Flipping bits

by shaka_shadows

You will be given a list of 32 bits unsigned integers. You are required to output the list of the unsigned integers you get by flipping bits in its binary representation (i.e. unset bits must be set, and set bits must be unset).

Input Format

The first line of the input contains the list size T, which is followed by T lines, each line having an integer from the list.

Constraints

$1 \leq T \leq 100$
$0 \leq integer < 2^{32}$

Output Format

Output one line per element from the list with the requested result.

Sample Input

```
3
2147483647
1
0
```

Sample Output

```
2147483648
4294967294
4294967295
```

Explanation

Take *1* for example, as unsigned 32-bits is *00000000000000000000000000000001* and doing the flipping we get *11111111111111111111111111111110* which in turn is *4294967294*.

Current Buffer (saved locally, editable) C++

```cpp
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <bitset>

using namespace std;


int main()
{
    int nListCount = 0;
    // vector<unsigned int> UIntVectorInput; // 32 bit unsigned integer vector
    cin >> nListCount;
    unsigned int nInput = 0;
    for( int i = 0; i < nListCount; i++ )
    {
        cin >> nInput;
        bitset<32> bInp( nInput );
        bInp.flip();
        // UIntVectorInput.push_back( static_cast<unsigned int>( bInp.to_ulong()));
        cout << static_cast< unsigned int >( bInp.to_ulong() ) << endl;
    }
    return 0;
}
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code