Bit Manipulation

Monday, 8 November 2021 9:05 PM

1 → True
0 → False

A = 1 | 0
~A = 0 | 1

0 – 9, 10, 11, 12, 13, :, 19, 20, 21, 2:

0 ← 0
1 ← 1
10 ← 2
11 ← 3
100 ← 4
101 ← 5
110 ← 6
111 ← 7
1000 ← 8
1001 ← 9

A	B	A & B	A | B	A ^ B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

1 << 2 → 100 → 4

2^3 → 0 . . 0 1 0 1 1 1 (32 bits): 32 16 8 4 2 1 → 16 + 4 + 2 + 1 = 23

23 << 1 → x * 2
32 16 8 4 2 1
. . 0 1 0 1 1 1 0 ← 46
32 + 8 + 4 + 2 = 46

30 – 16 = 14, – 8 = 6, – 4 = 2 → . . . 32 16 8 4 2 1: . . . 0 0 0 1 1 1 1 0

2 | 30
2 | 15 | 0
2 | 7 | 1
2 | 3 | 1
 | 1 | 1

11110 = 30

x << 1 → x * 2

x << 2 = x * 4

x << i = x * 2^i ✓

x >> i = x / 2^i

23 >> 1 → 11
. . 1 0 1 1 (1)
16 8 4 2 1
1 0 1 1 → 11

23 → 1 0 1 1 0, 1 0 1 1 = 11

20 → 1 0 1 0 0
20 >> 1 = 1010 = 10

---

11 → 1 0 1 (1) LSB

x & 1 → 1 → x = odd
 → 0 → x = even

x = 11 → 1011
& 0001
0001 ✓

6 → 0110
& 1 → & 1
0000

10 & 12
1010
& 1100
1000 → 8 ✓

10 | 12 = 14
1010
| 1100
1110 → 14

8 | 7 = 15
1000
0111
1111

x | 1
1010
or 0001
1011 → 11
or 0001
1011 → 11

10 ^ 11
1010
XOR 1011
0001 → 1 ✓

10 ^ 10 = 0

x ^ x = 0

11 ^ 12 = 7
1011
1100 > 0111

Q → Check if i^{th} bit is set.

x = 10, i = 2 → 0 0 . . . 1 0 1 0 (bits . . . 4 3 2 1 0)

Ans = True / 1

Sol1 → (x >> i) & 1 → 1 / 0 ✗

10 >> 1 = 5 → 10(1)

TC = O(i) ≈ O(constant)
$O(\log_2(N))$

Sol2 → x & (1 << i) → 0 ✗ / (1 << i) ✓
– 0 0 0 0 1 0 0 0 – – (i^{th} bit)

1010
0010
0010 = (1 << i)

x & 2^i

1 << i = 2^i
$i <= 2^{32}$
1 * 2 * 2 * 2 ... i times
$i <= \log_2(N)$
i <= 32
1 ← 1
10 ← 2
100 ← 4
1000 ← 8 . . .

x = 13 → 1 1 0 1 (3 2 1 0)
i = 2 → . . . 0 1 0 0 AND
. 0 0 1 0 0

x << i ✓✓
x * 2^i slow ✓

$\log_2(2^{32}) = 32$

(10 . 2) → binary representation?
→ 1010.

–10
0 . . . 0 0 1 0 1 0
1 . . . 1 1 1 0 1 0 1 ← 1's compliment → flip bits
+ 1 ← 2's compliment → add 1 ✓
1 . . . 1 1 0 1 1 0

Q → Check if x is power of 2. ($x = 2^k$ for some k) → 1 << k

x = 8 Ans = True
x = 6 Ans = False

1000 → only one bit is set
0110

0 0 0 – – 1 1 0 0 1 0 1 0 1 1 0 6
– 1
. . . 0 1 0 1 1

8 – 1 = 7
1 0 0 0
0 1 1 1

3 2 1 1
– 7
3 1 4 ✓

1 0 1 (1 0 0 0) → 1 0 1 | 0 1 1 1
– 1

2 3 0 6
– 8
2 9 2 ✓

1 0 1 0
– 1
1 0 1 0 0

(x – 1) → makes right most 1 = 0 & everything on left = 1

x & (x – 1) → rightmost 1 becomes 0 ✓

1000
0111
0000

1 1 0 1 0 0 (0 1)
– 1
1 1 0 0 1 1
& 1 1 0 1 0 0
1 1 0 0 0 0 ✓

x = 20 → 1 0 1 0 0
19 = (x – 1) → 1 0 0 1 1
1 0 0 0 0

5 6 0 0 3
– 8
5 1 1 5 ✓

x & (x – 1) → 0 ⇒ x is a power of 2 ✓
 → else → not a power of 2

count = 0;
while (x > 0) {
count ++;
x = x & (x – 1); ✓

x = 16 → 1 0 0 0 0 (16 8 4 2 1)
(x – 1) = 15 & 0 1 1 1 1
0 0 0 0 0 ✓

TC = O(#set bits) ✓ (3)

Q → Given an array of integers, every number is repeated twice except for one number. Find that number. (GS)

A = [1 ^ 2 ^ 1 ^ 2 ^ 3 ^ 3 ^ 2 ^ 5 ^ 7 ^ 7] (Ans)

Ans → 5

Ans → XOR of all numbers.

x ^ x = 0
(x ^ x) ^ y = y
0 ^ y = y

Q → Every number appears twice except for 2 numbers. Find both numbers. (SAP)

x ^ y = y ^ x
x ^ y ^ x = x ^ x ^ y = y

A = [2, 7, 3, 9, 11, 11, 2, 3, 5, 5]

Ans = 7, 9

→ sort & travel ✓ TC = O(N $\log_2$(N)) SC = O(1)
→ use frequency. if freq. == 1 ⇒ ans ✓ TC = O(N) ✓ SC = O(N) ✓

2 2, 3 3, 5 5, 7, 9, 11 11

XOR of all numbers = x ^ y ✓ where x, y → ans

p = 0;
for (j = 0; j < n; j++)
(P) = p ^ A[j];

≥ 0 ✓

TC = O(N)

7 ^ 9 = 0 1 1 1 (3 2 1 0)
^ 1 0 0 1
1 1 1 0 → 14

→ either set for x or y but not for both.

2 7 3 9 11 11 2 3 5 5
3 → 11, 9 → 1001

TC = O(N)
SC = O(1)

i = 1 set
2, 7, 3, 11, 11, 2, 3

i = 1 unset
9, 5, 5

XOR all elements → 7 → (Ans) ← 9

0 ^ A = A

x = y = 0;
∀ j from 0 – (n–1)
→ x = x ^ A[j] if i^{th} bit is set for A[j] ←
→ y = y ^ A[j] if i^{th} bit is unset for A[j] ←

x = 20 → 1 0 1 0 0
x – 1 = 19 → 1 0 0 1 1
AND 1 0 0 0 0 → 16

20 → 1 0 1 0 0 (4 3 2 1 0)
– 1 0 0 0 0
0 0 1 0 0 → 4 → 2

x – (x & (x – 1)) → Right most set bit of x.
x = 20 → 4

10 → 9 % 2 = 1

30 & 4 → 4 ✓
→ 0 ✗ binary
11110
10 → 2

1
13
+ 19
32

x → 1 0 0 1 0 1 1 0 0
– 1 0 0 1 0 1 0 0 0
0 0 0 0 0 0 1 0 0 → 4

x → 1 0 0 1 0 1 1 0 0
(x – 1) → 1 0 0 1 0 1 0 1 1
x → & 1 0 0 1 0 1 1 0 0
1 0 0 1 0 1 0 0 0

1 1 1 1
1 1 1 0 1
+ 1 1 0 1 1
1 1 1 0 0 0 ✓

1 → 3

A – (everything in A but x) = x

Count no. of set bits for all nos. from 1 to N

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

x/2

[x/4] * 2 + (x % 4 <= 2 ? 0 : (x % 4) – 1);

x % 4 == 2 → +1
== 3 → +2

x * 4 + (8)

0 ^ A = A

b = 4 ⇒ 2^{nd} bit is set

b = XOR of all elements
b = b – (b & (b – 1))
x = y = 0;
for (i = 0 ; i < n ; i++) {
if (A[i] & b > 0) x = x ^ A[i];
else y = y ^ A[i];
}
print (x, y);