Todays Content

1. Count Set Bits
2. Set a & yth bit
3. Continous n set bits & y unset bits
4. Unset ith Bit

Revision:

```
int Set(int N, int i){
    N = N | (1<<i)   #Set ith bit in N
    return N;
}
```

```
bool checkBit(int N, int i){
    return (N>>i)&1;
}
```

Count Set Bits:

Given N, return no: of set bits in N.

Eg1: $2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

N = 21 → 0 1 0 1 0 1 → return 3

N = 45 → 1 0 1 1 0 1 → return 4

Idea1: Iterate on every bit of N & if it's set inc cnt;

```
int cBit(int N){ TC: 32 iterating = O(1)
    int c=0;
    for(int i=0; i<32; i++){
        if((N>>i)&1==1){ # if ith Bit in N is set or not
            c++;
        }
    }
    return c;
}
```

Idea2:

```
int countSet(int N){ TC: O(log2 N)
    int c=0;
    while(N>0){
        if(N&1==1){
            c++;
        }
        N=N>>1; # N=N/2
    }
    return c;
}
```

	2^5 2^4 2^3 2^2 2^1 2^0	N&1==1	c = 0
N = 45	1 0 1 1 0 1	✓	c = 1
N>>1	1 0 1 1 0	X	c = 1
N>>1	1 0 1 1	✓	c = 2
N>>1	1 0 1	✓	c = 3
N>>1	1 0	X	c = 3
N>>1	1	✓	c = 4
N>>1	0	* return c = 4	

Issue: To update variable, we need to use = operator.

```
N=10;
print(N+10);//20
print(N);//10
```

```
N=10;
print(N>>1);//5
print(N);//10
```

#Comp:	#idea1	#idea2	#idea3: #No: of TODO
N=45	32	6	4

3rd Approach:

N		N-1		N & (N-1)
	2^6 2^5 2^4 2^3 2^2 2^1 2^0		2^6 2^5 2^4 2^3 2^2 2^1 2^0	
N=49	0 1 1 0 0 0 1	N=	_ _ _ _ _ _ _	_ _ _ _ _ _ _
N=24	0 0 1 1 0 0 0	N=	_ _ _ _ _ _ _	_ _ _ _ _ _ _
N=8	0 0 0 1 0 0 0	N=	_ _ _ _ _ _ _	_ _ _ _ _ _ _
N=52	0 1 1 0 1 0 0	N=	_ _ _ _ _ _ _	_ _ _ _ _ _ _
N=42	0 1 0 1 0 1 0	N=	_ _ _ _ _ _ _	_ _ _ _ _ _ _

#Obs: N & (N-1)

```
int countSetBits(int N){ TC: #No. of set bits.
    int c = 0;
    while(N > 0){
        N = N & (N-1)
        c++;
    }
    return c;
}
```

28 Given x & y: Set x^{th} & y^{th} Bit in 0.

Constraints

$0 <= x, y <= 30$

fn1:

$x = 3$ $y = 5$ ans = 40

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	0	1	0	0	0

fn2:

$x = 2$ $y = 4$ ans = 20

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	1	0	1	0	0

fn3:

$x = 3$ $y = 1$ ans = 10

2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	1	0	1	0

```
int setBits(int x, int y){

    return (1<<x) | (1<<y);

    return (1<<x) + (1<<y);

}
```

if x == y: Code fails.

Please check with a example

38 Given x, y set consecutive x bits & y unset bits.

Constraints

$0 <= x + y <= 30$

$x=3\ y=2$ ans $=28$ | $x=4\ y=3$ ans $=120$ | $x=2\ y=5$ ans $=96$

$2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$ | $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$ | $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

1 1 1 0 0 | 1 1 1 1 0 0 0 | 1 1 0 0 0 0 0

Ideas

1. Using loops: TODO
2.
   1. $x=3\ y=2$

      $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

      0 1 1 1 $= 2^3 - 1$

      $(2^3-1) << 2$: 0 0 1 1 1 0 0 $= 2^4 + 2^3 + 2^2 = 28$

   2. $x=4\ y=3$

      $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

      1 1 1 1 $= 2^4 - 1$

      $(2^4-1) << 3$: 1 1 1 1 0 0 0 ✓

   3. $x=2\ y=5$

      $2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$

      1 1 $= 2^2 - 1$

      $(2^2-1) << 5$: 1 1 0 0 0 0 0 ✓

Con: x: set y: unset

$ans = (2^x - 1) << y \rightarrow ((1 << x) - 1) << y \rightarrow (1 << (x+y) - 1 << y)$

Q8: Given N & i : Unset i^{th} bit in N, if already unset leave it.

Hint: &0

Eg1:

N=45 i=3

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unset i=3	0	1	0	1	1	0	1
N=37	0	1	0	0	1	0	1

N=57 i=3

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unset i=3	0	1	1	1	0	0	1
N=49	0	1	1	0	0	0	1

#Ideal:

N=45 i=3

Unset i=3

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
N=45	0	1	0	1	1	0	1
(1<<3)	0	0	0	1	0	0	0
~(1<<3)	1	1	1	0	1	1	1
N & ~(1<<3)	0	1	0	0	1	0	1

N=57 i=4

Unset i=4

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
N=57	0	1	1	1	0	0	1
(1<<4)	0	0	1	0	0	0	0
~(1<<4)	1	1	0	1	1	1	1
N & ~(1<<4)	0	1	0	1	0	0	1

```
int UnSet(int N, int i){
    N = N & ~(1<<i)          or    N = N - (1<<i) * ((N>>i) & 1)
    return N;                      return N;
}
```