Bit Representation

28 November 2020 05:45 PM

- An unsigned variable of n bits can hold number between 0 and 2^n-1
- A signed variable of n bits can hold number between -$2^{n-1}$and $2^{n-1}-1$, Here one extra bit is used for representing sign.
- Usually the negative are represented as two's complement
- A signed no. -X = an unsigned (complement form) $2^n - x$

Complements

I's :

Just invert the bits

2's:

Add 1 to 1's complement

Bit Operations

28 November 2020 06:29 PM

- And
- Or
- Xor : exclusive or
- Not
- Bit shifts : left and right

- Setting kth bit : x | (1 << k)
- Unsetting kth bit : x &~(1 << k)
- Inverting kth bit : x ^ (1 << k)

The g++ compiler provides the following functions for counting bits:
- __builtin_clz (x): the number of zeros at the beginning of the number
- __builtin_ctz (x): the number of zeros at the end of the number
- __builtin_popcount (x): the number of ones in the number
- __builtin_parity (x): the parity (even or odd) of the number of ones

While the above functions only support int numbers, there are also long long versions of the functions available with the suffix ll

Parity

It is the extra bit Used for checking that the number is received (example in network) correctly if a bit is altered we get to know.
Parity refers to the value of parity bit.

Even parity

In this case the Parity bit is set to 1 if the number has all odd 1 bits and 0 other wise so that in total a number will always has even ones.
Parity refers to even parity if odd is not mentioned.

Odd parity

In this case the Parity bit is set to 0 if the number has all odd 1 bits and 1 other wise so that in total a number will always has odd ones.

Used for checking that the number is received (example in network) correctly.
Parity refers to the value of parity bit.

Set Representation

01 December 2020 05:34 PM

Idea

Every subset of a set $\{ 0 , 1 , 2 ,...,n - 1 \}$ can be represented as an n bit integer whose one bits indicate which elements belong to the subset. This is an efficient way to represent sets, because every element requires only one bit of memory, and set operations can be implemented as bit operations.
For example, since int is a 32-bit type, an int number can represent any subset of the set $\{ 0 , 1 , 2 ,..., 31 \}$. The bit representation of the set $\{ 1 , 3 , 4 , 8 \}$ is 00000000000000000000000100011010 ,
which corresponds to the number 2^8 + 2^4 + 2^3 + 2^1 = 282.

Inserting element to a set

```
int x=0;
X|=(1<<i); //inserts i to the set
```

Counting the number of elements in a set implies counting the no of set bits in no.
Printing the elements of set

```
for (int i = 0; i < 32; i++) {
if (x&(1<<i)) cout << i << " ";
}
```

Set Operations

	set syntax	bit syntax
intersection	$a \cap b$	a & b
union	$a \cup b$	a | b
complement	$\bar{a}$	$\sim a$
difference	$a \setminus b$	a & $(\sim b)$

>>

Iterating through subsets

The following code goes through the subsets of $\{0, 1, \ldots, n-1\}$:

```
for (int b = 0; b < (1<<n); b++) {
    // process subset b
}
```

The following code goes through the subsets with exactly k elements:

```
for (int b = 0; b < (1<<n); b++) {
    if (__builtin_popcount(b) == k) {
        // process subset b
    }
}
```

The following code goes through the subsets of a set x:

```
int b = 0;
do {
    // process subset b
} while (b=(b-x)&x);
```

Anatomy of last code :

b=0

```
do {
//DO SOMETHING with subset b
} while (b=(b-x)&x);
```

What this code does is

First b starts with zero no bits

Then at the 'while bracket' two things happen

Boolean check of `(b-x)&x`

Assigning the value of `(b-x)&x` to `b`

Now how does (b-x)&x fires individual bits of X at a time is not known

XOR

10 February 2021 06:44 PM

- Distributive over | and &
- Not distributive over +

Headers

22 December 2020 05:58 PM

Headers	Functions	Remarks
bitset	bitset<--length of bit string-->(--the integer no.--)	Converts the integer to a bitstring
math.h	Pow,	
stdlib	strtoull	Convers char array of given radix to binary
algorithm	Max_element(startptr,endptr)	Returns pointer to max element
	Sort(startptr,endptr)	Ascending sort
	Sort(startptr,endptr,greater<int>())	Descending sort
	max	

Ordered & Unordered map

21 January 2021 06:36 PM

Ordered map

- Header : map
- Stores elements in a sorted (by key) order

Unordered map

- Header : unordered_map
- Stores elements in un sorted way
- It is actually a hash table

Iterating over a map

```
for (auto i:map)
    {
        cout<<i.first<<' '<<i.second<<"\n";
    }
```

String inputs

05 February 2021 07:07 PM

Cin always breaks the inputs at a space

Number theory

09 February 2021 06:33 PM

Sieve of Eratosthenes

Given a number n, print all primes smaller than or equal to n. It is also given that n is a small number.

N log(log(n))

todo

09 February 2021 10:45 PM

Number theory

- Properties of prime