Поразрядные операции в С++

Могут применяться только к данным в формате целого без знака

1. Сдвиг влево двоичного кода (умножение на 2)

формат операции

x<<n

где х - значение, код которого должен быть изменен

n – Количество разрядов, на который должен быть сдвинут код

Пример. Сдвиг влево двоичного кода числа 5 на два разряда.

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned int x=5, n=2;

x=x<<n;

cout<<x; //результат 20

cin.get();

return 0;

}

1. Сдвиг вправо двоичного кода (деление на 2)

формат операции

x>>n

где х - значение, код которого должен быть изменен

n – Количество разрядоd, на который должен быть сдвинут код

Пример. Сдвиг влево двоичного кода числа 20 на два разряда.

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned int x=20, n=2;

x=x>>n;

cout<<x; //результат 5

cin.get();

return 0;

}

1. Поразрядное сложение

Используется для установки в 1 заданного бита

Формат

X|Y

X и Y переменные одной разрядности

Таблица битового сложения

0|1=1

0|0=0

1|0=1

1|1=1

Пример установки в бит с номером 5 двоичного кода х значения 1.

Модель решения

Пусть х=1, то его двоичный код в формате одного байта 00000001

Чтобы установить пятый бит х в 1 к нему надо прибавить число с двоичным кодом 00100000 т.е. десятичное 32.

Представим решение в форме

Х : 00000001

|

32: 00100000

00100001 в десятичной системе это число 33

Реализация примера

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned char x=1;

x=x|32;

cout<<x; //результат 33

cin.get();

return 0;

}

1. Поразрядное умножение

Используется для обнуления заданного бита

Формат

X & Y

Таблица битового сложения

0&1=0

0&0=0

1&0=0

1&1=1

Пример установки в бит с номером 5 двоичного кода х значения 0.

Модель решения

X: 11111111

&

Y: 11011111

X&Y= 11011111

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned char x=255; // 11111111

unsigned char maska =0xEF; //239 =11101111

x=x& maska;

cout<<(int)x; //результат 239

cin.get();

return 0;

}

1. Инвертирование

Замена бита 0 на значение 1 и наоборот.

Применяется при проведении операций, требующих инвертирования разрядв

Формат: ~X

Пример. Создать обратный код целого числа.

Пусть Х=11011011, тогда Х=~Х. Результат: 00100100

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned char x=20, n=2;

x= ~x;

cout<<(int)x; //результат 235

cin.get();

return 0;

}

Пример. Инвертирование маски перед умножением. Установить пятый бит в 0

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned char x=255;

unsigned char maska =0x01; //1=00000001

x=x&(~ (maska<<4));

cout<<(int)x; //результат 239

cin.get();

return 0;

}

Самостоятельно.

1. Х=5. Установить 7 бит в 1.
2. Х=255. Установить 5 и 3 биты в 0.
3. Инвертировать 5 бит переменной х.
4. Вывести все биты значения переменной Х. Размер переменной неизвестен.

Пример. Написать функцию, которая возвращает значение **Х**, где **n** битов, начиная с позиции **р**, установлены так, как n самых правых битов в Y, а остальные биты не изменились.

Решение/

Пусть n=4, р – любая позиция в Х

xxx..,,xnnnnxx…x

yyy…yyyyyyynnnn

Чтобы установить n бит X в значения n последних бит Y, можно:

1. Очистить в Х n бит, начиная с позиции p.
2. Очистить в Y все биты левее n последних и сдвинуть их в позицию p.
3. Сложить результаты п.1 и п.2.

xxx..,,x0000xx…x

000…0nnnn00…0

xxx..,,xnnnnxx…x

Реализация п.1

Чтобы обнулить n бит в Х, надо умножить Х на число, в котором до позиции р будут 1, а с позиции р n нулей, а далее тоже 1.

Для этого:

А) устанавливаем n правых бит в 0: в~0<<n //11111111🡪11110000

Б)заменяем нули на 1, а единицы на нули: ~(~0<<n)

В) сдвигаем n правых единиц в позицию р: ~(~0<<n)<<(p+1-n)

Г) устанавливаем n ,битов в 1, а остальные в 0: ~(~(~0<<n)<<(p+1-n))

Д) очищаем в Х n битов с позиции р в 0: Х& ~(~(~0<<n)<<(p+1-n))

Готовим Y

Очистить все биты в у, кроме последних n: Y & ~(~0<<n)

Переместить последние n битов Y в позицию p: (Y& ~(~0<<n))<<(p+1-n)

Установить в Х нужные биты в значения Y:

(Х& ~(~(~0<<n)<<(p+1-n))) | ((Y& ~(~(~0<<n))<<(p+1-n))

#include "stdafx.h"

#include "iostream"

using namespace std;

unsigned setbit(unsigned int x,int n, int p, unsigned int y)

{

return (x& (~(~(~0<<n)<<(p+1-n))))|(y& ~(~0<<n))<<(p+1-n);

}

void coutp(unsigned int x)

{

int n=sizeof(int)\*8;

unsigned maska=(1<<n-1);

for(int i=1; i<=n;i++)

{

cout<<((x&maska)>>(n-i));

maska=maska>>1;

}

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

unsigned x=0xFFFFFFFF, y=0xFFFFFFFA;

int n=4, p=8;

y=setbit(x,4,10,y);

coutp(y);

//результат 239

cin.get();

return 0;

}