#include <iostream>

#include <math.h>

using namespace std;

void PrintBin(unsigned char Val) {

int b;

for(b = 7; b >= 0; b--)

{

if (Val & (1 << b)){

putchar('1');

}

else{

putchar('0');

}

}

}

void first\_task\_array (int start\_number=0, int max\_number=20, int step=1, bool empty\_line=false, bool doubling=false){

int steps = (max\_number - start\_number - 1) / step;

printf("Цикл 'for': %d шагов увеличения переменной:\n", steps);

for(int i = start\_number; i < max\_number; i = i + step){

printf("i = %d, двоичное = ",i);

PrintBin(i);

printf("\n");

if (doubling){

i \*= 2;

}

if (empty\_line){

printf("\n");

}

}

}

void first\_task\_reverse (int start\_number=0, int max\_number=20, int step=1){

int steps = (max\_number - start\_number + 1) / step;

printf("Цикл 'for': %d шагов увеличения переменной:\n", steps);

for(int i = start\_number; i > max\_number; i = i + step){

printf("i = %d, двоичное = ",i);

PrintBin(i);

printf("\n");

}

}

void first\_task ()

{

printf("\nПервое задание:\n");

// first\_task\_array(0, 40);

// first\_task\_array(0, 256);

// first\_task\_array(0, 61, 2);

// first\_task\_array(0, 91, 3);

// first\_task\_array(128, 149);

// first\_task\_array(0, 21, 1, true);

// first\_task\_reverse(255, 234, -1);

// first\_task\_array(1, 129, 0, false, true);

}

void second\_task\_array(bool left=true, unsigned char new\_k=1, int position=1){

printf("8 битовых сдвигов переменной:\n");

unsigned char k = new\_k;

for(int i = 0; i < 8; i++){

printf("%d,\t двоичное= ", k);

PrintBin(k);

printf("\n");

if (left){

k <<= position;

}

else{

k >>= position;

}

}

}

void second\_task\_part\_tree(){

int L[4] = {1, 3, 4, 5};

for (int i = 0; i < 4; i++){

second\_task\_array(true, L[i], 1);

}

}

void second\_task(){

printf("\nВторое задание:\n");

// second\_task\_array(false, 0x80);

// second\_task\_array(true, 1, 2);

// second\_task\_part\_tree()

}

int prepared\_a\_for\_third\_task(){

printf("Установка и сброс отдельных битов:\n");

unsigned char a = 144;

printf("Исходное число: \t%d,\t", a);

PrintBin(a);

printf("\n");

unsigned char b = 6;

unsigned char c = 3;

a |= (1 << b);

a |= (1 << c);

printf("Установка битов %d и %d:\t%d,\t",b, c, a);

PrintBin(a);

printf("\n");

unsigned char d = 4;

unsigned char e = 1;

a &= ~((1 << d) | (1 << e));

printf("Сброс битов %d и %d:\t%d,\t", d, e, a);

PrintBin(a);

printf("\n");

unsigned char f = 6;

unsigned char g = 1;

a^= ((1 << f) | (1 << g));

printf("Инвертирование битов %d и %d:\t%d,\t", f, g, a);

PrintBin(a);

printf("\n");

return a;

}

int third\_task\_part\_one(int a, int L, int M , int N){

printf("Первая часть: ");

a |= (1 << L); // adding bit on L position

a &= ~(1 << M); // deleting bit from M position

a^= (1 << N); // inverting bit from N position

PrintBin(a);

printf(", десятичная %d\n", a);

return a;

}

int third\_task\_part\_two(int a, int X, int Y , int Z){

printf("Вторая часть: ");

a <<= X; // shifting bit on X position left

a^= (1 << Y); // inverting bit from Y position

a >>= Z; // shifting bit on Z position right

PrintBin(a);

printf(", десятичная %d\n", a);

return a;

}

int third\_task\_part\_tree(int a, int O, int P){

printf("Третья часть: ");

if ((a >> O) & 1u){ // if bit == 1

a &= ~(1 << P); // del bit P

}

PrintBin(a);

printf(", десятичная %d\n", a);

return a;

}

int third\_task\_part\_four(int a, int R, int S){

printf("Четвертая часть: ");

if (!((a >> R) & 1u)){ // if bit != 1

a |= (1 << S); // setting bit S

}

PrintBin(a);

printf(", десятичная %d\n", a);

return a;

}

void third\_task(){

printf("\nТретье задание:\n");

int a = prepared\_a\_for\_third\_task();

unsigned char L = 6, M = 5, N = 1, X = 2, Y = 3, Z = 0, O = 3, P = 0, R = 6, S = 7;

a = third\_task\_part\_one(a, L, M, N);

a = third\_task\_part\_two(a, X, Y, Z);

a = third\_task\_part\_tree(a, O, P);

a = third\_task\_part\_four(a, R, S);

}

int main(int argc, char\* argv[])

{

// first\_task();

// second\_task();

third\_task();

}