Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 11

По дисциплине «Основы алгоритмизации и программирования»

На тему «Битовые операции»

Выполнил:

Студент 1 курса 10 группы

Сегренёв Кирилл Сергеевич

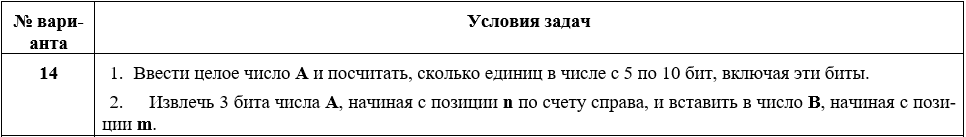
Преподаватель: асс. Андронова М.В.

2023, Минск

**ВАРИАНТ 14**

В соответствии со своим вариантом разработать программы, использующие ***битовые*** операции для решения задач, представленных в таблице.

Результаты одной из программ представить в **Отладчике**.



**Задание 1.**

**Код:**

#include <iostream>

using namespace std;

void main()

{

setlocale(LC\_CTYPE, "Russian");

// Объявляем переменные, инициализируем их

unsigned int A;

int i, DetectOne = 0;

const unsigned int mask = 1 << 15;

const unsigned int maskDetectOne = 0100000;

//Ввод числа с клавиатуры

cout << "Введите целое число ";

cin >> A;

//Запишем число в двоичном виде (чтобы было понятно, правильно ли получилось) с помощью цикла (число будет состоять из 16 символов в двоичном представлении)

cout << "Двоичный вид: ";

for (i = 1; i <= 16; i++)

{

putchar(mask & A ? '1' : '0');

if (i >= 7 && i <= 12) { // Условие для проверки бита на 1

(maskDetectOne & A ? DetectOne++ : DetectOne);

}

A <<= 1; // Сдвигаем число для дальнейших проверок и вывода на экран

// Условие для форматирования текста при выводе на экран

if (i % 4 == 0)

{

putchar(' ');

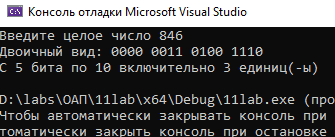
}

}

cout << '\n' << "С 5 бита по 10 включительно " << DetectOne << " единиц(-ы)" << endl; // Вывод на экран количество единиц

}

**Output:**

****

**Задание 2.**

**Код:**

#include <iostream>

using namespace std;

void main() {

setlocale(LC\_ALL, "ru");

// Объявление переменных

int n, m;

unsigned int A, B;

//Ввод переменных

cout << "Введите число A = ";

cin >> A;

cout << "Введите число B = ";

cin >> B;

cout << "Введите позицию извлечения битов из числа A n = ";

cin >> n;

cout << "Позиция вставки в число B m = ";

cin >> m;

// Объявление новых переменных(масок для чисел)

int maskA = 7 << (n - 1);

int maskB = ~maskA >> 2; // Сдвигаем маску для того, чтобы она справа начиналась с 00

// Создаем цикл для маски числа B, чтобы понять, на какую позицию нам нужно вставлять числа

for (int i = 1; i < m; i++) {

maskB = (maskB << 1) + 1;

}

char tmp[33]; // Объявляем массив с символами, который позволяет нам использовать itoa\_s

// Вывод числа А и числа B в двоичном формате и вывод результата действий над числами

\_itoa\_s(A, tmp, 2);

cout << "A: " << tmp << endl;

\_itoa\_s(B, tmp, 2);

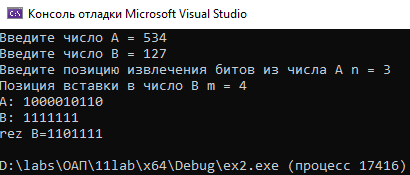
cout << "B: " << tmp << endl;

\_itoa\_s(((B & maskB) | (((A & maskA) >> n - 1) << m - 1)), tmp, 2);

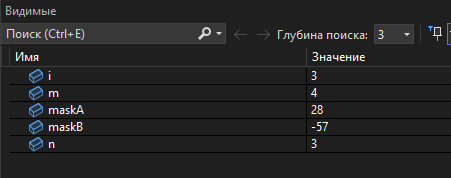
cout << "rez B=" << tmp << endl;

}

**Output:**

****

**Отладчик:**

****

**ДОПОЛНИТЕЛЬНЫЕ ЗАДАНИЯ**

**ВАРИАНТ 2**

****

**Задание 1.**

**Код:**

#include <iostream>

using namespace std;

// 2 вариант

void main(){

setlocale(LC\_ALL, "ru");

// Объявление и инициализирование переменных типа int, объявление массива типа char

char tmp[33];

int A, B, maskA = 31;

int maskB = (~maskA << 2) + 3;

// Ввод чисел

cout << "Первое число А=";

cin >> A;

cout << "Второе число В=";

cin >> B;

// Выведем введенные десятичные числы в двоинчный вид и заменим те биты, которые нужно изменить по условию

// Далее выводим результат после изменения

\_itoa\_s(A, tmp, 2);

cout << "A=" << tmp << endl;

\_itoa\_s(B, tmp, 2);

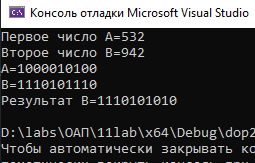
cout << "B=" << tmp << endl;

\_itoa\_s(((B & maskB) | ((A & maskA) >> 1) << 2), tmp, 2);

cout << "Результат B=" << tmp << endl;

}

**Output:**

****

**Задание 2.**

**Код:**

#include <iostream>

using namespace std;

// 2 в

void main() {

// Объявляем переменные типа int и массив типа char

char tmp[33];

int n, p;

unsigned int A, B;

// Ввод чисел

cout << "Введите число A = ";

cin >> A;

cout << "Введите сколько битов хотите изменить";

cin >> n;

cout << "Введите позицию битов, которые хотите изменить";

cin >> p;

// Создаем маску для числа А

int maskA = 1;

maskA = ~maskA;

// Здесь я уверен, что можно и без инверсии разобраться, но данный способ мне более понятен, поэтому я инвертировал maskA и считал с помощью циклов

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

{

maskA = maskA << 1;

}

for (int i = 1; i < p; i++) {

maskA = maskA << 1;

maskA++;

}

// Данный цикл предназначен для того, чтобы взять числа не левее данной позиции, а правее

for (int i = 0; i < n; i++)

{

maskA = maskA >> 1;

}

maskA = ~maskA; // После всех действий инвертируем обратно

// Выведем А в двоичном виде и маску для А, чтобы понять, что меняется

\_itoa\_s(A, tmp, 2);

cout << "A: " << tmp << endl;

\_itoa\_s(maskA, tmp, 2);

cout << "maskA: " << tmp << endl;

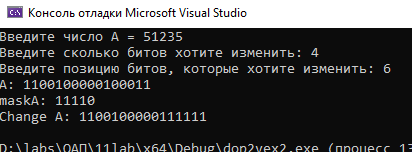
// И в конце выводим результат измененных битов

\_itoa\_s(A | maskA, tmp, 2);

cout << "Change A: " << tmp << endl;

}

**Output:**

****

**ВАРИАНТ 11**



**Задание 1.**

**Код:**

#include <iostream>

using namespace std;

// 11 вариант

void main()

{

setlocale(LC\_CTYPE, "Russian");

// Объявление переменной типа int и массива char

int A;

char tmp[33];

// Ввод числа

cout << "Введите число ";

cin >> A;

// Выведем число в двоичном виде

\_itoa\_s(A, tmp, 2);

cout << "Число в двоичном виде = " << tmp << endl;

if ((A & 15) == 0) // Условие кратности для числа A с дальнейшим выводом на экран

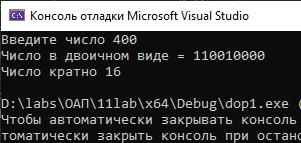
cout << "Число кратно 16" << endl;

else

cout << "Число не кратно 16" << endl;

}

**Output:**

****

**Задание 2.**

**Код:**

#include <iostream>

using namespace std;

// 11 вариант

void main() {

setlocale(LC\_ALL, "rus");

// Объявление переменных типа int, объявление массива типа char

char tmp[33];

int n, m, p, q;

unsigned int A, B;

//Ввод переменных

cout << "Введите число A = ";

cin >> A;

cout << "Введите число B = ";

cin >> B;

cout << "Введите число битов, которые превращаются в 0 в числе А: ";

cin >> n;

cout << "Позиция, где будут данные нули: ";

cin >> p;

cout << "Введите число битов, которые хотите взять из числа B: ";

cin >> m;

cout << "Позиция, с которой вы хотите их взять: ";

cin >> q;

// Инициализирование и создание маски для числа А с помощью массивов

int maskA = 1;

maskA = ~maskA;

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

{

maskA = maskA << 1;

}

for (int i = 1; i < p; i++) {

maskA = maskA << 1;

maskA++;

}

// Инициализирование и создание маски для числа B с помощью массивов

int maskB = 1;

maskB = ~maskB;

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

{

maskB = maskB << 1;

}

for (int i = 1; i < q; i++) {

maskB = maskB << 1;

maskB++;

}

maskB = ~maskB;

// Вывод на экран числа A и числа B в двоичном виде на экран

\_itoa\_s(A, tmp, 2);

cout << "A: " << tmp << endl;

\_itoa\_s(maskA & A, tmp, 2);

cout << "clear A: " << tmp << endl;

\_itoa\_s(B, tmp, 2);

cout << "B: " << tmp << endl;

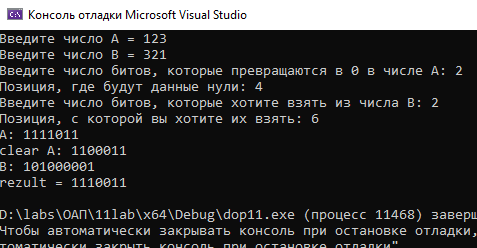
// Вывод результата дейтсвий, сказанный по условию

\_itoa\_s(((A & maskA) | ((B & maskB) >> (q - 1)) << p - 1), tmp, 2);

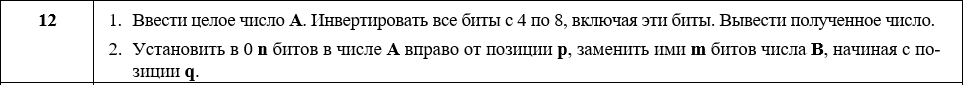
cout << "rezult = " << tmp << endl;

}

**Output:**

****

**ВАРИАНТ 12**

****

**Задание 1.**

**Код:**

#include <iostream>

using namespace std;

// 12 вариант

void main() {

setlocale(LC\_CTYPE, "Russian");

// Объявляем переменну типа int и массив char

char tmp[33];

unsigned int A;

//Ввод числа с клавиатуры

cout << "Введите число А = ";

cin >> A;

int maskA = 31 << 3; // Создание маски для инвертирования всех битов начиная с 4 заканчивая 8 включительно

// Выводим на экран маску для А, само А и резуьтат выислений

\_itoa\_s(A, tmp, 2);

cout << "A: " << tmp << endl;

\_itoa\_s(maskA, tmp, 2);

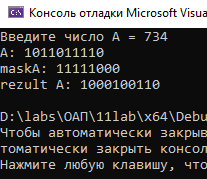
cout << "maskA: " << tmp << endl;

\_itoa\_s((A ^ maskA), tmp, 2); // Здесь происходит инвертирование битов с 4 по 8

cout << "rezult A: " << tmp << endl;

}

**Output:**

****

**Задание 2.**

**Код:**

#include <iostream>

using namespace std;

// 12 вариант

void main() {

setlocale(LC\_ALL, "rus");

// Объявление переменных типа int, объявление массива типа char

char tmp[33];

int n, m, p, q;

unsigned int A, B;

//Ввод переменных

cout << "Введите число A = ";

cin >> A;

cout << "Введите число B = ";

cin >> B;

cout << "Введите число битов, которые превращаются в 0 в числе А: ";

cin >> n;

cout << "Позиция, где будут данные нули: ";

cin >> p;

cout << "Введите число битов, которые хотите взять из числа B: ";

cin >> m;

cout << "Позиция, с которой вы хотите их взять: ";

cin >> q;

//Инициализируем маску для числа B и дальше, с помощью циклов, делаем нам нужную маску под каждый случай

int maskA = 1;

maskA = ~maskA;

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

{

maskA = maskA << 1;

}

for (int i = 1; i < p; i++) {

maskA = maskA << 1;

maskA++;

}

for (int i = 0; i < n; i++) // Это условие обязательно, т.к нам сказано, что числа должны быть справа от указаной позиции

{

maskA = maskA >> 1;

}

//Инициализируем маску для числа B и дальше, с помощью циклов, делаем нам нужную маску под каждый случай

int maskB = 1;

maskB = ~maskB;

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

{

maskB = maskB << 1;

}

for (int i = 1; i < q; i++) {

maskB = maskB << 1;

maskB++;

}

maskB = ~maskB;

// Вывод на экран чисел в двоимном представлении, масок, очищенных битов и изъятых битов

\_itoa\_s(A, tmp, 2);

cout << "A: " << tmp << endl;

\_itoa\_s(maskA, tmp, 2);

cout << "maskA: " << tmp << endl;

\_itoa\_s((A & maskA), tmp, 2);

cout << "clear A: " << tmp << endl;

\_itoa\_s(B, tmp, 2);

cout << "B: " << tmp << endl;

\_itoa\_s(maskB, tmp, 2);

cout << "mask B: " << tmp << endl;

\_itoa\_s((B & maskB) >> q - 1, tmp, 2);

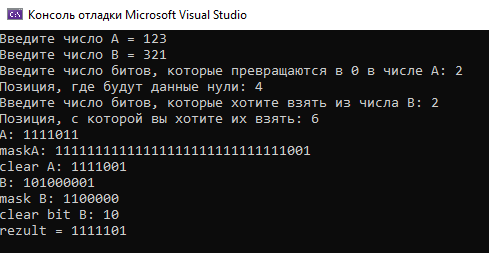
cout << "clear bit B: " << tmp << endl;

\_itoa\_s(((A & maskA) | (((B & maskB) >> (q - 1))) << (p - n) - 1), tmp, 2);

cout << "rezult = " << tmp << endl;

}

**Output:**

****