

$$\text{Clearly, } a=1, b=1$$

$$c^* f(n) + d^* f(n-1) = f(n)$$

$$\text{Clearly, } c=0, d=1$$

$$m = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\text{So, } m^* \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix} = \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix}$$

$$m^* \begin{bmatrix} f(n+1) \\ f(n) \end{bmatrix} = \begin{bmatrix} f(n+2) \\ f(n+1) \end{bmatrix}$$

$$m \times m \times \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix} = \begin{bmatrix} f(n+2) \\ f(n+1) \end{bmatrix}$$

$$m^k \times \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix} = \begin{bmatrix} f(n+k) \\ f(n+k-1) \end{bmatrix}$$

Put  $n = 1$

&  $k = n - 1$

$$m^{n-1} \times \begin{bmatrix} f(1) \\ f(0) \end{bmatrix} = \begin{bmatrix} f(1+n-1) \\ f(1+n-1-1) \end{bmatrix}$$

$$m^{n-1} \times \begin{bmatrix} f(1) \\ f(0) \end{bmatrix} = \begin{bmatrix} f(n) \\ f(n-1) \end{bmatrix}$$

↓  
This is matrix exponentiation

$$M = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$m^{n-1} \rightarrow \log n$$

$$A^n = \left\{ \begin{array}{ll} A^{n/2} \cdot A^{n/2} & , n = \text{even} \\ [A^{n/2} \cdot A^{n/2}] \cdot A & , n = \text{odd} \end{array} \right\}$$

Ebo sum

$n & m$

$$S_i^{\circ} = S_{i-1}^{\circ} + f(n)$$

$$f(n) = f(n-1) + f(n-2)$$

$$f(n-1) = f(n-2) + f(n-3)$$

:

$$f(1) = 1$$

(+)

$$f(n) = f(n-2) + f(n-3) + f(n-4) \\ \underline{+} \quad \underline{\underline{f(1)+1}}$$

$\Rightarrow$

$$f(n) = S(n-1) + 1$$

$$S(n-1) = f(n) - 1$$

$$S(n) = f(n+2) - 1$$

According to question,

$$S(m) - S(n-1)$$

$$= f(m+2) - 1 - f(n+1) + 1$$

$$f(m+2) - f(n+1)$$

$$\text{ans} = [f(m+2) - f(n+1)] / m$$

## Game theory

(2 player game)

combinatorial game theory - Finite game

Imperial game

(No-limitation)



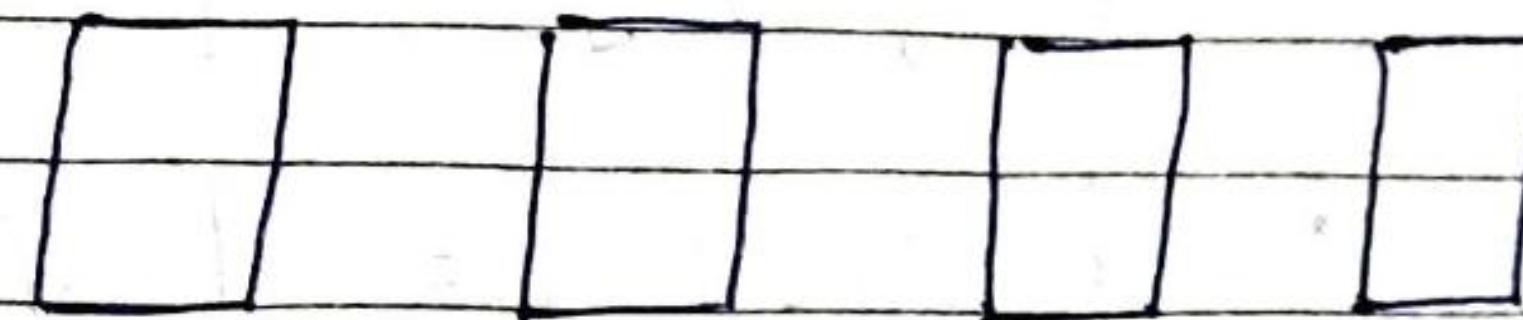
Partisan game

(Limitation)

or chess

Game theory deals with this part mostly.

→ Game of Nim



3

4

5

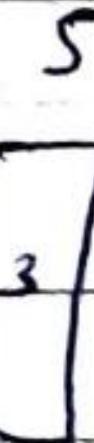
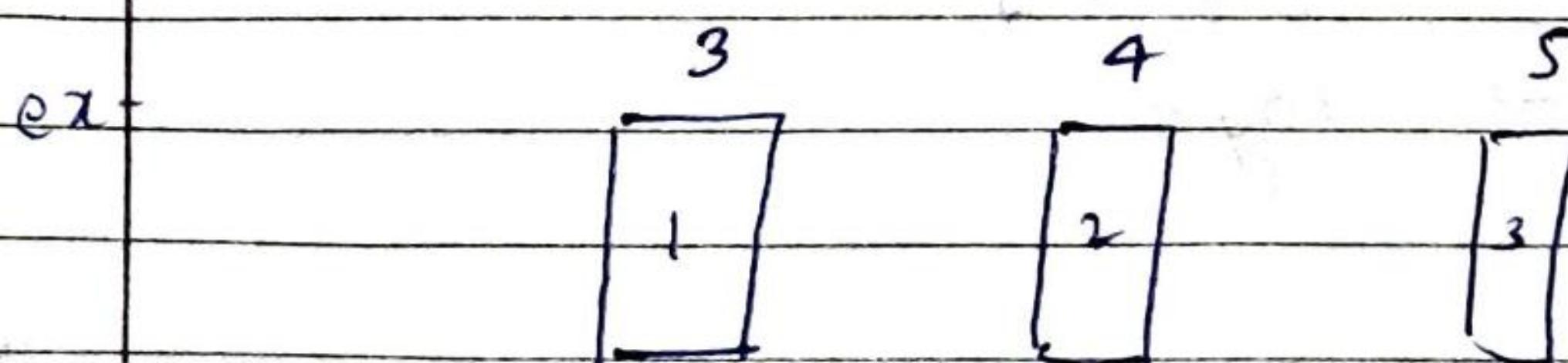
8

Player 1

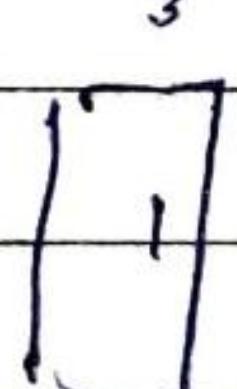
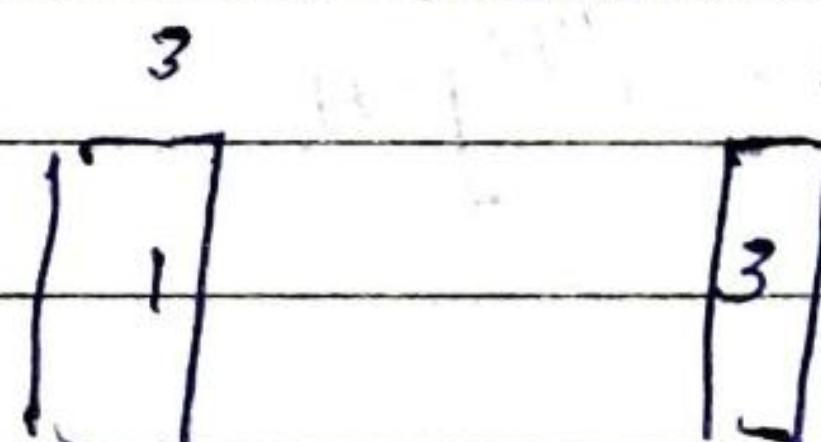
Player 2

Can remove any no. of file.

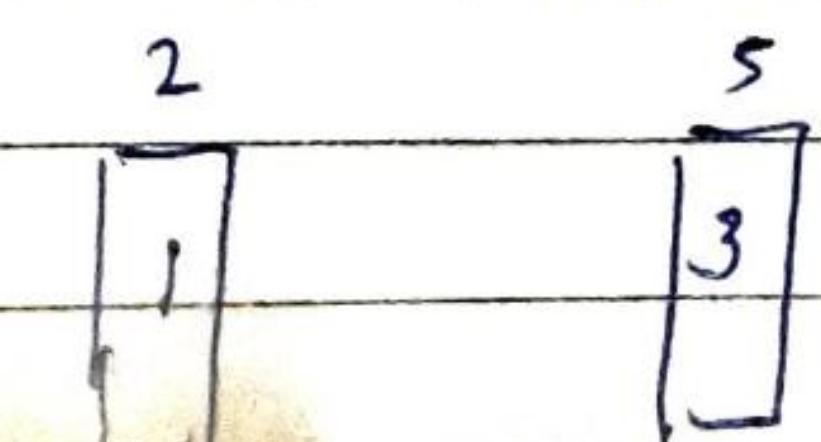
In the end who will be the winner



P1 - 2 → remove all



P2 → 1 → remove 1



P1 → 1 →

remove all



P2 → 3 → remove all

Player 1 lost

Player 2 wins

Will player 1 always loose?

Not necessary.

→ Nim sum theorem

Find Nimsum

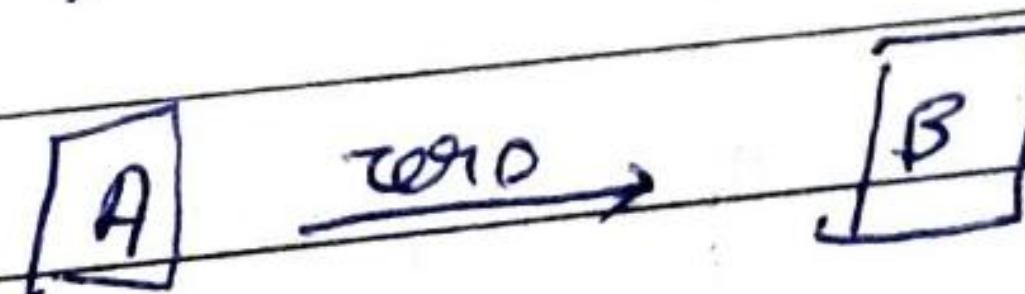
i.e. Commutative XOR of all piles

$$3 \wedge 4 \wedge 5.$$

$$\text{Nimsum} = \begin{cases} 0, & \text{Player 1 looses} \\ \neq 0 & \text{Player 1 wins} \end{cases}$$

On condition that both plays optimally  
and if  $n \rightarrow$  even, play 1 wins

Proof of Nimsum



Not needed

Max = minimum excluded set

max {0, 1, 3}, ans = 2

max {0, 1} ans = 2

Grundy numbers.

Grundy numbers = max {Grundy(n-1),  
Grundy(n-2)}

Grundy(0) = 0.

Grundy(1) = {g(0)} = {0} = 1

Grundy(2) = {g(0), g(1)}  
= {0, 1} = 2

⋮  
⋮

Grundy(n) = {g(0), g(1), g(2),  
..., g(n-1)} = n

Grundy number is a no. that defines a state of a game.

n stones → 1, 2, 3 can be removed.

Grundy(n) = max {g(n-1), g(n-2), g(n-3)}

Here,  $g(0) = 0, g(1) = 1, g(2) = 2$

$g(3) = 3, g(4) = 0$

$g(5) = 1, g(6) = 2, g(7) = 3$

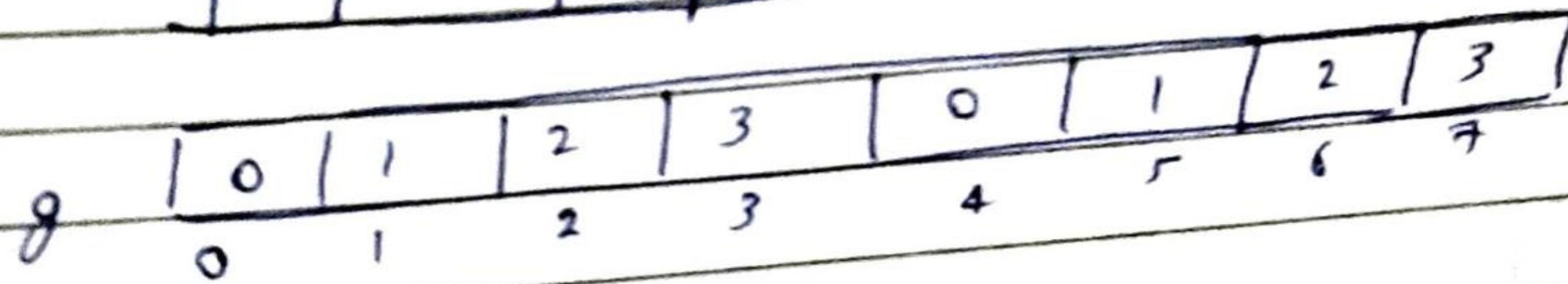
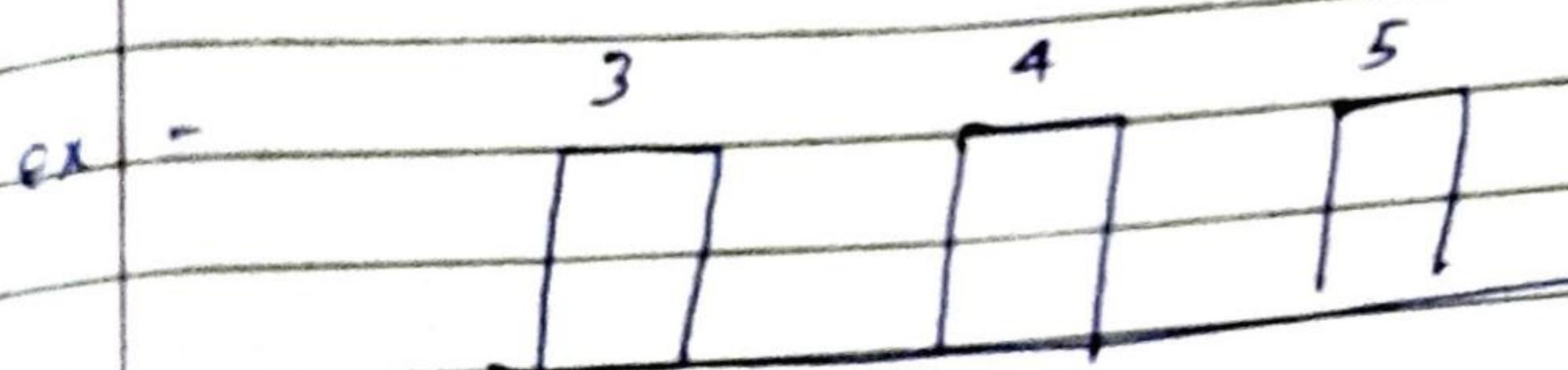
Date			
Page No.			

Sprague Grundy theorem

$$\text{Cumulative XOR} = (g(p_1) \wedge g(p_2) \wedge \dots)$$

$$\text{XOR} = \begin{cases} 0 & , \text{ player 1 loses} \\ \neq 0 & \text{player 1 wins} \end{cases}$$

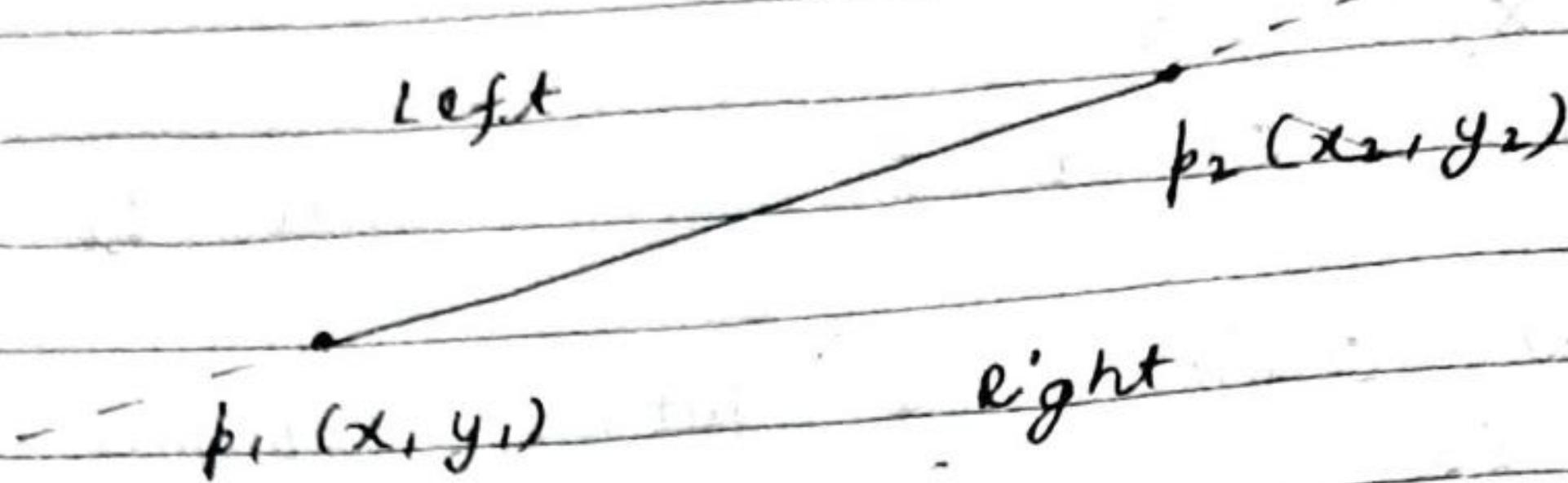
on condition that both players plays optimally.



$$\text{XOR} = g(3) \wedge g(4) \wedge g(5) \\ = 3 \wedge 0 \wedge 1$$

## Computational geometry

→ Point location test (CSES)

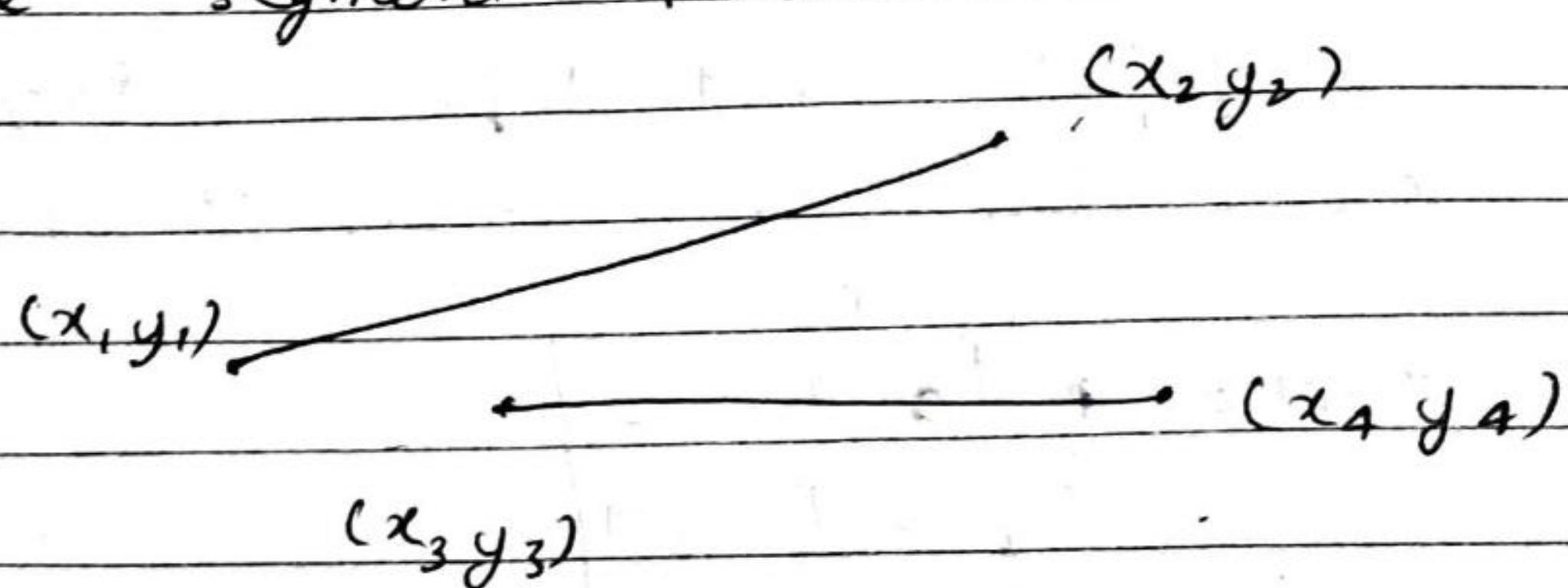


$$m = \frac{dy}{dx} = \frac{y_2 - y_1}{x_2 - x_1}$$

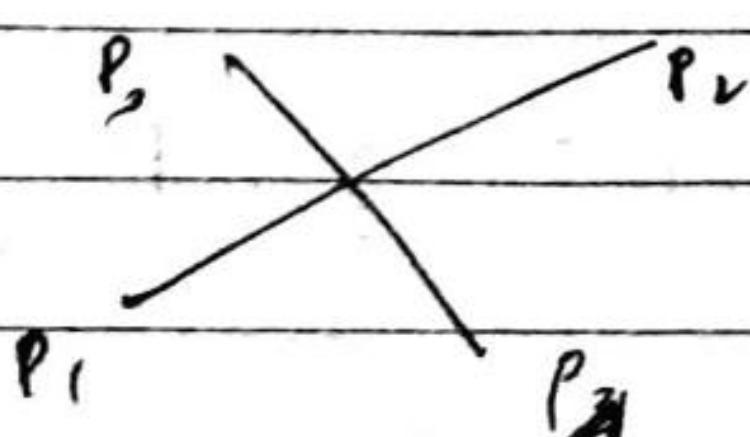
$$m' = \frac{dy'}{dx'} = \frac{y_3 - y_2}{x_3 - x_2}$$

```
if (m' - m > 0) { cout << "Left"; }
else if (m' - m < 0) { cout << "Right"; }
else cout << "On the line";
```

→ Line segment intersection - (CSES)



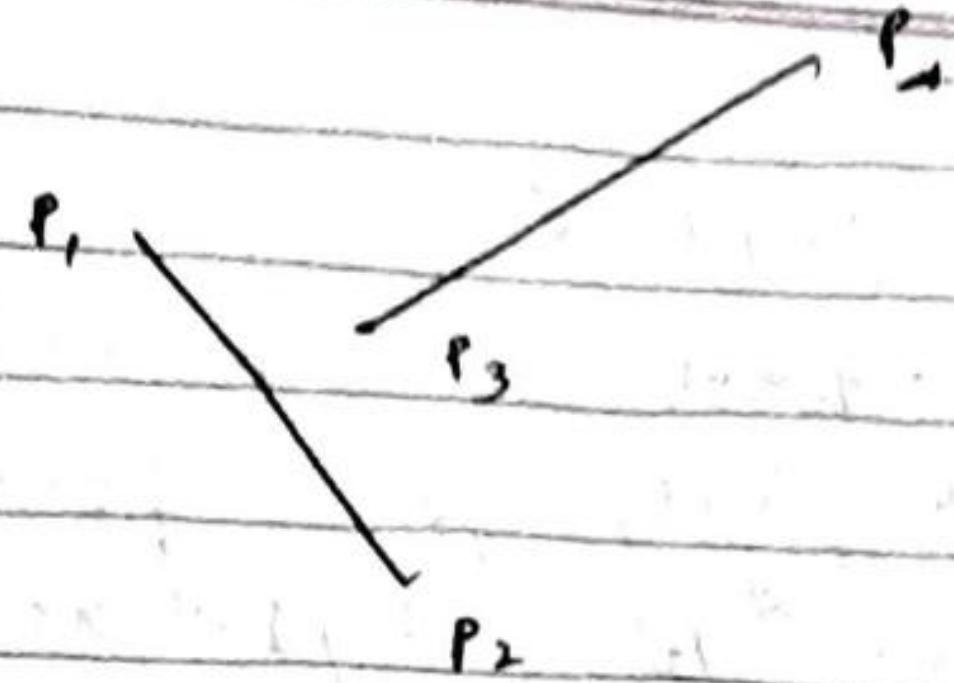
Algo



$P_1, P_2, P_3 = \text{Left}$

$P_1, P_4$  orientation must  
be different for each  
point

ex-



$$m_1 = \frac{y_2 - y_1}{x_2 - x_1}, \quad m_2 = \frac{y_3 - y_1}{x_3 - x_1}$$

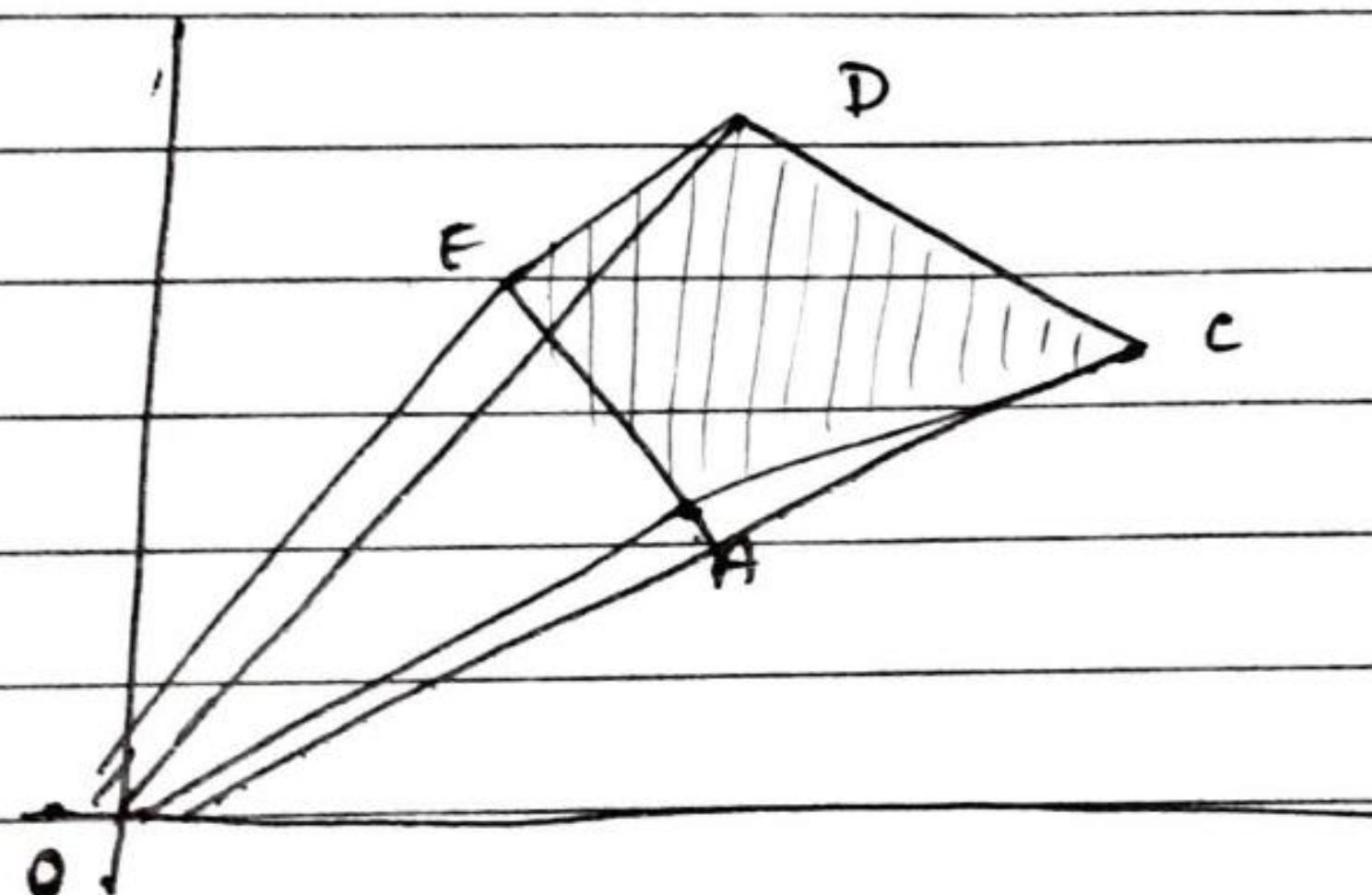
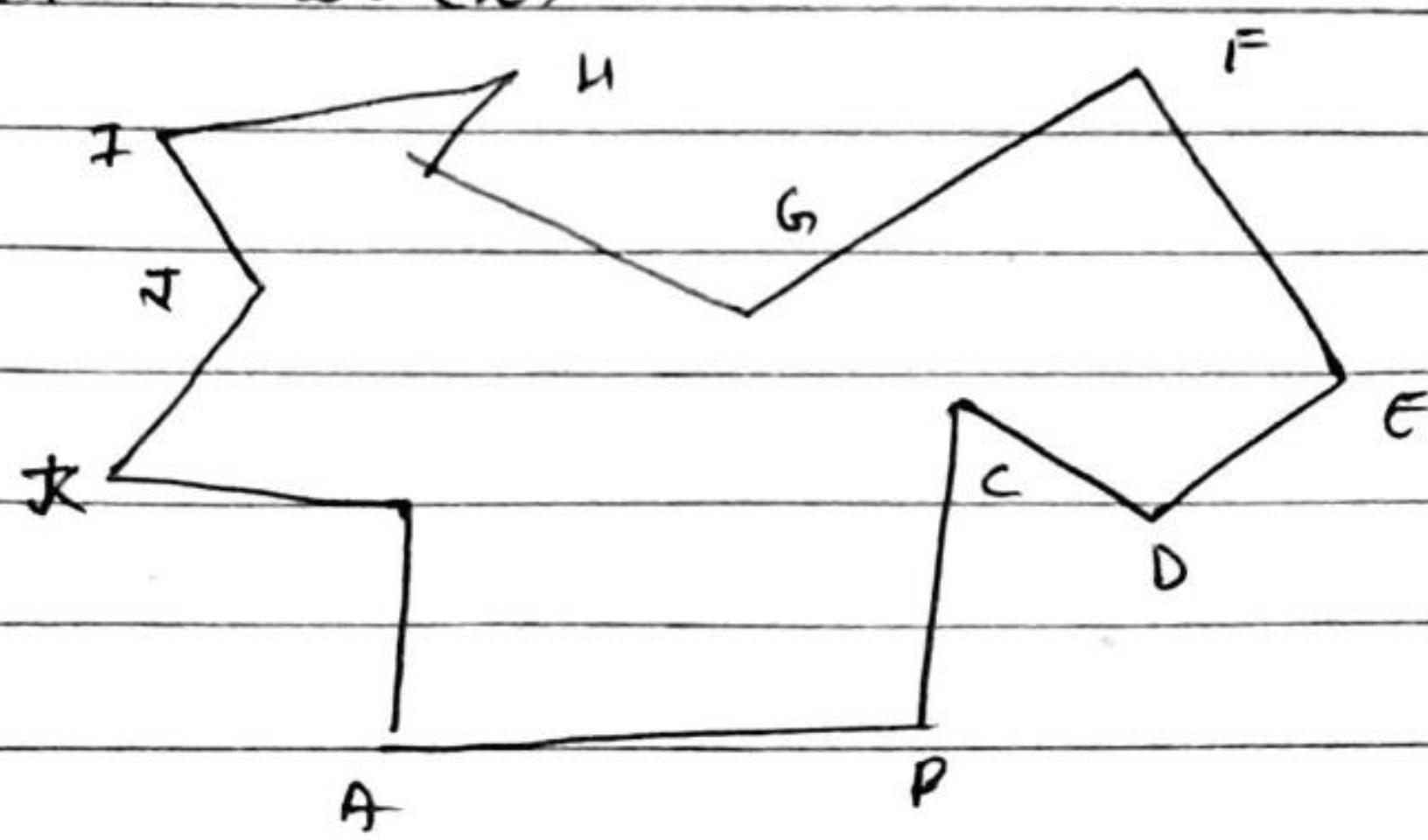
```

if ( $m_1 == m_2$ ) { return true; }
if ( $o1 != o2 \text{ } \&& \text{ } o3 != o4$ ) { return true; }
return false;

```

→ Polygon area - CSES

in > vertex;

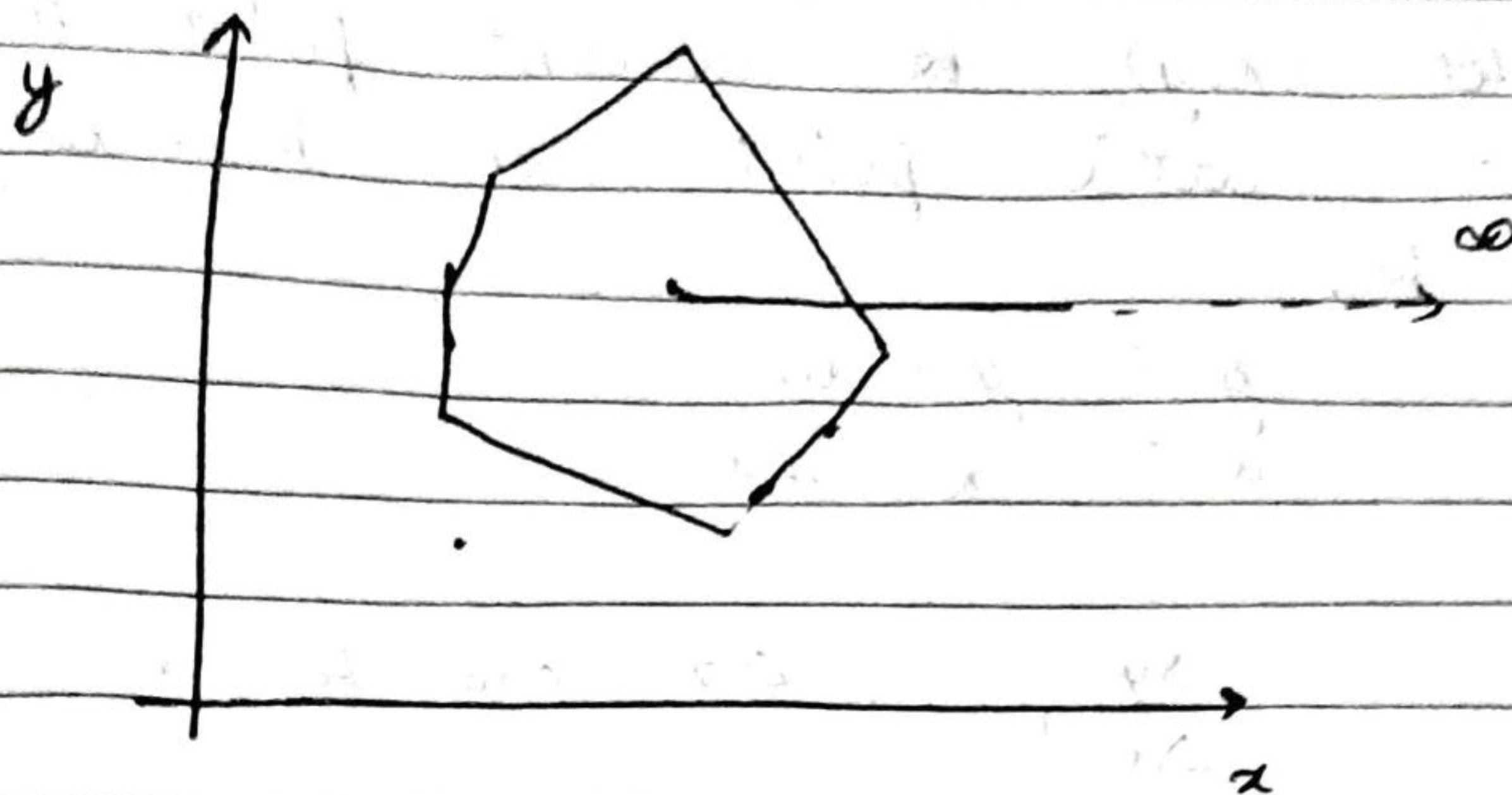


$$\text{OAC} + \text{OAD} - \text{ODE} - \text{OEA}$$

(Direction is important in cross product)

```
float ans = 0;  
for (i=0; i<n; i++) {  
    x1 = point[i].first, y1 = point[i].second  
    x2 = point[(i+n)%n].first, y2 = point[(i+1)%n].second  
    ans += ((x1 * y2) - (y1 * x2));  
}  
cout << abs(ans) << endl;
```

Point in a polygon



No. of intersection = odd  
(Inside polygon)

= even

(Outside polygon)

For outside on the line,

$(x, \infty)$

Lattice point

Rick's theorem.

$$A = i + \frac{b}{2} - 1$$

area ← ↓ → L.P. on  
lattice points boundary  
(inside the polygon)

boundary lattice points

$$m = \frac{dy}{dx}$$

let  $(x_1, y_1)$  be current point then  
nearest lattice point on boundary  
will be

$$\begin{aligned}y' &= y + dy \\x' &= x + dx\end{aligned}$$

But  $\frac{\Delta y}{\Delta x}$ ,  $\Delta x$  can be 0.

then  $a = (x_2 - x_1)$   
 $b = (y_2 - y_1)$

$$\gcd(a, b) + 1$$

(But it will include the given  
points as well).

use  $\gcd(a, b) - 1$  to find  
only boundary lattice excluding  
given points