

**BINARY**

# Bit Manipulation in C

- **Operators**

**&** AND    **|** OR    **^** XOR    **~** NOT    **<<** LEFT SHIFT    **>>** RIGHT SHIFT

- **Common Operations**

**set bit:** `num |= (1 << pos)`

**clear bit:** `num &= ~(1 << pos)`

**toggle bit:** `num ^= (1 << pos)`

**check bit:** `(num & (1 << pos)) != 0`

**extract bit:** `(num >> pos) & 1`

**extract a range of bits:** `(num >> pos) & ((1 << length) - 1)`

- **Example**

```
void copyBit(int *dst, int src, int srcPos, int dstPos) {  
    int bit = (src >> srcPos) & 1;  // extract bit  
    *dst &= ~(1 << dstPos); // clear destination bit  
    *dst |= (bit << dstPos); // set destination bit  
}
```

# Binary

- In C++, **std::bitset** represents a fixed-size sequence of N bits

- Example:

```
std::bitset<8> bitmask;
```

```
bitmask.reset(1)
```

```
bitmask.set(1)
```

```
if (bitmask.test(1)) { // true
```

```
...
```

- **reset** : set bit to false
- **set** : set a specific bit
- **test** : check a specific bit
- **count** : return the number of bits set to true
- **flip** : toggle the value of the bits (if true, set to false and vice-versa)

# Problem – 371. Sum of Two Integers

Medium



LeetCode

[leetcode.com/problems/sum-of-two-integers](https://leetcode.com/problems/sum-of-two-integers)

## Problem

- ...

# Problem – 371. Sum of Two Integers

Medium



LeetCode

[leetcode.com/problems/sum-of-two-integers](https://leetcode.com/problems/sum-of-two-integers)

## Solution

■ ...

# Problem – 371. Sum of Two Integers

Medium



LeetCode

[leetcode.com/problems/sum-of-two-integers](https://leetcode.com/problems/sum-of-two-integers)

**Code** Time:  $O(-)$  Space:  $O(-)$

■ ...

# Problem – 191. Number of 1 Bits

Easy



LeetCode

[leetcode.com/problems/number-of-1-bits](https://leetcode.com/problems/number-of-1-bits)

## Problem

■ ...

# Problem – 191. Number of 1 Bits

Easy



LeetCode

[leetcode.com/problems/number-of-1-bits](https://leetcode.com/problems/number-of-1-bits)

## Solution

- ...



# Problem – 191. Number of 1 Bits

Easy



LeetCode

[leetcode.com/problems/number-of-1-bits](https://leetcode.com/problems/number-of-1-bits)

## Code

- ...

# Problem – 191. Number of 1 Bits

Easy

 [leetcode.com/problems/number-of-1-bits](https://leetcode.com/problems/number-of-1-bits)

**Problem Statement / Solution / Code**   Time:  $O(-)$    Space:  $O(-)$

■ ...

# Problem – 338. Counting Bits

Easy

 [leetcode.com/problems/counting-bits](https://leetcode.com/problems/counting-bits)

**Problem Statement / Solution / Code**   Time:  $O(-)$    Space:  $O(-)$

■ ...

# Problem – 338. Counting Bits

Easy



LeetCode

[leetcode.com/problems/counting-bits](https://leetcode.com/problems/counting-bits)

## Problem

■ ...

# Problem – 338. Counting Bits

Easy



LeetCode

[leetcode.com/problems/counting-bits](https://leetcode.com/problems/counting-bits)

## Solution

- ...

# Problem – 338. Counting Bits

Easy



LeetCode

[leetcode.com/problems/counting-bits](https://leetcode.com/problems/counting-bits)

**Code** Time:  $O(-)$  Space:  $O(-)$

■ ...

# Problem – 268. Missing Number

Easy



LeetCode

<https://leetcode.com/problems/missing-number>

## Problem

■ ...

# Problem – 268. Missing Number

Easy



LeetCode

<https://leetcode.com/problems/missing-number>

## Solution

- ...



# Problem – 268. Missing Number

Easy



LeetCode

<https://leetcode.com/problems/missing-number>

**Code** Time:  $O(-)$  Space:  $O(-)$

■ ...

# Problem – 190. Reverse Bits

Easy



LeetCode

[leetcode.com/problems/reverse-bits](https://leetcode.com/problems/reverse-bits)

## Problem

■ ...

# Problem – 190. Reverse Bits

Easy



LeetCode

[leetcode.com/problems/reverse-bits](https://leetcode.com/problems/reverse-bits)

## Solution

- ...

# Problem – 190. Reverse Bits

Easy



LeetCode

[leetcode.com/problems/reverse-bits](https://leetcode.com/problems/reverse-bits)

**Code** Time:  $O(-)$  Space:  $O(-)$

■ ...

# Negabinary

- Non-standard positional numeral system that uses base of -2
- Allow representing negative numbers in binary
- Example:

$1101_{-2}$

$$(-2)^3 + (-2)^2 + 0 + (-2)^0 = -8 + 4 + 0 + 1 = -3$$

## Summing Negabinary

- Add as a regular binary number, but with **negative carry**

$$0 + 0 = 0$$

$$1 + 0 = 1$$

$$1 + 1 = 0 \quad \text{with a negative carry 1}$$

$$\mathbf{1} + 1 = 0 \quad (\text{subtract})$$

$$\mathbf{1} + 0 = 1 \quad \text{with a positive carry 1}$$

# Negabinary

## Example 1

$$\begin{array}{r} 11\ 1 \\ 1011 \\ + 1110 \\ \hline = 110001 \end{array}$$

$$1 + 0 = 1$$

$$1 + 1 = 0 \text{ with negative carry } 1$$

$$1 + 1 = 0$$

$$1 + 1 = 0 \text{ with negative carry } 1$$

$$1 + 0 = 1 \text{ with positive carry } 1$$

$$1 + 0 = 1$$

red 1 = negative carry

green 1 = regular carry

## Example 2

$$\begin{array}{r} 1111 \\ 101010 \\ + 101100 \\ \hline = 11110110 \end{array}$$

## Reference

<https://math.stackexchange.com/questions/3251605/how-to-add-negabinary-numbers>

# Problem 1073 – Adding Two Negabinary Numbers

Medium

<https://leetcode.com/problems/adding-two-negabinary-numbers>

Given two numbers `arr1` and `arr2` in base -2, return the result of adding them together.

Each number is given in *array format*: as an array of 0s and 1s, from most significant bit to least significant bit. For example, `arr = [1,1,0,1]` represents the number  $(-2)^3 + (-2)^2 + (-2)^0 = -3$ . A number `arr` in array, format is also guaranteed to have no leading zeros: either `arr == [0]` or `arr[0] == 1`.

Return the result of adding `arr1` and `arr2` in the same format: as an array of 0s and 1s with no leading zeros.

## Example 1

Input: `arr1 = [1,1,1,1,1]`, `arr2 = [1,0,1]`

Output: `[1,0,0,0,0]`

Explanation: `arr1` represents 11, `arr2` represents 5, the output represents 16.

## Example 2

Input: `arr1 = [0]`, `arr2 = [0]`

Output: `[0]`

## Example 3

Input: `arr1 = [0]`, `arr2 = [1]`

Output: `[1]`

# Solution 1073 – Adding Two Negabinary Numbers

Medium

<https://leetcode.com/problems/adding-two-negabinary-numbers>