Лабораторно упражнение N 4
Побитови операции

I. Теоретична обосновка

В C и C++ е предвидена възможност за реализиране на операции с отделните битове на операндите чрез т.нар. **поразредни** (**побитови**) оператори. Побитовите оператори могат да се прилагат само върху данни от тип ***int*** и неговите модификации ***short, long, unsigned***.

1. Преместване наляво

Поразрядното преместване наляво с определен брой позиции се осъществява чрез оператор **<<.** Освободените разряди се допълват с нули (0).

Примери:

```
x=0x4f;          // x=0100 1111b
x=x<<2;          // резултатът е: x = 1 0011 1100b =  0x13c

                 // всяко преместване с 1 разряд наляво е
y=1;             // равносилно на умножение по 2
y=y<<1;          // y=010b=2
y=y<<1;          // y=100b=4
```

2. Преместване надясно

Поразрядното преместване надясно с определен брой позиции се осъществява чрез оператор **>>.** Освободените разряди се допълват с нули (0).

Примери:

```
x=0xf8;          // x=1111 1000b
x=x>>2;                  // резултатът е: x = 0011 1110b =  0x3e

                 // всяко преместване с 1 разряд надясно е
y=4;             // равносилно на целочислено деление на 2
y=y>>1;                  // y=010b=2
y=y>>1;          // y=001b=1
```

3. Поразрядно И

Оператор поразрядно И е **&**. Резултатът от изпълнението на оператора е представен чрез таблица 1, като x и y са отделни битове.

Таблица 12.1

x	y	x&y
0	0	0
0	1	0
1	0	0
1	1	1

Побитов оператор И се използва при нулиране на определени битове. Например, нека да бъде нулиран 5 бит на даден 16 разряден регистър.

Стойността (маската), с която ще се нулира бит 5 от 16-разрядният регистър е: 1101 1111 (0xdf).

```
mask=0xdf;          // mask= 1101 1111b
reg=0xf8;           // reg=1111 1000b
reg=reg & mask;     // резултатът е reg = 1101 1000b (0xd8)
```

4. Поразрядно ИЛИ

Оператор поразрядно ИЛИ е | . Резултатът от действието е представен чрез таблица 2, като x и y са отделни битове.

Таблица 12.2

x	y	x|y
0	0	0
0	1	0
1	0	0
1	1	1

Побитова оператор ИЛИ се използва при установяване в 1 на определени битове. Например, нека да бъде установен в 1, бит 5 на даден 16 разряден регистър. Маската е: 0010 0000 (0x20).

```
mask=0x20;          // mask=0010 0000b
reg=0x88;           // reg=1000 1000b
reg=reg | mask;             // резултатът е reg = 1010 1000 (0xa8)
```

5. Поразрядно изключващо ИЛИ

Оператор поразрядно изключващо ИЛИ е ^ . Резултатът от изпълнението на оператора е представен чрез таблица 3, като x и y са отделни битове.

Таблица 12.3

x	y	x^y
0	0	0
0	1	1
1	0	1
1	1	0

Побитов оператор изключващо ИЛИ се използва когато е необходимо да бъдат инвертирани определи битове. Например, нека да бъде инвертиран бит 5 на даден 16 разряден регистър. Маската е: 0010 0000 (0x20).

```
mask=0x20;          // mask=0010 0000b
reg=0x88;                   // reg=1000 1000b
reg=reg ^ mask;             // резултатът е reg = 1010 1000 (0xa8)
reg=reg ^ mask;             // резултатът е reg = 1000 1000 (0x88)
```

6. Инвертиране на битове

Оператор инвертиране на битовете на операнда е ~. Резултатът е представен чрез таблица 4.

Таблица 12.4

x	~x
0	1
1	0

Пример:

```
reg=0x88;          // reg=1000 1000b
reg1=~reg;         // резултатът е reg1 = 0111 0111b=0x77
```

IV. Контролни въпроси

1. Какво ще се изведе на екрана след изпълнението на следния програмен код?

```
int i=5;
int j=10;
printf("\n%d %d %d",i&j,i|j, i^j);
```

2. Какво ще се изведе на екрана след изпълнението на следния програмен код?

```
int i=Ox5;
int j=Ox10;
printf("\n%d %d %d",i&j,i|j, i^j);
```

II. Задачи за изпълнение

1. Да се създаде конзолно приложение, чрез което се демонстрира действието на побитовите оператори << и >> като се въведе определено число и се преместят битовете определен брой пъти наляво и надясно. Да се използва извеждане на изходния резултат като десетично число със знак, десетично число без знак и шестнадесетично число.

2. Да се създаде конзолно приложение, чрез което се демонстрира действието на побитовите оператори &, |, ^, ~ като се въведат един или два операнда. Да се използва извеждане на изходния резултат като десетично число със знак, десетично число без знак и шестнадесетично число.

3. Да се създаде конзолно приложение, чрез което да се въведе стойност и да се нулира определен бит от числото; да се установи в 1 определен бит и да се промени стойността на определен бит (от 1 в 0 и обратно).

Упътване:
Да се използват побитовите оператори: << за да се премести 1 в определената позиция; | за установяване на бит в единица; ^ за инвертитане на определен бит; ~ за инвертиране на всички битове; & за нулиране на бит.