# SORU ÇÖZME

## Koddan Karmaşıklık Analizi

**Algoritmalar**

**iterative**
```
A()
{
for i=1 to n
max(a, b)
}
```

**Recursive**
```
A(n)
{
if()
A(n/2)
}
```

### İteratif Koddan Karmaşıklık Hesaplama

**ÖR:**

```
A()
{
  int i;
  for(i=1 to n)
      print("Merhaba")                    // n kez print yazılacak
 }
```

**Karmaşıklığı= O(n)**

**ÖR:**

```
A()
{
  int i;
  for(i=1 to n)                    // n kez
   for(j=1 to n)                   // n kez
      print("Merhaba")             // n^2 kez print yazılacak
 }
```

**Karmaşıklığı= O($n^2$)**

**ÖR:** A()

```
{
  i=1, s=1;
  while(s<=n)
  {
    i++;       →          lineer artar
    s=s+i;     →          i'ye bağlı artar
    printf("Merhaba")
  }
}
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| s | 1 | 3 | 6 | 10 | 15 | 21 | .......... n |
| i | 1 | 2 | 3 | 4 | 5 | 6 | .......... k |

Burada algoritma k İterasyon ilerleyecektir. Dolayısıyla kaç kez çalıştığını bulmak için k'yı bulmak gerekiyor.

$$\frac{k(k+1)}{2} > n$$

$$\frac{k^2+k}{2} > n$$

**k = O($\sqrt{n}$)**

**ÖR:**

```
A()
 {
    i=1
    for(i=1; i^2<=n; i++)
       printf("Merhaba");            // √n
 }
```

**O($\sqrt{n}$)**

**ÖR:** A()

```
 {
   int i, j, k, n;
    for(i=1; i<=n; i++)                          //n
    {
      for(j=1; j<=i; j++)                        //i
      {
        for(k=1; k<=100; k++)                    //100
        {
           printf("Merhaba")
        }
       }
    }
  }
```

| | | | | | |
|---|---|---|---|---|---|
| i=1 | i=2 | i=3 | i=4 | | i=n |
| j=1 | j=2 | j=3 | j=4 | …………… | j=n |
| k=100 | k= 2 X 100 | k=3 X 100 | k=4 X 100 | | k=n X 100 |

100 + 2 X 100 + 3 X 100 + 4 X 100+………..+n X 100

=100 (1 + 2 + 3 + …………+ n)

$=100(\frac{n.(n+1)}{2})$

**= O(n²)**

**ÖR:** A()

```
{
  int i, j, k, n;
  for(i=1; i<=n; i++)
  {
    for(j=1; j<=i^2; j++)
    {
      for(k=1; k<=n/2; k++)
      {
        Printf("Merhaba");
      }
    }
  }
}
```

| | | | | | |
|---|---|---|---|---|---|
| i=1 | i=2 | i=3 | i=4 | | i=n |
| j=1 | j=4 | j=9 | j=16 | ………….. | j=$n^2$ |
| k=n/2 X 1 | k= n/2 X 4 | k=n/2 X 9 | k=n/2 X 16 | | k=n/2 X $n^2$ |

n/2*1 + n/2*4 + n/2*9 + n/2*16 + ………… + n/2*$n^2$

= n/2 (1 + 4 + 9 + 16 + ……. + $n^2$)

= n/2 ($\frac{n(n+1)(2n+1)}{6}$)

**= O($n^4$)**

**ÖR:**

```
A()
{
  for(i=1; i<n; i=i*2)
    printf("Merhaba")
}
```

i = 1, 2, 4, 8, …….. , n

$2^0, 2^1, 2^2, 2^3, \ldots, 2^k$

$2^k = n$ ise $k = \lg n$ olur.

**O(lgn)**

**ÖR:**

```
A()
{
  int i, j, k;
  for(i=n/2; i<=n; i++)              // n/2
    for(j=1; j<=n/2; j++)            // n/2
      for(k=1; k<=n; k=k*2)          // lgn
        printf("Merhaba")
}
```

n/2 * n/2 * lgn

**= O($n^2$lgn)**

**ÖR:**

```
A()
{
  int i, j, k;
  for(i=n/2; i<=n; i++)              // n/2
    for(j=1; j<=n; j=2*j)            // lgn
      for(k=1; k<=n; k=k*2)          // lgn
        printf("Merhaba")
  }
```

$n/2 * (\lg n)^2 \longrightarrow$ **$O(n(\lg n)^2)$**

**ÖR:**

```
A()
{
  for(i=1; i<=n; i++)
    for(j=1; j<=n; j=j+i)
     printf("Merhaba")
 }
```

| | | | | |
|---|---|---|---|---|
| i=1 | i=2 | i=3 | .................... | i=n |
| j=1 to n | j=1 to n | j=1 to n | | j=1 to n |
| n kez | n/2 kez | n/3 kez | | n/n kez |

$= n(1 + \frac{1}{2} + \frac{1}{3} + \quad ... \quad + \frac{1}{n})$ lgn (Aritmetik seri toplamından)

= n(lgn)

**= O(nlgn)**

**ÖR:**

```
A()
{
  int n=2^(2^k)
  for(i=1; i<=n; i++)
  {
    j=2
    while(j<=n)
    {
      j=j^2
      print("Merhaba")
    }
  }
}
```

| | | |
|---|---|---|
| k=1 | k=2 | k=3 |
| n=4 | n=16 | n=256 |
| j=2, 4 | j=2, 4, 16 | j=2, 4, 16, 256 |
| n*2 kez | n*3 kez | n*4 kez |

$n=2^{2^k}$ $\quad$ $lg_2 n = 2^k$

$lglgn = k$ → n(k+1) kez çalışır

$= n(lglgn+1)$

**$= O(nlglgn)$**

**ÖR:** Aşağıdaki tabloda karmaşıklıkları verilen algoritmalar 1 GHz'lik bir bilgisayarda verilen süreler boyunca çalıştırılırsa kaç tane veri işlerler?

| | $n^2$ | (3/2)n | $\sqrt{n}$ |
|---|---|---|---|
| 1 dk | | | |
| 1 saat | | | |

**Çözüm:**

| | $n^2$ | (3/2)n | $\sqrt{n}$ |
|---|---|---|---|
| 1 dk | 245.000 | $4.10^{10}$ | $36.10^{20}$ |
| 1 saat | $147.10^{5}$ | $240.10^{10}$ | $2160.10^{20}$ |

**ÖR:** n*n matrisinin transpozunu alan algoritmayı yazarak karmaşıklığını bulunuz?

**Çözüm:**

```
TRANS(X, Y, N)
for i←1 to N
  do for j←1 to N
    do Y[j,i]←X[i,j]
```

**O($N^2$)**

**ÖR:** $T(n) = 4T(n/2) + n^2$ tekrarlı bağıntısını çözerek asimptotik notasyonunu yazınız?

**Çözüm:**
Master teoremini hatırlayalım.

$\lg_b a$

$T(n) = aT(n/b) + f(n)$ ise $a>=1$ ve $b>1$ olmak koşulu ile

$$T(n)= \begin{cases} \Theta(n^{\lg_b a}) & f(n) = O(n^{\lg_b a-\epsilon}) \longrightarrow \text{1. Durum} \\ \Theta(n^{\lg_b a}\lg n) & f(n) = \Theta(n^{\lg_b a}) \longrightarrow \text{2. Durum} \\ \Theta(f(n)) & f(n) = \Omega(n^{\lg_b a+\epsilon}) \longrightarrow \text{3. Durum} \end{cases}$$

$a=4$ $b=2$ $f(n)=n^2$

$\lg_2 4 = 2$

$f(n) = n^2 \longrightarrow$ 2. Durum

**$T(n) = \Theta(n^2 . \lg n)$**

**ÖR:** $T(n) = 3T(n/2) + n^2$ tekrarlı bağıntısını çözerek asimptotik notasyonunu yazınız?

**Çözüm:**

$a = 3$
$b = 2$

$lg_b a = lg_2 3 = 0,477$

$n^2 > n^{0,477} \longrightarrow$ olduğundan $f(n) = \Omega(n^{0,477})$ olur

Yani 3. Durum

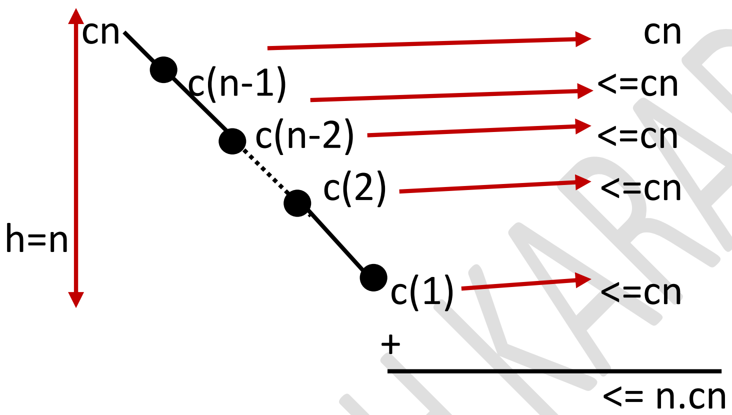
**$\Theta(n^2)$**

**ÖR:** $T(n) = T(n-1) + n$ tekrarlı bağıntısını çözerek asimptotik notasyonunu yazınız?

Master teoremini kullanamayız !!
Neden?

Çünkü a>=1   b>1 olmalı.
Burada b=-1
O zaman tekrarlı bağıntı için öz yineleme ağacı kullanalım.

$T(n) = T(n-1) + \Theta(n)$
$T(n) = T(n-1) + cn$




**O(n²)**