10 Kayan Noktalı Sayılar (*Floating Point Numbers*)

10 Tabanı:

$(12.34)_{10} = 12 + \frac{34}{100}$ ya da $(12.34)_{10} = 12 + \frac{3}{10} + \frac{4}{100} = 1 \cdot 10^1 + 2 \cdot 10^0 + 3 \cdot 10^{-1} + 4 \cdot 10^{-2}$

2 Tabanı:

$(101.11)_2 = 5 + \frac{3}{4}$ ya da $(101.11)_2 = 5 + \frac{1}{2} + \frac{1}{4}$, $(0.1111)_2 = \frac{15}{16}$, $(0.1)_2 = \frac{1}{2}$

Gerçek (*real numbers*) sayıları (çok küçük ve çok büyük sayıları da) bilgisayar belleğinde tutmak için üstel gösterilim (bilimsel gösterilim) kullanılır.

10.1 Üstel gösterilim (*Scientific notation, exponential notation*)

$\pm F \times B^{\pm E}$

F: *Fraction* (Kesir, Mantis)
E: *Exponent* (Üs)
B: *Base* (Taban)

Bellekte ± , **F** ve **E** tutulur.
Taban **B**'nin bellekte tutulmasına gerek yoktur.
Tüm sayılar için aynıdır (bilgisayarlarda B=2).

Örnekler (10 tabanı):

a)
976,000,000,000,000 = **+0.976**x10^{15}
+976 ve +15 bellekte tutulur.

b)
0.000 000 000 000 976 = **+0.976**x10^{-12}
+976 ve -12 bellekte tutulur.

Normalize Sayı:

Bir sayı üstel notasyonda farklı şekillerde yazılabilir.
Örneğin; 3.14 = 314x10^{-2} = 3.14x10^{0} = 0.314x10^{1}

Normalize gösterilimde noktanın yerine önceden karar.
Örneğin, noktanın her zaman sıfırdan farklı en yüksek anlamlı sayının solunda olduğu kabul edilir ve üs E ona göre ayarlanır.

Örnek:

3.14 normalizasyon $\rightarrow$ 0.314x10^{1}
Bu sayı bellekte şöyle tutulur: ±F ±E $\rightarrow$ +314 +01
Taban B'yi (bu örnekte 10) ve noktanın yerini bellekte tutmaya gerek yoktur.

Başka normalizasyon yöntemleri de vardır.
Örneğin nokta en yüksek anlamlı sayının sağına da yerleştirilebilir.

Yükseltilmiş Üs (*Biased Exponenet*):

Üs değeri işaretli bir sayıdır.
İşaretli sayıları karşılaştırmak (2'ye tümleyen yöntemi) zordur.
Üs değerinin negatif olmaması için üs değeri bellekte saklanmadan önce belli bir değer "ökçe" (*bias*) ile toplanır (üs yükseltilir).
Böylece üssün işaretinin saklanmasına gerek kalmaz ve aritmetik işlemlerde (karşılaştırmada) kolaylık sağlanır.

10.2 IEEE 754 Standardı (1985, güncelleme 2008)

IEEE "Standard for Floating-Point Arithmetic" (IEEE 754) 1985'te IEEE (the Institute of Electrical and Electronics Engineers) tarafından oluşturulmuştur ve günümüzde de bilgisayar sistemlerinde kullanılmaktadır.

Single (32 bit)

İşaret	E	F
S	Üs	Kesir
1	8	23

Üs 127 yükseltilmiştir.

Double (64 bit)

İşaret	E	F
S	Üs	Kesir
1	11	52

Üs 1023 yükseltilmiştir.

E'deki bit sayısı k olmak üzere üs $(2^{k-1} - 1)$ kadar yükseltilir.

Güncel standartta 16 bitlik (*half*) ve 128 bitlik (*quadruple*) sayılar da bulunmaktadır.

Normalize Sayı (IEEE 754):

Noktanın her zaman sıfırdan farklı en yüksek anlamlı sayının sağında olduğu kabul edilir.

İkili düzende çalışıldığına göre "0"dan farklı sayı "1"dir.

Örnek:

$$(10110.101)_2 \xrightarrow{\text{(normalizasyon)}} \mathbf{1}.0110101 \times 2^4$$

Noktadan önce her zaman 1 olduğu bilindiğinden bu 1 değeri de bellekte tutulmaz.

Buna **gizli 1** (*hidden one*) denir.

Örnek:

$(+22.625)_{10}$ sayısı IEEE 754 32 bit formatında nasıl bellekte tutulur?

$(22)_{10} = (10110)_2$

.625'in 2 tabanındaki karşılığının bulunması:

2 x 0.625 = **1** + 0.25
2 x 0.25 = **0** + 0.5
2 x 0.5 = **1** + 0

Yüksek anlamlı bit

5/8

$\Rightarrow (0.625)_{10} = (0.101)_2$

Gizli 1. Bellekte tutulmaz.

Taban 2, bellekte tutulmaz.

$(+22.625)_{10} = (+10110.101)_2 = +1.0110101 \times 2^4$ (Normalize)

IEEE Single:

0	10000011	0110101...000
İşaret +	Yükseltilmiş üs 4 + 127 = 131	Kesir (Mantis)
1 bit	8 bit	23 bit

Mantisin düşük anlamlı bitleri "0" ile dolduruldu.
Çünkü solunda nokta (.) var.

Sayının 10 tabanındaki değeri (N):

Single: $N = (-1)^S (1.F) 2^{E-127}$ *Double*: $N = (-1)^S (1.F) 2^{E-1023}$

$N = (-1)^S (1 + \frac{F}{2^{23}}) 2^{E-127}$ $N = (-1)^S (1 + \frac{F}{2^{52}}) 2^{E-1023}$

Sıfır için özel bir gösterilim kullanılır: E=0 ve F=0 → N=0

Örnekler (*Single*):

S	E	F	Değer (Anlamı):
0	10010011	10100010000000000000000	= +1.1010001 x 2^{10100} = +1.6328125 x 2^{20}
1	10010011	10100010000000000000000	= - 1.1010001 x 2^{10100} = -1.6328125 x 2^{20}
0	01101011	10100010000000000000000	= +1.1010001 x 2^{-10100} = +1.6328125 x 2^{-20}
1	01101011	10100010000000000000000	= - 1.1010001 x 2^{-10100} = -1.6328125 x 2^{-20}

81/128

E (üs) ve F(mantis) özel değerleri:

1. E = 0 ve F = 0 → N = 0 İki tane sıfır var (+,-)
2. E = 255 , F = 0 → N = ±∞
3. E = 255, F ≠ 0 → NaN (*Not a Number*) 0/0 , ∞/∞
4. E = 0, F ≠ 0 → Normalize olmayan sayı

Son durum normalize olarak gösterilemeyen mutlak değeri çok küçük sayılar için kullanılır.

Normalize olmayan sayılar:

Normalize yapı ile mutlak değeri çok küçük olan sayıları göstermek mümkün değildir.

Normalize olarak gösterilebilecek en küçük sayı S =0, E = 0000 0001 , F = 000.... 0

N= $+(1+0)2^{1-127} = 2^{-126}$

0 ile 2^{-126} arasındaki sayılar normalize olarak gösterilemez.

Normalize olmayan biçim (E = 0, F $\neq$ 0) kullanılırsa sayının 10 tabanındaki değeri öncekinden farklı olarak aşağıdaki gibi hesaplanır:

Single: $N = (-1)^S (\frac{F}{2^{23}}) 2^{-126}$ *Double*: $N = (-1)^S (\frac{F}{2^{52}}) 2^{-1022}$

Normalize olmayan en küçük sayı: S =0, E = 0000 0000 , F = 000.... 01

N= $+(1/2^{23})2^{-126} = 2^{-149}$

0 ile 2^{-149} arasındaki sayılar gösterilemez (*underflow*).

Sınır değerleri:

Mutlak değeri en küçük sayı:

Single: 2^{-149} (Denormalize) , Double: 2^{-1074} (Denormalize)

Mutlak değeri en büyük sayı:

Single: 0 11111110 11111111111111111111111 = $(2-2^{-23})x2^{127} \approx 10^{38.53}$ (Normalize)

Double: $(2-2^{-52})x2^{1023} \approx 10^{308.3}$ (Normalize)

Örnek: Kayan Noktalı Sayıların Toplanması ve Çıkartılması

Aritmetik İş Hattı

X= $A*2^a$

Y= $B*2^b$