Bit Manipulation Techniques

① 10000

Basic Ideas.

and , or , shift, xor

Left bit set w nst?

$$n \& (1 << k-1)$$

setting left bit

$$n \mid (1 << k-1)$$

Cleaning left bit

$$n \& \sim(1 << k-1)$$

So getting left bit

$$n \wedge (1 << k-1)$$

Toggling right most bit.

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

Isolating rightmost 1 bit

$$(n \& -n)$$

Isolating rightmost zero bit

$$(\sim n \& n+1)$$

Check $2^n \sim n$

if $(n \& n-1 == 0)$

Multiplying by a power of 2

$$n \times 2^k$$
$$(n << k) \quad i > > k$$

Modulo of a given #.

$$n \& 0 \times 1F \qquad n \% F$$

reversing binary #:

```
nt = n;
s = size of (n)
while (n >> = 1)
{   nt << = 1;
    nt |= n & 1;
    s--;
}
nt << = s;
```

# of 1's:

```
while (n) {
    count f = (n & 1);
    n >> = 1;
}
```

Mark for trailing zeroes:

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

Swap odd and even bits:

$$n = 0\,1\,0\,0\,0\,1\,0\,1\,1$$

en N = n & 0x AA   0000 1010
odd N = n & 0x 55   0100 0
                     0 1 0 0 0 0 0

even N >> = 1
odd N << = 1

final = even N | odd N

left rotate by d bits
n << d | n >> (INT_BITS - d);

right
n >> d | n << (INT_BITS - d);

abs method
n >> n >> 31;
mask = n >> 31;
mask + n;
(mask + n) ^ mask;

swap nibbles
((n & 0x0f) << 4) | ((n & 0xf0) >> 4)

while (y != 0)
{
  c = x & y;
  x = x ^ y;
  y = carry << 1;
}