**BİTSEL İŞLEMLER**



Bitsel işlemler sayesinde bir verinin veya değişkenin bir bitine veya bir gurup bitine erişme ve işlem yapma imkanı sunmaktadır. C dilinde gerçel(float,double) sayılar dışındaki bütün tipler mantıksal değişkenler olarak kullanılabilmektedir. Bitesel operatörler;



|  |  |  |
| --- | --- | --- |
| Operatör |  |  |
| & | a & b | ve |
| | | a | b | veya |
| ^ | ^a | özel veya |
| << | a << p | Sola kaydırma |
| >> | a >> p | Sağa kaydırma |



|  |  |  |
| --- | --- | --- |
| x = x << y | yerine | x <<= y |
| x = x >> y | yerine | x >>= y |
| x = x & y | yerine | x &= y |
| x = x ^ y | yerine | x ^= y |
| x = x | y | yerine | x |= y |



**Bitsel Değil Operatörü**



~ tümelem işlemi yapar 0 olan bitleri 1, 1 olan bitleri 0 olarak değiştirir.



#include <stdio.h>

#include <conio.h>



int main()

{



unsigned short x = 0x1AC3; /\* x = 0001 1010 1100 0011 \*/



unsigned short y;



y = ~x; /\* y = 1110 0101 0011 1100 \*/

printf("y = %x\n", x);



printf("y = %x , %u\n", y,y); /\* y = 0xE53C \*/

printf("%d",sizeof(short));



getch();

return 0;

}

**Bitsel Ve Operatörü (bitwise and)**



|  |  |  |
| --- | --- | --- |
| x | y | x & y |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Ve işleminde 1 etkisiz eleman, 0 ise yutan elemandır. Bu nedenle herhangi bir sayının istenilen biti ve operatörü kullanılarak 0 yapılabilir.



#include <stdio.h>

#include <conio.h>

int main()

{

unsigned short x = 0x1BC5; /\* x = 0001 1011 1100 0101 \*/



unsigned short y = 0x3A0D; /\* y = 0011 1010 0000 1101 \*/



unsigned short z;



z = x & y; /\* z = 0001 1010 0000 0101 \*/



printf("z = %x , %u", z,z); /\* z = 0x1A05 \*/



getch();



return 0;

}

**Bitsel Veya Operatörü (bitwise or operator)**



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

Ve işleminde 0 etkisiz eleman, 1 ise yutan elemandır. Bu nedenle herhangi bir sayının istenilen biti veya operatörü kullanılarak 1 yapılabilir.

#include <stdio.h>

#include <conio.h>

int main()

{

unsigned short x = 0x1BC5; /\* x = 0001 1011 1100 0101 \*/

unsigned short y = 0X3A0D; /\* y = 0011 1010 0000 1101 \*/



unsigned short z;



z = x | y; /\* z = 0011 1011 1100 1101 \*/

printf("z = %x , %d", z); /\* z = 0x3BCD \*/

getch();

return 0;

}

**Bitsel Özel Veya Operatörü (bitwise exor)**



|  |  |  |
| --- | --- | --- |
| x | y | x ^ y |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

#include <stdio.h>

#include <conio.h>

int main()

{

unsigned short x = 0x1BC5; /\* x = 0001 1011 1100 0101 \*/



unsigned short y = 0X3A0D; /\* y = 0011 1010 0000 1101 \*/



unsigned short z;



z = x ^ y; /\* z = 0010 0001 1100 1000 \*/

printf("z = %x , %u", z , z); /\* z = 0x21C8 \*/

getch();

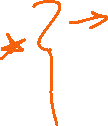


return 0;

}



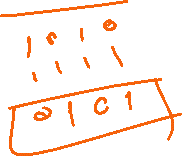
Bir değerin iki defa aynı değer özel veya işlemine sokulması kendisini verir.



#include <stdio.h>



#include <conio.h>



int main()

{

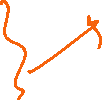
unsigned short x = 0x1BC5; /\* x = 0001 1011 1100 0101 \*/



unsigned short y = 0X3A0D; /\* y = 0011 1010 0000 1101 \*/



unsigned short z;



z = x ^ y;

z = z ^ y; /\* z = 0010 0001 1100 1000 \*/



printf("x= %u , y=%u z = %x , %u", x, y , z , z); /\* z = 0x1BC5 \*/

getch();

return 0;

}



#include <stdio.h>

#include <conio.h>



int main()

{

unsigned short x = 0x1BC5; /\* x = 0001 1011 1100 0101 \*/



unsigned short y = 0X3A0D; /\* y = 0011 1010 0000 1101 \*/

unsigned short z;



printf("x= %u , y=%u\n", x, y);



x = x ^ y;



y = x ^ y;



x = x ^ y; /\* z = 0010 0001 1100 1000 \*/



/\*x ^= y ^= x^= y;\*/



printf("x= %u , y=%u", x, y); /\* z = 0x1BC5 \*/

getch();



return 0;

}



**Bitsel Kaydırma Operatörleri (bitwise shift operators)**



Bitsel sağa kaydırma operatörü >> (bitwise right shift)

Bitsel sola kaydırma operatörü << (bitwise left shift)



#include <stdio.h>

#include <conio.h>



int main()

{



unsigned short x = 52; /\* x = 0000 0000 0011 0100 \*/



unsigned short y = 2;



unsigned short z;



printf("z = %d\n", x);



z = x << y; /\* z = 0000 0000 1101 0000 \*/



printf("z = %d", z); /\* z = 208 \*/

getch();



return 0;

}

Bir sayıyı sola bitsel olarak 1 kaydırmakla o sayının ikiyle çarpılmış değeri elde edilir. Bir sayıyı sağa bitsel olarak 1 kaydırmakla o sayının ikiye bölünmüş değeri elde edilir.



**İstenilen bitin değerinin 1 yapılması**



Bir sayının belirli bir bitini set etmek için, söz konusu sayı, ilgili biti 1 olan ve diğer bitleri 0 olan bir sayıyla veya işlemine tabi tutulmalıdır.



#include <stdio.h>

#include <conio.h>

int main()

{

short a = 0x0041; /\* a = 65 0000 0000 0100 0001 \*/

short maske = 0x0020; /\* maske = 32 0000 0000 0010 0000 \*/



printf("a = %d\n", a);

a |= maske; /\* a = 0000 0000 0110 0001 \*/

printf("a = %d", a); /\* a = 97 \*/



getch();

return 0;

}

x |=1 << k



**İstenilen bitin değerinin 0 yapılması**



Bir sayının belirli bir bitini sıfırlamak için, söz konusu sayı, ilgili biti 0 olan ve diğer bitleri 1 olan bir sayıyla bitsel ve işlemine tabi tutulmalıdır.

#include <stdio.h>

#include <conio.h>

int main()

{

int a = 0x0061; /\* a = 97 0000 0000 0110 0001 \*/

int maske = ~0x0020; /\* maske = ~32 1111 1111 1101 1111 \*/



printf("ch = %d\n", a);

a &= maske; /\* a 0000 0000 0100 0001 \*/

printf("ch = %d", a); /\* a = 65 \*/

getch();

return 0;

}

x &= ~(1 << k);

**Bir Sayının Belirli Bir Bitinin Değerinin Test Edilmesi (0 mı 1 mi)**

if (x & 1 << k)

/\* k. bit 1 \*/

else

/\* k. bit 0 \*/

**Bir Sayının Belirli Bir Bitini Ters Çevirmek**

x ^= 1 << k;

ÖRNEK 1: gönderilen veri içersinde istenilen aralıktaki bitleri sıfırlama ve birleme fonksiyonları



#include <stdio.h>

#include <conio.h>



int sifirla(int a, int start, int end)

{

int k;

for (k = start; k <= end; k++)

a &= ~(1 << k);

return a;

}

int birle (int a, int start, int end)

{

int k;

for (k = start; k <= end; k++)

a |= 1 << k;

return a;

}

int main()

{

int s;

s=sifirla(15,1,2); // 1111 -->1001

printf("%d\n",s);

s=birle(s,1,2); // 1001 -->1111

printf("%d\n",s);

getch();

return 0;

}

ÖRNEK 2: Fonksiyonel bir yapıda kişinin doğum tarihini tek bir tam sayıda tutan kod

#include <stdio.h>

#include <conio.h>

void ikiliyaz(int x)

{

int i = sizeof(int) \* 8 - 1;

for (; i >= 0; --i)

{

if (x >> i & 1 == 1)

putchar('1');

else

putchar('0');

if (i%4==0) putchar(' ');

}

printf("\n");

}

int sifirla(int a, int start, int end)

{

int k;

for (k = start; k <= end; k++)

a &= ~(1 << k);

return a;

}

int birle (int a, int start, int end)

{

int k;

for (k = start; k <= end; k++)

a |= 1 << k;

return a;

}

int bityaz(int a, int start, int end, int value)

{

int temp = value << start;

a=sifirla(a, start, end);

a |= temp;

return a;

}

int bitoku(int a, int start, int end)

{

int temp = a >> start;

temp=sifirla(temp, end-start + 1, sizeof(int) \* 8 - 1);

return temp;

}

int main()

{

int gun, ay, yil;

int data=0;

printf("Tarihi Giriniz \n");

printf("Gun : "); scanf("%d",&gun); // [0000 0000 0000 0000 0000 0000 000\*0 0100]

printf("Ay : "); scanf("%d",&ay); // [0000 0000 0000 0000 0000 0000 0000\* 0011]

printf("Yil : "); scanf("%d",&yil); // [0000 0000 0000 0000 0000 0000 \*0111 0100]

yil=yil-1900;

ikiliyaz(data);

getch();

data=bityaz(data,0,4,gun);

ikiliyaz(data);

getch();

data=bityaz(data,5,8,ay);

ikiliyaz(data);

getch();

data=bityaz(data,9,16,yil);

ikiliyaz(data);

getch();

printf("\nData :%d\n",data);

gun=bitoku(data,0,4);

ay=bitoku(data,5,8);

yil=bitoku(data,9,16);

printf("\nGun :%d Ay:%d Yil:%d\n",gun,ay,yil+1900);

getch();

return 0;

}

**ÖDEVLER:**

1. Kendisine parametre olarak gelen tamsayıyı bitsel operatörler kullanarak yine kendisine parametre olarak gelen sayı kadar left rotated shift yapan fonksiyon kodunu yazınız. Yazılan fonksiyonu program içeresinde kullanınız.
2. 0 ile 100 arasında rasgele oluşturduğunuz 100 sayıdan 5. bitleri 0 olan sayıların 4. bitini 1 yaparak sayilar0.txt dosyasına 5. bitleri 1 olan sayıların 3. bitini 0 yaparak sayilar1.txt dosyasına yazan program kodunu yazınız.