

# BLM 112- Programlama Dilleri II

## Hafta 12 Bit İşlem Operatörleri

Dr. Öğr. Üyesi Caner Özcan

*İnsanım, öyleyse özgürüm. Özgürsem, sorumlu olmalıyım.*

SARTRE

# İkilik Sayı Sistemi

- ▶ İkilik sayı sisteminde 0'lar ve 1'ler bulunur.
- ▶ Bilgisayar sistemleri yalnızca ikilik sayı sistemini kullanır.

$$(d_4d_3d_2d_1d_0)_2 = (d_0 \cdot 2^0) + (d_1 \cdot 2^1) + (d_2 \cdot 2^2) + (d_3 \cdot 2^3) + (d_4 \cdot 2^4)$$

$$(10011)_2 = (1 \cdot 2^0) + (1 \cdot 2^1) + (0 \cdot 2^2) + (0 \cdot 2^3) + (1 \cdot 2^4) = 19$$

# Onaltılık Sayı Sistemi

- ▶ Onaltılık (Hexadecimal) sayı sisteminde 16 farklı sembol bulunur.

**Decimal** : 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

**Hexadecimal** : 0 1 2 3 4 5 6 7 8 9 A B C D E F

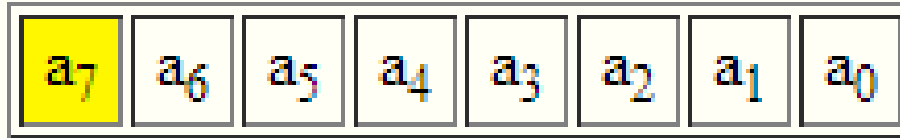
$$(3FC)_{16} = (3 \cdot 16^2) + (F \cdot 16^1) + (C \cdot 16^0) = 768 + 240 + 12 = 1020$$

$$(1FA9)_{16} = (1 \cdot 16^3) + (F \cdot 16^2) + (A \cdot 16^1) + (9 \cdot 16^0) = 4096 + 3840 + 160 + 9 = 8105$$

$$(75)_{16} = (7 \cdot 16^1) + (5 \cdot 16^0) = 112 + 5 = 117$$

# İkili Sistemde İşaretli Sayılar

- ▶ C dilinde değişkenler işaretli veya işaretsiz olabilir.
- ▶ 8 bitlik bir sayı düşünelim.



- ▶ Eğer sayı işaretli ise en soldaki biti işaret bitidir.
- ▶ İşaret bitinin 1 olması sayının negatif olduğunu, 0 olması pozitif olduğunu gösterir.

# İkili Sistemde İşaretli Sayılar

- ▶ İşaretli ikilik bir sayıyı onluk düzene çevirmek için

$$(a_7 a_6 a_5 a_4 a_3 a_2 a_1 a_0)_2 = (a_7 \cdot -2^7) + (a_6 \cdot 2^6) + \dots + (a_1 \cdot 2^1) + (a_0 \cdot 2^0)$$

- ▶  $(1011\ 1011)_2 = -69$  (eğer sayı işaretli ise)  
 $(1011\ 1011)_2 = 187$  (eğer sayı işaretsiz ise)
- ▶  $(1100\ 1101)_2 = -51$  (eğer sayı işaretli ise)  
 $(1100\ 1101)_2 = 205$  (eğer sayı işaretsiz ise)
- ▶  $(0110\ 1101)_2 = 109$  (eğer sayı işaretli ise)  
 $(0110\ 1101)_2 = 109$  (eğer sayı işaretsiz ise)

# Bit İşlem Operatörleri

- ▶ C dilinden bit seviyesindeki işlemler bit işlem operatörleri ile gerçekleştirilir.
- ▶ 8 bit 1 bayt oluşturmak üzere bir araya gelerek dijital donanım sistemlerinde erişilebilen en küçük veri formunu oluştururlar.
- ▶ Her bit 1 veya 0 değerini alır.

Sembol	Operator
&	Bitwise AND
	Bitwise Inclusive OR
^	Bitwise Exclusive OR
<<	Sola kaydır
>>	Sağa kaydır
~	Bire tümleyen

# Bit İşlem AND ( & )

- ▶ AND bit işlem operatörünün simgesi ampersand: &.
- ▶ Bit işlem AND operatörü AND operatörünün yaptığı işi ikilik sistemdeki iki sayının bitleri arasında gerçekleştirir.
- ▶ Girişlerden her hangi birinin 0 olması sonucun 0 olmasına sebep olur.
- ▶  $11001110 \& 10011000 = 10001000$
- ▶  $5 \& 3 = 1$  (  $101 \& 011 = 001$  )

# Bit İşlem OR ( | )

- ▶ Bit işlem OR, bit işlem AND ile aynı şekilde çalışır.
- ▶ Girişlerden herhangi birisi 1 ise çıkış 1 olur. Her iki girişte 0 ise çıkış 0 olur.
- ▶ Sembolü '|'
- ▶  $11001110 \mid 10011000 = 11011110$
- ▶  $5 \mid 3 = 7$  ( $101 \mid 011 = 111$ )



# Bit İşlem Exclusive OR ( ^ )

- ▶ Bit işlem XOR mantıksal XOR işlemi gerçekleştirir.
- ▶ İki bitin toplamında elde var ise onun çıkarılmasını sağlar.
- ▶ Girişler farklı ise çıkış 1, aynı ise çıkış 0 dır.
- ▶ XOR bazen bitleri 1 ile 0 arasında sürekli değiştirmek için kullanılır.
- ▶  $i = i \wedge 1$  işlemi bir döngü içinde kullanılırsa değerini 1 ile 0 arasında değiştirir.
- ▶  $5 \wedge 3 = 6$  (  $101 \wedge 011 = 110$  )

# Bit İşlem Operatörleri

bit a	bit b	a & b (a AND b)	a   b (a OR b)	a ^ b (a XOR b)
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

# Sağa Kaydırma ( >> )

- ▶ Sağa kaydırma sembolü: >>
- ▶ Çalışması için iki operanda ihtiyaç duyar.
- ▶ Soldaki operandın tüm bitlerini sağdaki operandın değeri kadar sağa kaydırır.
- ▶ `number >> 3` işlemi `number` sayısının tüm bitlerini 3 kez sağa kaydırır.
- ▶ En solda oluşan yeni boşluklar sıfırlar ile doldurulur.
- ▶ Bir sayıyı sağa kaydırmak onu ikiye bölmek demektir.
- ▶  $10 \gg 1 = 5$        $(1010) \gg 1 = (0101)$

# Sağa Kaydırma ( >> )

- ▶ Eğer sayı işaretli ise, işaret genişletme kuralı uygulanır.
- ▶ İşaret genişletmede sayının sağa kaydırılması sonucu solda oluşacak boşlukların sayının en solundaki bitin değeriyle doldurulmasıdır.

```
100000000000000000000100000001100000000  
1111000000000000000000001000000011000000
```

- ▶ Bu örnek orijinal sayının en soldaki biti 1 olduğundan sağa kaydırma sonucu oluşan boşluklar 1 ile doldurulmuştur.

# Sola Kaydırma ( << )

- ▶ Sola kaydırmanın sembolü: <<
- ▶ Sayının bitlerini sola kaydırır. Sağa kaydırma operatörünün tersi işlem yapar.
- ▶ En sağda oluşan yeni boşluklar sıfırlar ile doldurulur.
- ▶ Sola kaydırma bir sayının iki ile çarpılmasıdır.
- ▶  $5 \ll 1 = 10$       $(101) \ll 1 = (1010)$

# Birin Tümleyeni ( $\sim$ )

- ▶ Birin tümleyeni ( $\sim$ ) bir sayının tümleyenini verir.
- ▶ Bir sayının tüm bitleri ters çevrilir. Her bit için bit değeri 1 ise 0, 0 ise 1 yapılır.
- ▶  $\sim 5 = 2$  ( $\sim 101 = 010$ )

# Bit İşlem Operatörlerinin Kullanımı

- ▶ Bit işlem operatörlerini öğrendik, ancak bu operatörleri kod yazarken nasıl kullanabiliriz?
- ▶ OR operatörü bitlerin birleşimidir. İki sayının 1 olan bitlerini birleştirir.

```
10101010101010101010101010101010101010101010
01010101010101010101010101010101010101010101
|-----
11111111111111111111111111111111111111111111
```

# Bit İşlem Operatörlerinin Kullanımı

- ▶ AND operatörü bitlerin kesişimidir. İki sayıda da 1 olan bitleri 1 yapar.

```
1010101010101010101010101010101010101010101010101010
0101010101010101010101010101010101010101010101010101
& -----
0000000000000000000000000000000000000000000000000000
```

- ▶ Bu örnekte iki sayıda da 1 olan bit yoktur ve sonuç tüm bitler için 0 çıkar.



# Bit İşlem Operatörlerinin Kullanımı

- ▶ OR operatörü bir sayının istenen bitlerini 1 yapmak için kullanılabilir.

Önce	:	00000000111111110000000011111111
1 yapılacak bitler	:	000 <b>1</b> 0000000000000000 <b>1</b> 000000000000
Sonra	:	00010000111111110001000011111111

- ▶ AND operatörü bir sayının herhangi bir bitinin 1 olup olmadığını kontrol etmek için kullanılabilir.

000001110101101 <b>1</b> 1100110100010101	
00000000000000001000000000000000	→ Maske

# Örnek: Klavye Kodları

- Bilgisayar klavyesinin durumunu gösteren 8 bitlik bir sayı hafızadan okunmuş olsun. Her bitin anlamı:

Bit	Durum
0	Sağ shift basılı mı
1	Sol shift basılı mı
2	Ctrl basılı mı/pressed/not
3	Alt basılı mı
4	Scroll on/off
5	Num Lock on/off
6	Caps Lock On/off

# Örnek: Klavye Kodları

- ▶ Numlock tuşunun aktif olup olmadığının anlaşılabilmesi için klavye bilgisini tutan sayının 6. biti kontrol edilmelidir.
- ▶ Bunun için klavye bilgisini tutan sayı, 6. biti 1 olan ve diğer bitleri 0 olan bir sayı ile (32 sayısı) AND işlemine tabi tutulmalıdır.
- ▶ Örneğin klavye bilgisi 01101011 olsun,  
01101011 &  
00100000 → Mask
- ▶ Klavye bilgisinin 6. biti 1 olduğundan sonuç sıfırdan farklı bir sayı çıkacaktır.
- ▶ Sonucun 0 olması kontrol edilen bitin 0 olduğu diğer durumda 1 olduğu anlamına gelir.

# Örnek: Ipv4 Adresi

- ▶ IPv4 adresi ağ paketlerinden taşınan 32 bitlik bir sayıdır.
- ▶ Sayının her 8 biti ip numarasının birbirinden nokta ile ayrılmış segmentini oluşturur.
- ▶ Örneğin: 192.168.1.2 olarak verilen ip numarası hexadecimal sistemde 0xc0a80102 olur.
- ▶ 32 bitlik ip adresi okuyup onu bizim anlayacağımız şekilde bir ip adresine dönüştüren programın kodunu nasıl yazabiliriz?

# Örnek: Ipv4 Adresi

- ▶ Bunun için 32 bitlik ip adresinin her 8 bitini uygun bir maske ile AND işleminden geçirerek almalıyız.
- ▶ Örneğin ip adresinin en düşük 8 bitini almak istiyorsak kullanmamız gereken maske `0x000000ff` olmalıdır.
- ▶ Bu maske sayının ilk 8 bitini koruyacaktır.

# Örnek: Ipv4 Adresi

- ▶ Eğer sayıdan korunarak alınan bitleri sayının ilk 8 biti değilse bu bitleri en sağdaki 8 bite kaydırmalıyız.

**Value : 11000000101010000000000100000010 c0a80102 3232235778**

**Mask : 11111111000000000000000000000000 ff000000 4278190080**

**Result : 11000000000000000000000000000000 c0000000 3221225472**

- ▶ Sayının en solundan korunarak alınan 8 bitin değeri 3221225472'dir. Bu sayı bizim istediğimiz 192 değildir.
- ▶ Çünkü elde edilen ikilik sayı en düşük 8 bitte değildir. Bunun için elde edilen sonucu 24 bit sağa kaydırmalıyız. ( $\gg 24$ )

**Value : 11000000101010000000000100000010 c0a80102 3232235778**

**Mask : 11111111000000000000000000000000 ff000000 4278190080**

**Result : 000000000000000000000000011000000 c0000000 192**

# Örnek: Ipv4 Adresi

```
1  #include <stdio.h>
2  int main(void)
3  {
4      unsigned int ipAdres = 0xc0a80102;
5      unsigned maske =0xff000000;
6      int segment1,segment2,segment3,segment4;
7      int i, bit=32;
8      unsigned tmp;
9      for(i=1;i<=4;i++)
10     {
11         tmp = ipAdres & maske;
12         if(i!=4){
13             maske = maske >> 8;
14             tmp = tmp >> (bit-i*8);
15             printf("%d.",tmp);
16         }
17         else printf("%d",tmp);
18     }
19
20     getchar();
21     return 0;
22 }
```

# Örnek: İkili Toplama

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  //binary addition
5  int main()
6  {
7      unsigned int x=3, y=1, sum, carry;
8      sum = x ^ y;
9      carry = x & y;
10     while(carry!=0)
11     {
12         carry = carry << 1;
13         x = sum;
14         y = carry;
15         sum = x ^ y;
16         carry = x & y;
17     }
18     printf("%d", sum);
19     getchar();
20     return 0;
21 }
```



# Kaynaklar

- ▶ Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015.
- ▶ Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002.
- ▶ Paul J. Deitel, “C How to Program”, Harvey Deitel.
- ▶ “A book on C”, All Kelley, İra Pohl

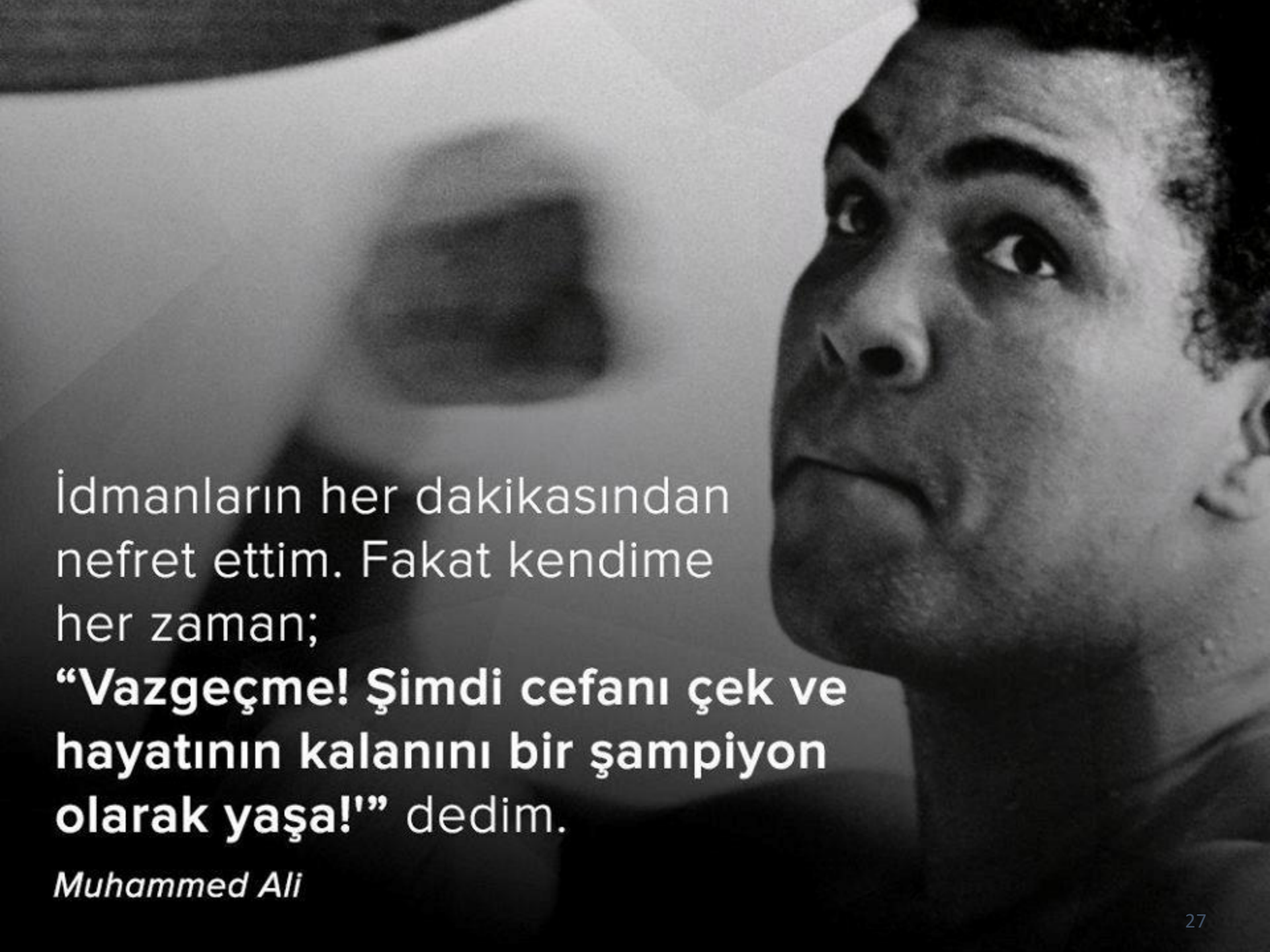
S o r u l a r  
?



Dinlediğiniz için teşekkürler

CANER ÖZCAN

 canerozcan@karabuk.edu.tr



İdmanların her dakikasından  
nefret ettim. Fakat kendime  
her zaman;

**“Vazgeçme! Şimdi cefanı çek ve  
hayatının kalanını bir şampiyon  
olarak yaşa!”** dedim.

*Muhammed Ali*