Синтаксис C++. Битовые операции.

Любое целое число можно представить в двоичной системе счисления. И именно в двоичной системе числа и хранятся в памяти компьютера. Однако, тут нужно заметить, что положительные и отрицательные числа представляются по-разному - если для положительных чисел это именно двоичное представление числа, то отрицательные значения хранятся в так называемом *дополнительном коде*. Дополнительный код позволяет заменить операцию вычитания на операцию сложения и нужен, главным образом, для упрощения вычислений с отрицательными числами с точки зрения архитектуры ЭВМ. Подробнее про него можно почитать, например, здесь или заглянуть в википедию.
В тексте программы на C++ целочисленные литеральные константы можно представлять не только в привычном нам десятичном представлении, но и двоичном:

```
int x = 0b0101; // x = 5
int y = 0b10000;  // y = 16
```

Вывести число на печать в двоичном представлении можно разными способами, например, с помощью класса bitset стандартной библиотеки:

```
#include <iostream>
#include <bitset>

int main() {
    int a = -58, b = 127, c = 315;

    std::cout << "a=" << std::bitset<8>(a)  << std::endl;
    std::cout << "b=" << std::bitset<8>(b)  << std::endl;
    std::cout << "c=" << std::bitset<16>(c) << std::endl;

    return 0;
}
```

Впрочем, это не самый эффективный способ, но для учебных целей подходит :-)

Поскольку целые числа хранятся в двоичном представлении - в виде набора битов, принимающих значения 0 и 1 - то к ним применимы функции алгебры логики - отрицание, конъюнкция (И), дизъюнкция (ИЛИ) и исключающее ИЛИ, работающие напрямую с ноликами и единичками, представляющими числа.

В C++, так же, как и в других языках, для этих функций есть специальные обозначения:

& –побитовое И (AND);
|–побитовое ИЛИ (OR);
^–побитовое ИСКЛЮЧАЮЩЕЕ ИЛИ (XOR);
~ –побитовое ОТРИЦАНИЕ (NOT);

Кроме этого, есть две операции циклического сдвига:
>> –циклический сдвиг вправо;
<< –циклический сдвиг влево;

Сдвиг влево сдвигает все биты влево на указанное количество позиций, при этом заполняет нулями освободившиеся биты. Сдвиг вправо сдвигает все биты вправо на указанное количество позиции, при этом несколько правых позиции отбрасываются.

Например, при выполнении следующего кода:

```
unsigned int a = 6, b = 5, c = 9;

std::cout << "a & b = " << (a & b);
std::cout << "a | b = " << (a | b);
std::cout << "a ^ b = " << (a ^ b);

std::cout << "a << 2 = " << (a << 2);
std::cout << "c >> 2 = " << (c >> 2);
```

будет выведено:

a & b = 4 (6 = 110, 5 = 101, 110 & 101 = 100)
a | b = 7 (6 = 110, 5 = 101, 110 | 101 = 111)
a ^ b = 3 (6 = 110, 5 = 101, 110 ^ 101 = 011)
a << 2 = 24 (6 = 110, 110 << 2 = 11000)
c >> 2 = 2 (9 = 1001, 1001 << 2 = 0010)

Во всех задачах этого листочка предполагается, что все числа положительные или 0, если не обговорено обратное. Т.о. с дополнительным кодом заморачиваться не потребуется, разве что для собственного развития посмотреть на содержимое std::bitset от отрицательного значения.

Задачи.

A. *Неполное частное.*
Дано два числа N и k. Выведите неполное частное N и 2^k. Можно использовать только битовые операции.

Input	Output
10 2	2

B. *Остаток.*
Дано два числа N и k. Выведите остаток от деления N и 2^k. Помимо битовых операций можно вычитать 1.

Input	Output
37 3	5

C. *Инвертировать бит.*
Дано два числа N и i.
В числе N инвертируйте i-тый бит. То есть, если он был равен 0, сделайте его равным 1, если он был равен 1, сделайте его равным 0. Логические операции использовать запрещается.

Input	Output
15 2	11

D. *Без умножения умножьте.*
Дано два числа N и M. Посчитайте их произведение, не используя операции умножения, деления, взятия остатка.

Input	Output
2 3	6

E. *Федин шифр.*
Федя решил зашифровать восьмибитное число, переставив все его биты влево по циклу: нулевой бит станет первым, первый — вторым, седьмой — нулевым. Помогите Феде.

Input	Output
129	3

F. *Никитин шифр.*
Посмотрев на Федю из предыдущей задачи, Никита извернулся и придумал более сложный шифр: все биты сдвигаются влево по циклу на несколько позиций (не более чем 7, возможно на 0). Сплагиатил, конечно. Тем не менее - помогите и Никите тоже.

Input	Output
129 3	12

G. *Пашин и Машин шифр.*

Паша и Маша считают, что предыдущие два шифра - для неучей, и потому решают шифровать сразу 32-х битные числа.

Для этого они решают поменять местами четные и нечетные биты. То есть, местами меняются 0-ой и 1-ый биты, 2-ой и 3-ий, и т.д. Использовать арифметические операции запрещается.

Притворитесь Пашей и Машей и напишите программу, шифрующую числа таким образом.

Input	Output
78	141

H. *Единственный и неповторимый.*

Кате говорят много целых чисел (неотрицательных). При этом известно, что все числа, кроме одного, были названы ровно по два раза, а это число только один раз. Необходимо найти это уникальное число. Учтите, что Катя не может запомнить все числа, которые ей говорят (в задаче запрещается использовать массивы).

На первой строке вводится одно нечетное число — количество чисел. Далее на каждой строке ровно по одному числу.

Input	Output
5 2 1 5 2 1	5

I. *Код, исправляющий ошибку.*

В племени Мутумба всего N слов. Секретная служба племени закодировала каждое слово целым числом от 0 до $2^{32} - 1$ (включительно). В службе понимают, что во время передачи данных может возникнуть ошибка в нескольких битах. Поэтому, каждые два числа, участвующие в коде, отличаются не менее чем в трех битах. Такой код называется *исправляющим одну ошибку.* Действительно, если во время передачи информации один бит будет передан ошибочно, принимающая сторона может найти число из кода, отличающееся от полученного ровно в одном бите. Такое число будет единственным.

На первой строчке через пробел вводятся N чисел, которыми закодированы слова племени Мутумба. Гарантируется, что любые два числа различаются не менее, чем в трех битах. На второй строчке число M, которое необходимо расшифровать. То есть, если M отличается от одного из чисел, которым зашифровано слово, не более, чем в одном бите, необходимо вывести число, которое нам хотели передать. Если M отличается от каждого числа не менее, чем в двух битах, выведите Ne ponimat'.

Input	Output
537966573 2845156959 1442095319	Ne ponimat'

Input	Output
1 209 47 43	47