BLM112

PROGRAMLAMA DİLLERİ II

Yrd. Doç. Dr. Baha ŞEN

baha.sen@karabuk.edu.tr

KBUZEM
Karabük Üniversitesi
Uzaktan Eğitim Araştırma ve Uygulama Merkezi

1. BİT İLİŞKİLİ(BITWISE) İŞLEMLER

VE , VEYA ve DEĞİL işleçleri mantıksal ifadeleri bağlamakta kullanıldığı gibi doğrudan tamsayılar üzerinde de kullanılabilir. Bu kullanımda işleçler doğrudan tamsayıyı oluşturan bitleri etkiler.

1.2. VE (AND) İşlemcisi

x **&** y : x ve y tamsayılarının bitlerinin sırayla VE işlemine tutar.

Örnek:
5 & 3 = 1
(101) & (011) = (001)

1.3. VEYA (OR) İşlemcisi

x | y : x ve y tamsayılarının bitlerinin sırayla VEYA işlemine tutar.

Örnek:
5 | 3 = 7
(101) | (011) = (111)

1.4. DEĞİL (NOT) İşlemcisi

~ x : x tamsayının her bir bitinin tersini alır. 1'leri 0, 0'ları 1 yapar.

~ 5 = 2
~ (101) = (010)

1.5. ÖZELVEYA (EXOR) İşlemcisi

x ^ y : x ve y tamsayılarının bitlerinin sırayla ÖZEL VEYA işlemine tutar. (Exclusive OR)

Örnek:

5 ^ 3 = 6
(101) ^ (011) = (110)

1.6. SOLA KAYDIRMA << İşlemcisi

x << n : x sayısını n bit sola kaydırır. Boşalan yerlere 0 gelir.

Örnek:

5 << 1 = 10
(101) << 1 = (1010)

1.7. SAĞA KAYDIRMA >> İşlemcisi

x >> n : x sayısını n bit sağa kaydırır. Boşalan yerlere 0 gelir.

Örnek:

10 >> 1 = 5
(1010) >> 1 = (0101)

Gösterim

x	y	&	|	^
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Hex(16), Octal(8), Binary(2) ve DECIMAL gösterim

0xn	: n 16 lıdır (HEXADECIMAL)	0x11	: 17
0n	: n 8 lidir (OCTAL)	011	: 9
0bn	: n 2 lidir (BINARY)	0b101	: 5
n	: n 10 ludur (DECIMAL)	11	: 11

Klavyedeki tuşların durumunu gösterir bilgi bellekten okunduğunda her bitin anlamı şöyledir.

Bit No

0	sağ shift basılı/basılı değil
1	sol shift basılı/basıldı değil
2	kontrol tuşu basılı/basılı değil
3	alt tuşu basılı/basılı değil
4	scroll tuşu açık/kapalı
5	num lock tuşu açık/kapalı
6	caps lock tuşu açık/kapalı

Bu durumda num lock tuşunun açık olup olmadığını anlamak için okunan bilginin 5. bitinin değerini sınamak gerekecektir. Okunan bilginin x olduğunu varsayar ise;

```
if (x & 32)
    printf("num lock açık");
else
    printf("num lock kapalı");
```

x & 32 işlemine x değerinin 32 sayısı ile maskelenmesi denir. 32 değeri maske diye çağırılır. 32 sayısının 8 bit olduğunu düşünürsel sayının sadece 5. biti bir olup diğerleri sıfırdır.

32 = (0010 0000)

x & 32 işlemi ile x değerinin 5. bitinin bir olup olmaması sınanır. Bir ise sonuç 32 (farklı sıfır), sıfır ise sonuç 0 olur.

Örnek : Klavyedeki Num Lock tuşunun durumunu söyleyen programı yazınız.

```
#include<stdio.h>
#include<dos.h>
//Program TurboC derleyicisinde çalışır.
int main(void)
{
     int deger;
     puts("Durumlar");
     deger = peek(0x0040, 0x0017);

     if(deger & 16)
     puts("Scroll acik");
     else
     puts("Scroll kapali");

     if(deger & 32)
     puts("Num lock acik");
     else
     puts("Num lock kapali");

     if(deger & 64)
     puts("Caps lock acik");
     else
     puts("Caps lock kapali");
     getch();
}
```

İlgili işlevler

`poke (segment, offset, int value)`	X	`peek (segment, offset)`
`pokeb (segment, offset, char value)`	X	`peekb(segment, offset)`

Örnek: Öğrencinin doğum tarihi 2 Byte tamsayı olarak saklanmaktadır. Bu bilginin kodlaması şöyledir.

bit	anlamı
0 - 4	gün
5 - 8	ay
9 -15	yıl (+1970)

Böyle bir bilgiyi çözen program.

```
#include <stdio.h>
#include <dos.h>
int main()
{
 unsigned int i;

 scanf("%d", &i);

 printf("%2d", i&0x1f);
 i = i >> 5;
 printf("/%2d", i&0x1f);
 i = i >> 4;
 printf("/%4d", i+1970);

 return 0;

}
```

Örnek :

unsigned swap(unsigned) : gönderilen tamsayının düşük ile yüksek sekizlisinin yerini değiştirilmişini verir

unsigned max(unsigned) : gönderilen tamsayının düşük sekizlisi ile yüksek sekizlisinden büyüğünü verir

```
#include "stdio.h"
#include "conio.h"
unsigned short int swap(unsigned short int a);
unsigned short int max(unsigned short int a);
int main()
{
  unsigned short i, j;
  i = 0xAA11;
  printf("%x   %x   %x\n", i, swap(i), max(i));
  getch();
}

unsigned short int swap(unsigned short int a)
{ return (a>>8) | (a<<8);}
unsigned short int max(unsigned short int a)
{  unsigned short int lo, hi, m;
  lo = a & 0x00FF;
  hi = a >> 8;
```

```
  m = (lo>hi) ? lo:hi;
  return m;
}
```

Örnek : Aşağıdaki işlevlerin gerçekleştirilmesi.

void binary_yaz(unsigned x); { x tamsayısının 2li düzendeki karşılığını yazar }
unsigned copybits(x, b, n) { x sayısının sağdan b. bitinden itibaren n bitini verir}
unsigned ters(x, b, n): { x sayısının sağdan b. bitinden itibaren n bitini tersini alır}
unsigned rdon(x, n): { x sayısını n bit sağa döndürür}

```
#include "stdio.h"
#include "conio.h"
/* Programın başlangıcı */
typedef unsigned short int word; /* kolaylık için */

void binary_yaz(word);
word copybits(word x, word b, word  n);
word ters(word x, word b, word n);
word rdon(word x, word n);
word sdon(word x, word n);

int main()
{
    word  i, j;



    for (i=16; i>0; i--)//0-16 ya kadar ekrana yaz
        printf("%x ", i-1);
    printf("\n");

    i = 0xee;
    binary_yaz(i);
    printf("\n");
    j = copybits(i, 5, 4);
    binary_yaz(j);
    printf(" copybits(i, 5, 4) \n");

    j = ters(i, 5, 4);
    binary_yaz(j);
    printf(" ters(i, 5, 4)\n");
```

```
    j = rdon(i, 4);
    binary_yaz(j);
    printf(" rdon(i, 4)\n");

    getch();
    return 0;

}
word copybits(word x, word b, word  n)
/*  x sayısının sağdan b. bitinden itibaren n bitini
verir.  */
/*  ilk bitin numarası 0  */
{
    word i;

    i = x >> (b + 1 - n);   /*  ilgili bit bloğunun
sağa dayalı olacak  */
    /*  biçimde kaydır.                    */

    i = i & ~(~(0) << n);   /*  sağdaki n bit için
maske oluştur  */
    return i;
}   /*  End Of copybits  */
```

word ters(word x, word b, word n)

```
/*  x'in b. bitinden itibaren n bitin tersini alır
*/
{
    word p, r;

    p = ~(x);
    r = ~(~(0) << n) << (b - n + 1);   /*  seçilen
bitler için maske */
    p = p & r;

    x = x & ~(r);
    return x | p;
    /*
    8 bit için ters(. ,5,4)
          76543210
    x = 0110 1000
    p = 1001 0111
```

```
    r =  00111100    ~(~(0) << n) << (b - n + 1)
    p = 00010100       p = p & r
    x = 0100 0000      x & ~(r)
       = 01010100       x | p  dönen
    }   /*  End of test  */
}
```

```
word rdon(word x, word n)
/*  x'i n bit sağa döndürür  */
{
    word i;
    for (i = 1; i <= n; i++)
    {
        if (x % 2 == 1)
        {
            x = x / 2;
            x = x | 0x8000;
        }
        else
            x = x / 2;
    }
    return x;
}   /*  End of rdon  */
```

```
word sdon(word x, word n)
/*  x'i n bit sağa kaydırır. RDon fonksiyonun başka
biçimi */
{
    word i;

    i = copybits(x,n - 1,n);
    x = x >> n;
    i = i << (16 - n + 1);
    return x | i;
}   /*  End of RDon  */
```

```
void binary_yaz(word x)
{
    int i;
    word m;

    m = 0x8000;
```

```
    for (i = 0; i<16; i++)
    {
        if (x & m)
            printf("1");
        else
            printf("0");
        m = m >> 1;
    }
}
```