Bit Manipulation

Raj
DATE / /
PAGE

* $a << b = a \times 2^b$

Common Patterns

1) Checking if i^{th} bit is set or not
2) Check if no. is odd / even $(n \& 1)$ < 1 = odd, 0 = even
3) Check if power of 2 $n \& (n-1)$ < 1 = No, 0 = Yes
* 4) Count no. of set bits

$(n \& n-1)$ → $n--$ → $n \& (n-1)$ → $n-1$

(! = 0)

Unit zero comes = no of steps = no of set bits

5) Swap 2 numbs (by XOR)

$a = a \wedge b$ $b = a \wedge b$ $a = a \wedge b$

(7) Divide 2 no. (Without '/', '%', '*')

Process

dividend = 22
divisor = 3

$\left(\frac{22}{3} = 7\right)$

$7 = 2^2 + 2^1 + 2^0$

> We will take out big no. from dividend until dividend is less than divisor

eg:

3×2^3 ✗
3×2^2 ✓ = 12
3×2^1 ✓
3×2^0 ✓

22 − 12 = 10 − 6 = 4 − 3 = 1

3×2^2 ✗
3×2^1 ✓
3×2^0

3×2^0

Here, divisor > dividend
So STOP

DATE / /
PAGE

Code

```
n = abs (dividend) , d = abs (divisor

while ( n >= d )
{
    cnt = 0 ;
    while ( n >= ( d x 2^(cnt+1) ) )
    {
        cnt ++ ;
    }
    ans += 2^cnt ;    // remove previous / last addable power.
         ||
         1 << cnt ;

    // update 'n'
    n = n - (d x (2^cnt)) ;
}
```

$\Rightarrow d << cnt+1 = d \times 2^{cnt+1}$

★ Max Shift allowed in 32 bit system is 31.★

eg : 1 << i   i cannot be more than 31

'i' [0 to 31]

Print all possible Sub SEQUENCES

∞ Bits Mam. edition

Raj

DATE / /

PAGE

- No of Subsets = 2^n

eg: nums = - [1, 2, 3]

Index 0 1 2

Index:	2	1	0	
0 ←	0	0	0	[ ]
1 ←	0	0	1	[1]
2 ←	0	1	0	[2]
3 ←	0	1	1	[1,2]
4 ←	1	0	0	[3]
	1	0	1	[1,3]
	1	1	0	[2,3]
2^n	1	1	1	[1,2,3]

eg: 1 2 3

[ ] [ ] -

[1]

[2]

[1,2]

[3]

[1,3]

[2,3]

[1,2,3]

```
for (index=0 ; index < 2^n ; index++)
{
    vector <int> subset ;
    for (i=0 to n)
    {
        if (index of (1<<i)) subset.push( arr[i]);
    }
    powerset .push (subset)
}
```

every time index of arr[i]

0 (excluded) 1 (included)

Check if ith bit set in 'i'

Raj
DATE / /
PAGE

Q Find 2 element appearing odd times while other even.

eg: [1,2,1,3,5,2] Output: [3,5]

$x = a \wedge b$

Once you XOR all element you will get $x = 3 \wedge 5$

Now

1) Find the first (rightmost) set bit where a, b differ
   ↳ $x \& (-x)$
2) Divide array on basis of that bit in 2 parts
3) XOR x with both groups which uniquely gives a|b seperately

Code

```
int xor_all = 0
for (i=0 to n)   x = x ^ arr[i];
```

We get $x = a \wedge b$ {a, b are answer}

```
int setbit = x & (-x);
```
rightmost setbit that differ a, b

```
int a=0, b=0
for (i=0 to n)
{ if (arr[i] & setbit)
    {
      a = a ^ arr[i]
    }
  else {
      b = b ^ arr[i];
  }
```

XOR ~~Multiple~~ a with one half which have that same set bit

And XOR ~~Multiple~~ b with other half not having set bit

Print all prime factors of N

Raj
DATE / /
PAGE

eg 24 1, 2, 3, 6, 8, 12, 24

Since we only need to travel 2 to $\sqrt{N}$

eg: 780

e	780
2	390
3	195
5	65
13	13

$O(\sqrt{N} \times \log n)$

```
for (i=2 to √N)
{
    if (N%i==0)
    {   v.push-back(i);
we will divide it with same factor until it can't
        while (n%i==0) n=n/i;
    }
}
if(N!=1) v.push-back(N)
```

Print all divisors

eg: 24: 1 2 3 6 8 12 24

If i is factor of n then (n/i) is also a factor.

No need to traverse if all the way

Traverse only till $\sqrt{N}$

```
for ( i=1 to sqrt(N))
  { if (N%.i==0)
      { arr.push-back(i)
        if (n/i!=i) arr.pushback(n/i);
      }
  }
```

Check for Perfect square so that no don't repeats

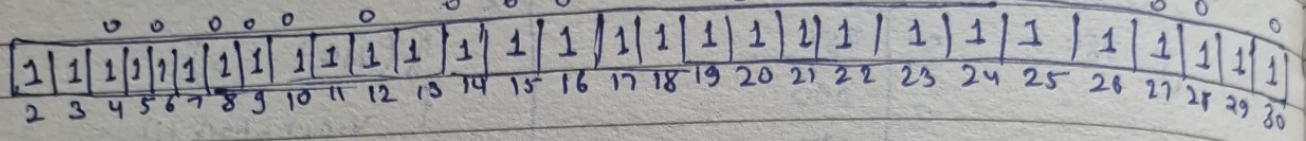
$O(N \log(\log N))$

Raj

DATE / /
PAGE

SIEVE OF ERATOSTHENES

Q Given a no. (N). Print all prime no 1 to N



NO Iterate from 2 to $\sqrt{N}$ and unke factor ko zero karda

repeat → so start with i*i

2x2 ✓ 3x2

2x3 3x3 ✓ → 4x2 4x3

2x4 ← 3x4 5x5 . . . { }

3x5 4x4 ✓ 5x6

4x5

2 to 5 (i.e 2 to $\sqrt{N}$)

```
for (2 to √N)
{
  if (arr[i] == 1) // Prime ✓
  {
      for (j = i^2 to N)        j = j + i
      {
          arr[j] = 0;
      }
  }
}
```

Make the factor of the prime zero.