

位运算

1. 概述

1.1. 字节和位

字节: byte, 计算机中数据表示的**基本**单位

位 : bit, 计算机中数据表示的**最小**单位

1 byte = 8 bit

1.2. 位运算

以bit为单位进行数据的运算

1.3. 位运算的基本方法

★ 按位进行 (只有0、1)

★ 要求运算数据长度相等, 若不等, 则右对齐, 按最高位补齐左边

char a=0x37; 0000 0000 0011 0111

short b=0x1234; 0001 0010 0011 0100

char a=0xA7; 1111 1111 1010 0111

short b=0x8341; 1000 0011 0100 0001

★ 数在计算机内是用补码表示的

位运算

2. 常用的位运算

2.1. 与(&)

运算规则：遇0得0

例：char a=3, b=5; 求a&b

```
0000 0011
& 0000 0101
0000 0001    a&b=1
```

例：char a=3; short b=5; 求a&b

```
0000 0000 0000 0011
& 0000 0000 0000 0101
0000 0000 0000 0001    a&b=1
```

例：char a=0xb6, b=0xc2; 则a&b

```
1011 0110
& 1100 0010
1000 0010    a&b=0x82
```

有符号十进制：-126

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a1=3, b1=5;
    cout << "a=" << (int)a1 << " b=" << (int)b1
         << " a&b=" << (a1&b1) << endl;

    char a2=3;
    short b2=5;
    cout << "a=" << (int)a2 << " b=" << b2
         << " a&b=" << (a2&b2) << endl;

    char a3=0xb6, b3=0xc2;
    cout << "a=" << hex << (int)a3 << " b=" << (int)b3;
    cout << " a&b=0x" << hex << (a3&b3) << " "
         << dec << (a3&b3) << endl;
}
```

```
a=3 b=5 a&b=1
a=3 b=5 a&b=1
a=ffffffb6 b=ffffffc2 a&b=0xffffff82 -126
```

位运算

2. 常用的位运算

2.1. 与(&)

运算规则：遇0得0

应用：

★ 清零

例：char a=0xb6;现要求将该数清零，则：

1011 0110			
<u>& 0?00 ?00?</u> 要清零数为1的位，本数对应位为0			
0000 0000			
a&0x0	a&0x1	a&0x8	a&0x9
a&0x40	a&0x41	a&0x48	a&0x49

★ 取指定位

例：char a=0xb6;现要求只保留低4位，而高4位清0，则：

1011 0110	
<u>& 0000 1111</u> 要保留的位，本数对应位为1	
0000 0110	

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    /* &的应用：清0 */
    char a1=0xb6;
    cout << "char a=" << hex << (int)a1 << endl
         << "    a&0x0 =" << dec << (a1&0x0) << endl
         << "    a&0x1 =" << dec << (a1&0x1) << endl
         << "    a&0x8 =" << dec << (a1&0x8) << endl
         << "    a&0x9 =" << dec << (a1&0x9) << endl
         << "    a&0x40=" << dec << (a1&0x40) << endl
         << "    a&0x41=" << dec << (a1&0x41) << endl
         << "    a&0x48=" << dec << (a1&0x48) << endl
         << "    a&0x49=" << dec << (a1&0x49) << endl;

    /* &的应用：取指定位 */
    char a2=0xb6;
    cout << "char a=0x" << hex << (int)a2
         << " a&0x0F=" << dec << (a2&0x0F) << endl;
}
```

```
char a=ffffffb6
a&0x0 =0
a&0x1 =0
a&0x8 =0
a&0x9 =0
a&0x40=0
a&0x41=0
a&0x48=0
a&0x49=0
char a=0xffffffb6 a&0x0F=6
```

位运算

2. 常用的位运算

2.2. 或(|)

运算规则：遇1得1

例：char a=3, b=5; 求a|b

```
0000 0011
| 0000 0101
```

0000 0111 a|b=7

例：char a=3; short b=5; 求a|b

```
0000 0000 0000 0011
| 0000 0000 0000 0101
```

0000 0000 0000 0111 a|b=7

例：char a=0xb6, b=0xc2; 则a|b

```
1011 0110
| 1100 0010
```

1111 0110 a|b=0xF6

有符号10进制：-10

应用：★ 设定某些位为1

例：char a=0xb6; 要求1, 4位设为1, 其它不变

```
1011 0110
| 0000 1001
```

1011 1111 (0xBF)

要设置的位，本数对应位为1

(0xBF)

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a1=3, b1=5;
    cout << "a=" << (int)a1 << " b=" << (int)b1
         << " a|b=" << (a1|b1) << endl;

    char a2=3;
    short b2=5;
    cout << "a=" << (int)a2 << " b=" << b2
         << " a|b=" << (a2|b2) << endl;

    char a3=0xb6, b3=0xc2;
    cout << "a=" << hex << (int)a3 << " b=" << (int)b3;
    cout << " a|b=0x" << hex << (a3|b3) << " "
         << dec << (a3|b3) << endl;

    /* |的应用，将1、4 bit位设为1，其它不变 */
    char a4=0xb6;
    cout << "a=" << hex << (int)a4
         << " a|0x9=0x" << (a4|0x9) << endl;
}
```

```
a=3 b=5 a|b=7
a=3 b=5 a|b=7
a=ffffffb6 b=ffffffc2 a|b=0xfffffffff6 -10
a=ffffffb6 a|0x9=0xffffffffbf
```

位运算

2. 常用的位运算

2.3. 异或(^)

运算规则：相同为0，不同为1

例：char a=3, b=5; 求a^b

```
0000 0011
^ 0000 0101
-----
0000 0110    a^b=6
```

例：char a=3; short b=5; 求a^b

```
0000 0000 0000 0011
^ 0000 0000 0000 0101
-----
0000 0000 0000 0110    a^b=6
```

例：char a=0xb6, b=0xc2; 求a^b

```
1011 0110
^ 1100 0010
-----
0111 0100    a^b=0x74
```

有符号10进制：116

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a1=3, b1=5;
    cout << "a=" << (int)a1 << " b=" << (int)b1
          << " a^b=" << (a1^b1) << endl;

    char a2=3;
    short b2=5;
    cout << "a=" << (int)a2 << " b=" << b2
          << " a^b=" << (a2^b2) << endl;

    char a3=0xb6, b3=0xc2;
    cout << "a=" << hex << (int)a3 << " b=" << (int)b3;
    cout << " a^b=0x" << hex << (a3^b3) << " "
          << dec << (a3^b3) << endl;
}
```

a=3 b=5 a^b=6

a=3 b=5 a^b=6

a=fffffb6 b=fffffc2 a^b=0x74 116

位运算

2. 常用的位运算

2.3. 异或(^)

运算规则：相同为0，不同为1

应用：

★ 特定位翻转（0，1互换）

例：char a=0xb6；高4位翻转，低4位不变

1011 0110
^ 1111 0000 要翻转的位，本数对应位为1
0100 0110

★ 两数交换

例：char a=0xb6, b=0xc2；要求a, b互换

三步：a=a^b b=b^a a=a^b

(1) a=1011 0110
b=1100 0010
a=0111 0100 a=a^b=0x74

(2) b=1100 0010
a=0111 0100
b=1011 0110 b=b^a=0xb6

(3) a=0111 0100
b=1011 0110
a=1100 0010 a=a^b=0xc2

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    /* ^的应用：特定位翻转 */
    char a1=0xb6;
    cout << "a=" << hex << (int)a1
         << " a^0xF0=0x" << (a1^(char)0xF0) << endl;

    /* ^的应用：两数交换 */
    char a2=0xb6, b2=0xc2;
    cout << "a=" << hex << (int)a2 << " b=" << (int)b2 << endl;
    a2 = a2^b2;
    b2 = b2^a2;
    a2 = a2^b2;
    cout << "a=" << hex << (int)a2 << " b=" << (int)b2 << endl;
}
```

a=fffffffb6 a^0xF0=0x46
a=fffffffb6 b=fffffffc2
a=fffffffc2 b=fffffffb6

```
int main()
{
    char *info="This is my student";
    char *sec ="周伯通黄药师郭靖黄蓉";
    char dst[80];
    char *p1 = info, *p2 = sec, *p3 = dst;

    /* 加密 */
    for (;p1; p1++, p2++, p3++)
        *p3 = *p1 ^ *p2;
    *p3 = 0;

    /* 打印加密串 */
    cout << dst << endl;

    /* 解密(info没有被传送过来，不允许使用) */
    for (p3 = dst, p2 = sec; *p3; p3++, p2++)
        *p3 = *p2 ^ *p3;

    cout << dst << endl;
}
```

位运算

2. 常用的位运算

2.4. 取反(~)

运算规则：0/1互反

例：char a=0x5c; 求~a

a=0101 1100

~a=1010 0011

~a=0xa3

有符号10进制：-93

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a=0x5c;
    cout << "a=" << hex << (int)a
         << " ~a=0x" << (~a) << " "
         << dec << (~a) << endl;

    return 0;
}
```

a=5c ~a=0xffffffffa3 -93

位运算

2. 常用的位运算

2.5. 左移(<<)

运算规则：左移数据，右补0

例：char a=0x12;

a=0001 0010

0010 0100 a<<1=0x24

0100 1000 a<<2=0x48

1001 0000 a<<3=0x90

0x12 = 18

0x24 = 36

0x48 = 72

0x90 = -112

无符号:144

例：int b=0x12;

a<<1=0x24

a<<2=0x48

a<<3=0x90

0x24 = 36

0x48 = 72

0x90 = 144

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
```

为什么是(int)(char)(a<<1)?
先 a<<1
转为 char, 此时若有溢出, 则会丢弃
再转 int, 以int方式输出

```
{
    char a=0x12;
    cout <<"a=0x"<<hex<< int(a)<<" "<<dec<<int(a)<< endl;
    cout << "a<<1=0x" << hex << (int)(char)(a<<1) << " "
        << dec << (int)(char)(a<<1) << endl;
    cout << "a<<2=0x" << hex << (int)(char)(a<<2) << " "
        << dec << (int)(char)(a<<2) << endl;
    cout << "a<<3=0x" << hex << (int)(char)(a<<3) << " "
        << dec << (int)(char)(a<<3) << endl;
}
```

a=0x12 18
a<<1=0x24 36
a<<2=0x48 72
a<<3=0xffffffff90 -112

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
```

```
{
    /* 直接是int型的情况 */
    int b=0x12;
    cout << "a<<1=0x" << hex << (b<<1) << " "
        << dec << (b<<1) << endl;
    cout << "a<<2=0x" << hex << (b<<2) << " "
        << dec << (b<<2) << endl;
    cout << "a<<3=0x" << hex << (b<<3) << " "
        << dec << (b<<3) << endl;
}
```

a<<1=0x24 36
a<<2=0x48 72
a<<3=0x90 144

位运算

2. 常用的位运算

2.5. 左移(<<)

运算规则：左移数据，右补0

例：char a=0x12; 求a<<3

a=0001 0010

1001 0000 a<<3=0x90 有符号 -112

无符号144

★ 在不溢出(1不被舍去)的情况下，左移n位等于乘2的n次方(当做无符号数理解)

例：char a=0x12; 求a<<4

a=0001 0010

1 0010 0000 a<<4=0x20 0x12=18 0x20=32
 $32+256(2^8)=288=18*16(2^4)$

例：char a=0x9c; 求a<<2

a=1001 1100

10 0111 0000 a<<2=0x70 0x9c=156 0x70=112
 $112+512(2^9)=624=156*4(2^2)$

例：char a=0xc2; 求a<<2

a=1100 0010

11 0000 1000 a<<2=0x8 0xc2=194 0x8=8
 $8+512(2^9)+256(2^8)=776=194*4(2^2)$

```
#include <iostream>
```

```
#include <iomanip>
```

```
using namespace std;
```

```
int main()
```

```
{ char a1=0x12;
```

```
cout << "a<<4=0x" << hex << (int)(char)(a1<<4) << " " << dec << (int)(char)(a1<<4) << endl;
```

```
char a2=0x9c;
```

```
cout << "a<<2=0x" << hex << (int)(char)(a2<<2) << " " << dec << (int)(char)(a2<<2) << endl;
```

```
char a3=0xc2;
```

```
cout << "a<<2=0x" << hex << (int)(char)(a3<<2) << " " << dec << (int)(char)(a3<<2) << endl;
```

```
}
```

a<<4=0x20 32
a<<2=0x70 112
a<<2=0x8 8

位运算

2. 常用的位运算

2.6. 右移(>>)

运算规则：右移数据，左补0（逻辑右移）

右移数据，左补符号位（算术右移） - C/C++的位运算时算术右移

例：char a=0x18;

a=0001 1000

0000 1100 a>>1=0xc

0000 0110 a>>2=0x6

0000 0011 a>>3=0x3

0000 0001 a>>4=0x1

0x18 = 24

0xc = 12

0x6 = 6

0x3 = 3

0x1 = 1

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a=0x18;
    cout << "a>>1=0x" << hex << (int)(a>>1) << " "
         << dec << (int)(a>>1) << endl;
    cout << "a>>2=0x" << hex << (int)(a>>2) << " "
         << dec << (int)(a>>2) << endl;
    cout << "a>>3=0x" << hex << (int)(a>>3) << " "
         << dec << (int)(a>>3) << endl;
    cout << "a>>4=0x" << hex << (int)(a>>4) << " "
         << dec << (int)(a>>4) << endl;
}
```

a>>1=0xc 12
a>>2=0x6 6
a>>3=0x3 3
a>>4=0x1 1

位运算

2. 常用的位运算

2.6. 右移(>>)

运算规则：右移数据，左补0（逻辑右移）

右移数据，左补符号位（算术右移） - C/C++的位运算时算术右移

★ 在不溢出(1不被舍去)的情况下，右移n位
等于除2的n次方(当作有符号数理解)

例：char a=0x84; 求a>>1

a=1000 0100

1100 0010 a>>1=0xc2

0x84 = -124

无符号: 132

0xc2 = -62

无符号: 194

最高位为1，若作为符号位，则表示负数

a=1000 0100
-) 1
1000 0011
0111 1100

补码 => 原码

(1) 减1

(2) 取反

(3) 绝对值

|a|=124

|a>>1|=62

a=1100 0010
-) 1
1100 0001
0011 1110

```
#include <iostream>
#include <iomanip>
using namespace std;
```

a=0xffffffff84 -124
a>>1=0xffffffc2 -62

```
int main()
{
    char a=0x84;
    cout << "a=0x" << hex << int(a) << " "
         << dec << int(a) << endl;
    cout << "a>>1=0x" << hex << (int)(a>>1) << " "
         << dec << (int)(a>>1) << endl;
}
```

```
#include <iostream>
#include <iomanip>
using namespace std;
```

a=0x84 132
a>>1=0x42 66
无符号数补0 !!!

```
int main()
{
    unsigned char a=0x84;
    cout << "a=0x" << hex << int(a) << " "
         << dec << int(a) << endl;
    cout << "a>>1=0x" << hex << (int)(a>>1) << " "
         << dec << (int)(a>>1) << endl;
}
```

位运算

2. 常用的位运算

2.6. 右移(>>)

例: char a=0x18;

a=0001 1000	(24)
0000 1100 a>>1=0xc	(12)
0000 0110 a>>2=0x6	(6)
0000 0011 a>>3=0x3	(3)
0000 0001 a>>4=0x1	(1) 溢出舍去了1
0000 0000 a>>5=0x0	(0) 再次溢出舍去1
0000 0000 a>>6=0x0	(0) >>6以上都是0

例: char a=0x84;

a=1000 0100	(-124)
1100 0010 a>>1=0xc2	(-62)
1110 0001 a>>2=0xe1	(-31)
1111 0000 a>>3=0xf0	(-16) 溢出舍去了1
1111 1000 a>>4=0xf8	(-8)
1111 1100 a>>5=0xfc	(-4)
1111 1110 a>>6=0xfe	(-2)
1111 1111 a>>7=0xff	(-1)
1111 1111 a>>8=0xff	(-1) >>8以上都是-1

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a=0x18;
    int i;

    for(i=1; i<=6; i++) {
        a >>= 1;
        cout << "a>>" << i
              << "=0x" << hex << int(a) << " "
              << dec << int(a) << endl;
    }
}
```

a>>1=0xc 12
a>>2=0x6 6
a>>3=0x3 3
a>>4=0x1 1
a>>5=0x0 0
a>>6=0x0 0

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    char a=0x84;
    int i;
    for(i=1; i<=8; i++) {
        a >>= 1;
        cout << "a>>" << i
              << "=0x" << hex << int(a) << " "
              << dec << int(a) << endl;
    }
}
```

a>>1=0xffffffc2 -62
a>>2=0xffffffe1 -31
a>>3=0xfffffff0 -16
a>>4=0xfffffff8 -8
a>>5=0xfffffff4 -4
a>>6=0xffffffe -2
a>>7=0xfffffff -1
a>>8=0xfffffff -1

位运算

2. 常用的位运算

2.7. 复合位运算符

&= |= ^= <<= >>=

★ 将例中 `a = a>>1;` 改为 `a >>= 1;` 结果相同

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{   char a=0x18;
    int i;

    for(i=1; i<=6; i++) {
        a >>= 1;
        cout << "a>>" << i
             << "=0x" << hex << int(a) << " "
             << dec << int(a) << endl;
    }
}
```

```
a>>1=0xc 12
a>>2=0x6 6
a>>3=0x3 3
a>>4=0x1 1
a>>5=0x0 0
a>>6=0x0 0
```

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{   char a=0x84;
    int i;
    for(i=1; i<=8; i++) {
        a >>= 1;
        cout << "a>>" << i
             << "=0x" << hex << int(a) << " "
             << dec << int(a) << endl;
    }
}
```

```
a>>1=0xffffffffc2 -62
a>>2=0xffffffe1 -31
a>>3=0xfffffff0 -16
a>>4=0xfffffff8 -8
a>>5=0xfffffff4 -4
a>>6=0xffffffe -2
a>>7=0xfffffff -1
a>>8=0xffffffff -1
```