# CENG 114 BİLGİSAYAR BİLİMLERİ İÇİN AYRIK YAPILAR

Prof. Dr. Tufan TURACI
tturaci@pau.edu.tr

- Pamukkale Üniversitesi
- Mühendislik Fakültesi
- Bilgisayar Mühendisliği Bölümü

- Hafta 7

# Ders İçereği

- **Algoritma Analizi (Hafta 6 Devam)**

  **--- Yürütme Zamanı**

  **--- Karmaşıklık**

  **--- Çeşitli Örnekler**

- **Listeler ve Sayma (Kombinatorik)**

  **--- Permütasyon**

  **--- Güvercin Yuvası Prensibi**

# Algoritma ve Algoritma Analizi

Bir algoritma, bir problemi çözmek ya da bir işlevi hesaplamak için izlenecek sonlu, açıkça belirtilen talimat dizisidir.

Bir algoritma genel olarak

- *Bir (birkaç) girdi alır.*
- *Sınırlı bir süre içerisinde komutlar bir çıktı üretmektedir.*

Etkili bir talimat, temelde kalem ve kağıt kullanarak gerçekleştirmenin mümkün olduğu kadar basit bir işlemdir.

# Algoritmaların Analizi

- Bir algoritmanın complexity'sini (karmaşıklığı) çalışma
  - *Algoritma ne kadar iyi?*
  - *Diğer algoritmalarla karşılaştırma işlemi nasıl yapılacak?*
  - *En iyi yazılabilecek algoritma bu mudur?*

- Karmaşıklık
  - *Alan Karmaşıklığı*
    - Bit sayısı
    - Eleman sayısı

  - *Zaman Karmaşıklığı*
    - Toplamda çalıştırılacak işlem sayısı
      - *Modele göre değişir*
      - *RAM*

# Algoritmaların Run-Time (Çalışma Zamanı) Analizi

- Algoritma karmaşıklığı, problemin boyutunu gösteren parametre n’ nin bir fonksiyonu olarak hesaplanabilmektedir.

- Zaman karmaşıklığı, T (n), algoritmanın en önemli işlemi olan - temel işlem olarak adlandırılan – işlemin çalıştırılma sayısı olarak hesaplanabilir.

- Space (Alan) karmaşıklığı, S (n), genellikle algoritmanın yürütülmesi sırasında kullanılan bellek alanının büyüklüğü olarak hesaplanır.

# Tablo Metodu

- *Tablo Metodu, bir algoritmanın karmaşıklığını hesaplamak için kullanılır*

## Örnek: <u>Bir dizinin elemanlarını toplama</u>

| | İşlem sayısı |
|---|---|
| `top = 0` | 1 |
| `for (i=0; i<n; i++){` | 1+n+1+n |
| `top = top + a[i]}` | n |
| `print top` | 1 |
| | **3n+4** |

Kaç işlem yapılır? T(n)=3n+4 (Yürütme zamanı)

# Tablo Metodu

- Örnek:

<u>Matris Toplama</u>

*a, b, c* ‘nin mxn boyutunda matrisler olduğunu varsayalım.

| | işlem | toplam |
|---|---|---|
| `for (i=0; i<m; i++){` | 1+m+1+m | 2m+2 |
| `for (j=0; j<n; j++){` | m(1+n+1+n) | 2mn+2m |
| c[i,j] = a[i,j] + b[i,j] } } | mn | mn |
| | | **3mn+4m+2** |

Eğer m=n ise $T(n) = 3n^2 + 4n + 2$ (Yürütme zamanı)

# Asimptotik Notasyon Ve Temel Verimlilik  Sınıfları

## (Büyüme Sırası) Order of growth

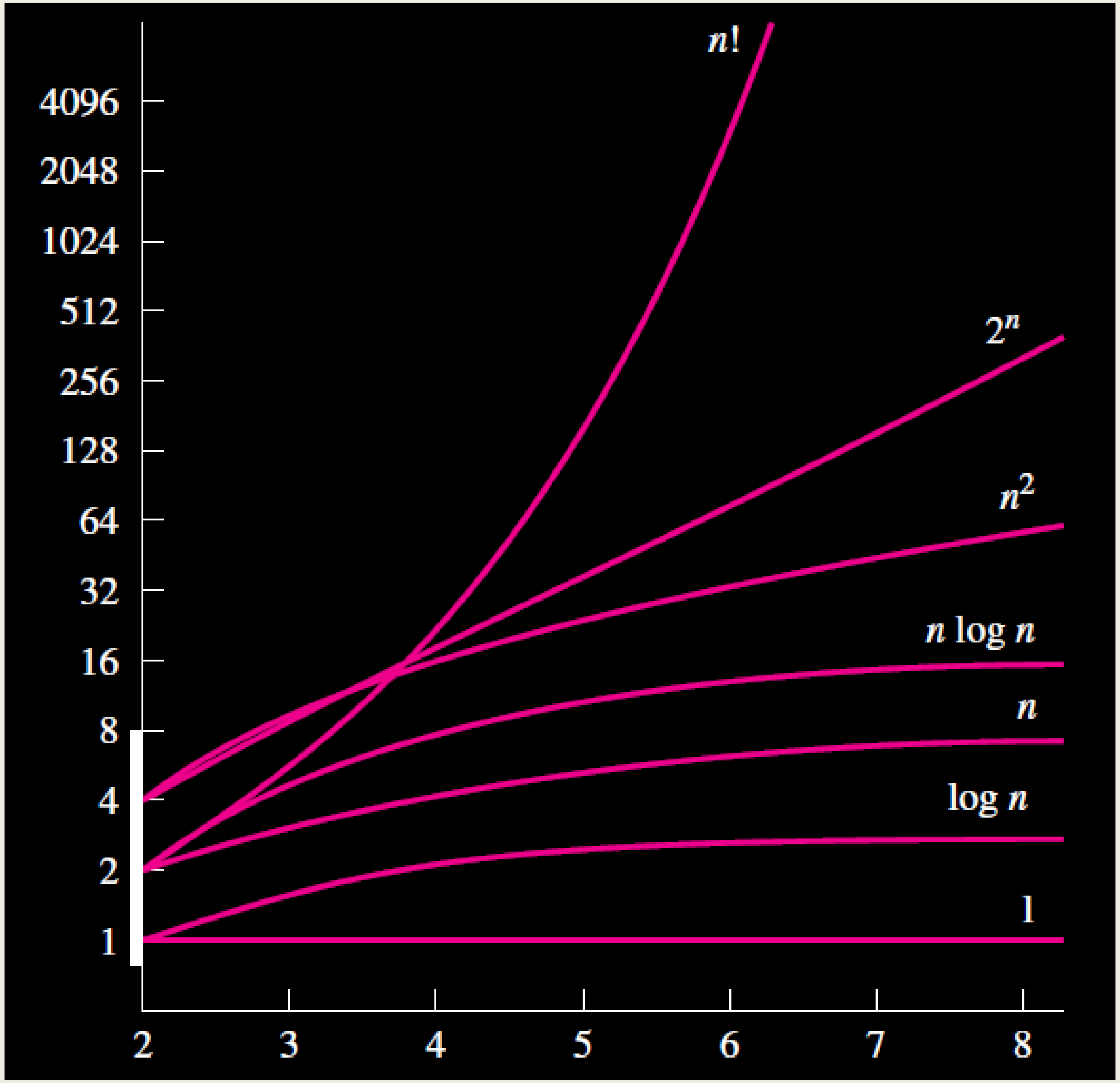
- En önemlisi : $n \rightarrow \infty$'a giderken algoritmanın performansı hangi sınırlarda bunu anlayabilmektir.

- Örnek:
  - *İki katı kadar hızlı bir bilgisayarda algoritma ne kadar hızlanıyor?*
  - *Girdi boyutu iki katına çıktığında algoritma ne kadar yavaşlıyor?*

# $n \to \infty$ giderken bazı önemli fonksiyonların değerleri

**Tablo:** Algoritma analizi için bazı önemli fonksiyonların değerleri

| $n$ | $\log_2 n$ | $n$ | $n \log_2 n$ | $n^2$ | $n^3$ | $2^n$ | $n!$ |
|---|---|---|---|---|---|---|---|
| $10$ | $3.3$ | $10^1$ | $3.3 \cdot 10^1$ | $10^2$ | $10^3$ | $10^3$ | $3.6 \cdot 10^6$ |
| $10^2$ | $6.6$ | $10^2$ | $6.6 \cdot 10^2$ | $10^4$ | $10^6$ | $1.3 \cdot 10^{30}$ | $9.3 \cdot 10^{157}$ |
| $10^3$ | $10$ | $10^3$ | $1.0 \cdot 10^4$ | $10^6$ | $10^9$ | | |
| $10^4$ | $13$ | $10^4$ | $1.3 \cdot 10^5$ | $10^8$ | $10^{12}$ | | |
| $10^5$ | $17$ | $10^5$ | $1.7 \cdot 10^6$ | $10^{10}$ | $10^{15}$ | | |
| $10^6$ | $20$ | $10^6$ | $2.0 \cdot 10^7$ | $10^{12}$ | $10^{18}$ | | |


n!
4096
2048
1024
512
256
128
64
32
16
8
4
2
1
$2^n$
$n^2$
n log n
n
log n
1
2
3
4
5
6
7
8

# Asimptotik Büyüme Dereceleri

Asimtotik Analiz

$n \to \infty$ iken $T(n)$'nin büyümesi nedir?

Asimtotik Notasyon

| | | |
|---|---|---|
| $O$ | $\Omega$ | $\Theta$ |
| Büyük O | Büyük omega | Teta |
| (Big O) | (Big Omega) | (Theta) |

$O$ Notasyonu (Üst Sınır)

$\Omega$ Notasyonu (Alt Sınır)

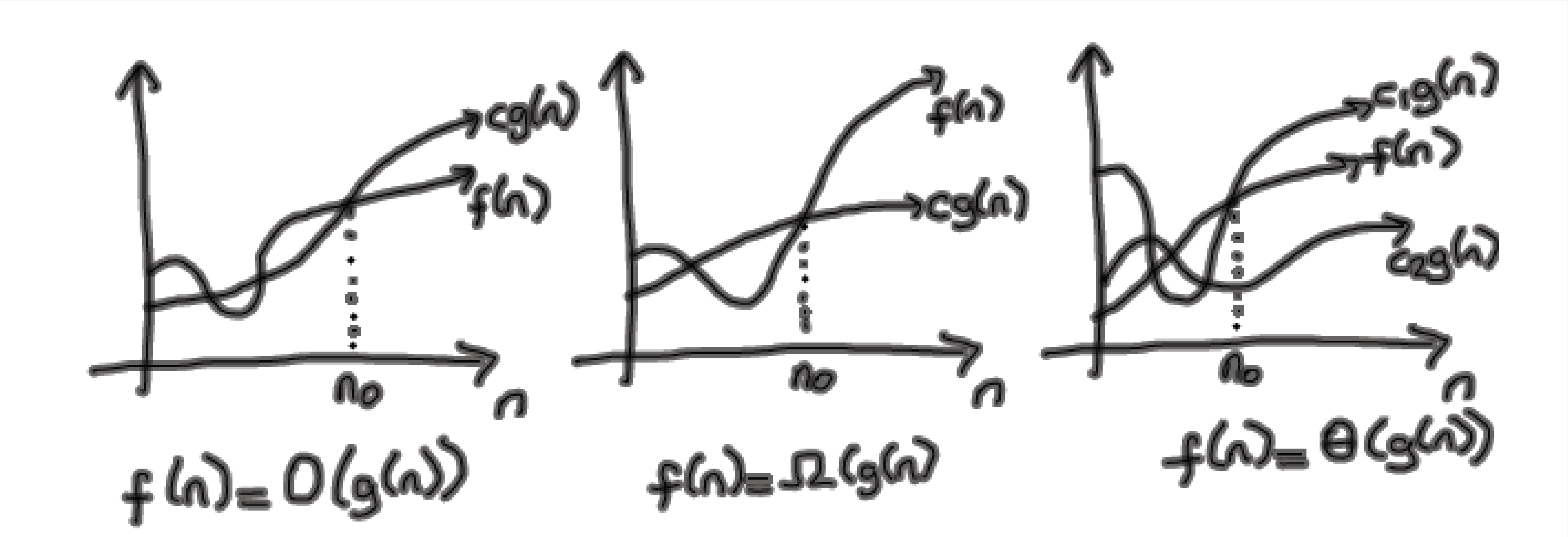
$\Theta$ Notasyonu (Sıkı Sınır, Ortalama durum)

**Kısaltmalar ve Anlamları**

| $O$ | $\Omega$ | $\Theta$ |
|---|---|---|
| $\leq$ | $\geq$ | $=$ |

$\Theta(n^2)$ algoritması aynı problemi çözen bir $\Theta(n^3)$ algoritmasından daha hızlıdır.

# Asimtotik Notasyonların Grafik Üzerinde Gösterimi




*) Asimtotik analiz $n \to \infty$ iken fonksiyonun büyümesi ile ilgilidir.

# Big *O* Formal Tanımı

**Tanım:** $f(n) \in O(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$'in büyüme sabit bir sayı ile çarpımının büyüme derecesinden küçüktür.

$$f(n) \leq c\, g(n)\ ,\ \ \forall\ \ n \geq n_0$$

Eşitsizliğini sağlayan pozitif bir sabit c ve pozitif bir tamsayı $n_0$ vardır.

--- O notasyonu ilk olarak alman Matematikçi Bochmann tarafından 1894 yılında tanımlanmıştır.

--- Algoritmanın en kötü durum analizini yapmak için kullanılan notasyondur.

# Örnek:

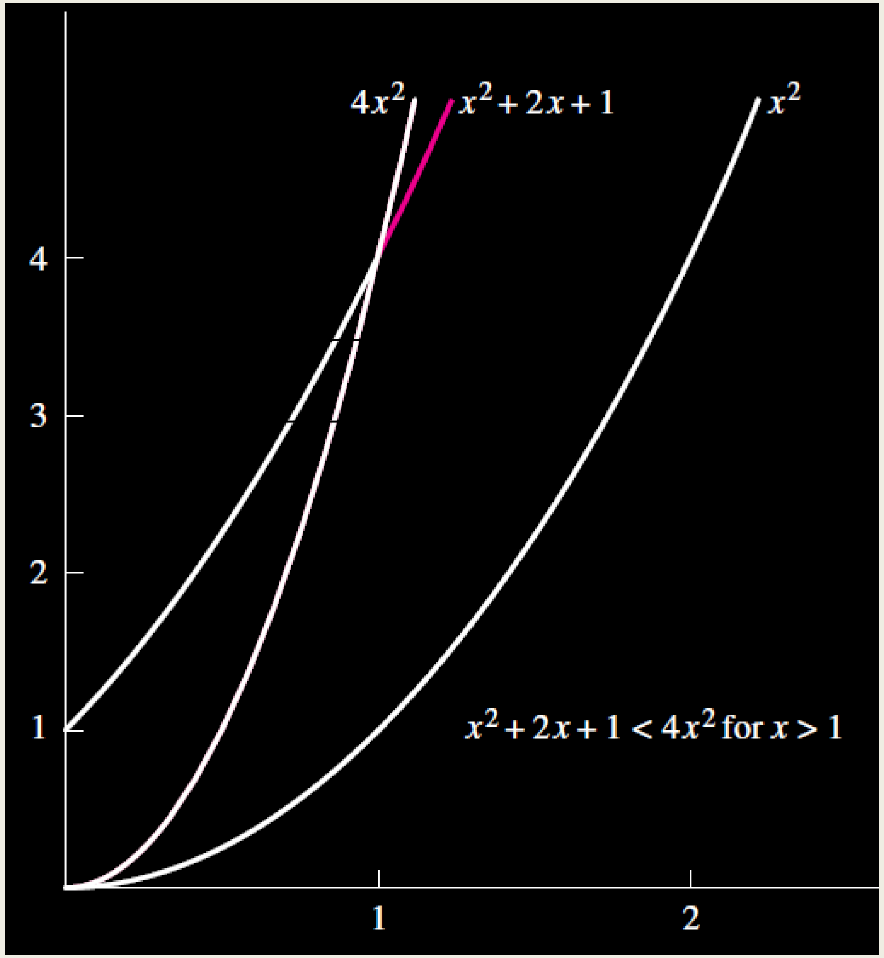
- $f(x) = x^2 + 2x + 1$ ise $O(x^2)$ dir.
- Eğer $x>1$ ise $x<x^2$ dir ve, eğer $x>1$ ise $1<x^2$ dir.
- Çünkü:

$$0 \leq x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2 = 4x^2$$

$x>1$ olduğu her değer için

- Böylece, $c = 4$ ve $n_0 = 1$ **alınırsa** $f(x) \in O(x^2)$ elde edilir.

# Grafiği



## Örnek:

$T(n)=3n+8 = O(n)$ dir. ( $3n+8 \in O(n)$ )

$3n+8 <= c*n$

$n>=1$ için $3n+8<=3n+8n<=11n$ ($c=11$, $n_0 =1$)

$n>=4$ için $3n+8<= 5n$ ($c=5$, $n_0 =4$)

Diğer c ve $n_0$ değerleri bulunabilir.

## Örnek:

$T(n)=3n+8= O(n^2)$ dir. ( $3n+8 \in O(n^2)$ )

$3n+8<= c*n^2$

$n>=5$ için $3n+8<= c * n^2$ ($c=1$, $n_0 =5$)

$n>=3$ için $3n+8<= c * n^2$ ($c=2$, $n_0 =3$)

Diğer c ve $n_0$ değerleri bulunabilir.

# Bazı Önemli Fonksiyonların Büyüme Dereceleri

- Tüm logaritmik fonksiyonlar $\log_a n$ aynı asimptotik sınıfa sahiptir.

  $O(\log n)$ logaritmanın tabanı $a > 1$ önemli değil.

- Aynı derece *k'ye sahip olan tüm polinomlar* aynı asimptotik sınıfa sahiptir. :
- $a_k n^k + a_{k-1} n^{k-1} + \ldots + a_0 \in O(n^k)$

- Üstel fonksiyonlar $a^n$, a değerine göre farklı büyüme sınıfına aittir.

- order $\log n$ < order $n^\alpha$ $(\alpha>0)$ < order $a^n$ < order $n!$ < order $n^n$

# Temel Asimptotik Verimlilik Sınıfları

| | |
|---|---|
| **1** | **sabit** |
| **log *n*** | **logaritmik** |
| ***n*** | **lineer** |
| ***n* log *n*** | ***n*-log-*n* or linerritmetik** |
| $n^2$ | **quadratic** |
| $n^3$ | **kübik** |
| $2^n$ | **üstel** |
| ***n*!** | **faktoriyel** |

# $\Omega$- Formal Tanımı

- **Tanım:** $f(n) \in \Omega(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$'in sabit bir sayı ile çarpımının büyüme derecesinden büyük veya eşittir.

$$f(n) \geq c\, g(n)\ ,\ \forall\ n \geq n_0$$

Eşitsizliğini sağlayan pozitif bir sabit c ve pozitif bir tamsayı $n_0$ vardır.

Örn: T(n)=3n+5 $\Omega(n)$

3n+5 >= 3n, tüm n>=1 için sağlanır (c=3, $n_0$=1)

Örnek: $\underbrace{n}_{f(n)} = \Omega(\underbrace{\lg n}_{g(n)})$ 'dir.

$\boxed{\log_2 n = \lg n}$

$c.g(n) \leq f(n)$

$1.\lg n \leq n$

$n=2$ için $1 \leq 2$

$c=1$ ve $\forall n \geq n_0$ için, $c.gn \leq f(n)$

olduğundan $f(n) = \Omega(g(n))$ 'dir.

$\boxed{n = \Omega(\lg n)}$ 'dir.

Çalışma Sorusu: $n^3 = \Omega(n^2)$ ?

# $\Theta$ Formal Tanımı

- **Tanım:** $f(n) \in \Theta(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$ fonksiyonun bir sabit katından yüksek aynı zamanda $g(n)$ fonksiyonun bir sabit katından da düşük olmaktadır.

$$c_1 g(n) \leq f(n) \leq c_2 g(n), \quad \forall \; n \geq n_0$$

Eşitsizliğini sağlayan pozitif sabit $c_1$, $c_2$ *sayıları* ve pozitif bir tamsayı $n_0$ vardır.

$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$ 'dir.

Hem alttan, hem de üstten sınırlıdır.

$$5n^2 + n - 7 = \Theta(n^2) \text{ 'dir}$$

*)

$\Theta$ notasyonunda küçük terimler ihmal edilir

**Örnek:** $\frac{1}{2}n^2 - 2n \in \Theta(n^2)$ olduğunu gösteriniz.

$f(n) = \frac{1}{2}n^2 - 2n \quad g(n) = n^2$ dir.

$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n) \quad \forall n \geq n_0$

$c_1, c_2 > 0$

$c_1 < c_2$

$\frac{1}{4} \cdot n^2 \leq \frac{1}{2}n^2 - 2n \leq \frac{1}{2} \cdot n^2 \Rightarrow \forall n$ için doğrudur.

$\frac{1}{4}n_0^2 = \frac{1}{2}n_0^2 - 2n_0$

$2n_0 = \frac{1}{4}n_0^2$

$\boxed{n_0 = 8} \Rightarrow$ eşik değeri.

Böylece: $c_2 = \frac{1}{2} > 0$

$n_0 = 8$ için

$c_1 = \frac{1}{4} > 0$

$\frac{1}{2} n^2 - 2n \in \Theta(n^2)$ dir.

$\Theta(n)$ $n_0 = 9$ için

$\frac{1}{4} \cdot 9^2 \leq \frac{1}{2} \cdot 9^2 - 2 \cdot 9 \leq \frac{1}{2} \cdot 9^2$

$\frac{81}{4} \leq \frac{45}{2} \leq \frac{81}{2}$

# Önemli Problem Tipleri

- Sorting (Sıralama)
- Searching (Arama)
- String Processing (String İşleme)
- Graph Problems (Graph Problemleri)
- Combinatorial Problems (Kombinasyon Problemleri)
- Geometric Problems (Geometrik Problemler)
- Numerical Problems (Nümerik Problemler)

# Algoritmaların Zaman Karmaşıklığı

- Best Case (En iyi durum)
- Worst Case (En Kötü Durum)
- Average Case (Ortalamada)

# Arama Algoritması

## Problem tanımı:

- Bir listede aranan 'key''i bulmak.

## Lineer Arama (Sequential Search)

### Yaklaşım

1. ***Birinci elemanla karşılaştır***
2. ***Eğer aranan eleman ise sonucu belirt.***
3. ***Aksi halde bir sonraki elemana geç.***
4. ***Eğer liste sonuna geldiysen aranan elemanın bulunamadığı belirt.***

# Lineer Arama Algoritması

## Sözde Kod

*A[1], … , A[n] ve aranacak eleman verilsin.*

```
1. i=1
2. while i ≤ n
3.     if A[i] = key return i
4.     else i = i+1
5. return fail
```

en iyi, en kötü ve ortalama durumlar nelerdir ?

# Lineer Arama Algoritması Analizi

**Algoritmanın Zaman Karmaşıklığı:**

- Best case (En iyi Durum)

  A[1] = key ( O(1) ) (İlk elemanın aranan eleman olması)

- Worst case (En kötü Durum)

  A[i] $\neq$ key Herhangi bir key değeri için

  - En son elemanın aranan eleman olmasıdır.
  - En kötü durumda karmaşıklık ( O(n) )

- Average case (Ortalama Durum)
  - Aranan elemanın ortalarda olması~ n/2 ( O(n) )

# İki önemli Sıralama Algoritması ve Analizleri

## 1-) Eklemeli Sıralama (Insertion Sort)

- Hedef
  - *Verilen A[1] … A[n], listesini sıralamak*

- Yaklaşım (j. index açısından bakıldığında)

1. ***A[1] sıralıdır.***
2. ***A[1] ≤ A[2] … ≤ A[j-1] olduğunu farzedelim.***
3. ***Bir sonraki öğeyi alın ve uygun yerine yerleştirin***

# Eklemeli Sıralama

- <u>Sözde Kod</u>

```
for j=2 to n
    key = A[j]
    i = j-1
    while i>0 and A[i]>key
        A[i+1] = A[i]
        i = i-1
    A[i+1] = key
```

# Örnek:

| | A[1] | A[2] | A[3] | A[4] | A[5] | A[6] |
|---|---|---|---|---|---|---|
| Girdi: | 8 | 2 | 4 | 9 | 3 | 6 |


j=2   1>0   8>2

key=2   8 8 4 9 3 6

i=1

i=0   2 8 4 9 3 6   j=2 için

---

j=3   2>0 ?   A[2] > key (8 > 4) ✓

A[3]=4=key

i=2   2 8 8 9 3 6

i=1   1>0   A[1] > key   2>4 ✗   whileden çıktı

$A[i+1] = A[2] = key$

$\leftarrow 4$

$2 \quad 4 \quad 8 \quad 9 \quad 3 \quad 6$

j=3 için son durum.

j=4 için

key=9

i=3

$3 > 0 \quad \overset{8}{A[3]} > \overset{9}{key}$ ✕ while çalışmadı

$2 \quad 4 \quad 8 \quad 9 \quad 3 \quad 6$

j=4 durumda

2 3 4 8 9 | 6

2 3 4 6 8 9

# Eklemeli Sıralama

- **Analizi**

| Adım | Maliyet | Zaman |
|---|---|---|
| 1 | $c_1$ | 2n |
| 2 | $c_2$ | n-1 |
| 3 | $c_3$ | n-1 |
| 4 | $c_4$ | $\sum_{j=2}^{n} t_j$ |
| 5 | $c_5$ | $\sum_{j=2}^{n} (t_j - 1)$ |
| 6 | $c_6$ | $\sum_{j=2}^{n} (t_j - 1)$ |
| 7 | $c_7$ | n-1 |

$t_j$ = O satırın j değeri için kaç kez çalıştığını gösterir.

- **Sözde Kod**

```
1.  for j=2 to n
2.      key = A[j]
3.      i = j-1
4.      while i>0 and A[i]>key
5.          A[i+1] = A[i]
6.          i = i-1
7.      A[i+1] = key
```

# Eklemeli Sıralama

$t_j$ problemin örneğine göre değişmektedir (girdi)

**En İyi Durum**

A sıralı olduğunda

$t_j$=0 for all j

T(n) = lineer fonksiyon ( $T(n) = an + b$ )

= O(n)

# Eklemeli Sıralama

<u>En Kötü Durum</u>

A tersten sıralı olduğunda

$t_j = j$ *for* $j=2,\ldots,n$

$$\sum_{j=1}^{n} t_j = \sum_{j=1}^{n} j = \frac{n(n+1)}{2}$$

T(n) = quadratic function ( $T(n) = an^2 + bn + c$ )

= O($n^2$)

# Eklemeli Sıralama

## Ortalama Durum

Dizi rastgele bir şekilde oluşturulmuş olsun.

Hangi adımda A dizisi sıralı olacaktır?

Eğer:

$$t_j \approx \frac{j}{2}$$

$$\sum_{j=1}^{n} t_j = \sum_{j=1}^{n} \frac{j}{2} = \frac{1}{4}(n^2 + n)$$

T(n) → quadratic

# 2-) Birleştirmeli Sıralama (Merge-Sort)

- <u>Hedef</u>
  - *Verilen A[1] ... A[n], listesini sıralamak*

- <u>Yaklaşım (j. index açısından bakıldığında)</u>

  ***Diziyi sürekli ikiye bölüp bölünen parçaları sıralayıp en son birleştirmek. (Böl ve Fethet Yöntemi…)***

(n) 11 7 8 1 5 6 3 17 12

n=1 oluncaya kadar böl

11 7 8 1 5 6 3 17 12

7 11 | 8

11 8 7 | 5 1

6 3 | 17 12

1 5 7 8 11 | 3 6 12 17

Birleştirme

sıralıyor) 1 3 5 6 7 8 11 12 17

# Algoritma Analizi

$T(n)$ Merge-Sort $(A[1,\dots,n])$

$\Theta(1)$ if $n=1$ return

$T(\frac{n}{2})$ Merge-Sort $(A[1\dots,\lceil\frac{n}{2}\rceil])$

$T(\frac{n}{2})$ Merge-Sort $(A[\lceil\frac{n}{2}\rceil+1\dots n])$

$\Theta(n)$ 2 sıralı listeyi birleştir.

kapsar

$T(n)=2.T(\frac{n}{2})+\Theta(n)+\cancel{\Theta(1)}$

$\boxed{T(n)=2.T(\frac{n}{2})+\Theta(n)}$

özyineleme bağıntısı

recurrence relation

yürütme zamanı

? Özyineleme ağacı

- Master(Ana) Teorem

Yerine koyma metodu

$T(n)=\Theta(n.\log n)$

4. dönem Algoritmalar dersinde nasıl elde edildiği öğretilecek.

| Insertion Sort | Merge Sort |
| --- | --- |
| $T(n) = \Theta(n^2)$ | $T(n) = \Theta(n.\log n)$ |
| * iteratif Algoritma | * Rekürsif Algoritma |
| * Ayrı dizi üzerinde gezer. | * Geçici dizi kullanır. |

| $n^2$ | $n.\log n$ | ? |
| --- | --- | --- |

n=100 için

10000 665

Merge Sort algoritması insertion sort'a göre daha hızlıdır.

# Listeler ve Sayma (Kombinatorik)

Tanım: Kombinatorik nesnelerin düzeninin incelenmesidir. Kombinatoriğin en önemli alanı, belirli özelliğe sahip nesnelerin sayılmasıdır.
(Örnek: n elemanlı bir kümeden m elemanlı bir başka kümeye yazılabilecek örten fonksiyonların adedi)
Uygulamalar:
Algoritmaların çalışma zamanlarının analizi (Karmaşıklık)
Şans Oyunları
İsteği karşılamak için yeterli IP adresi var mı?
7 karakterli kaç adet şifre olabilir?
**Örnek:** 3 harfi takip eden 3 rakam bulunan altı karakterli kaç adet ruhsat plakası olabilir? (Ğ harfi kullanılmadı…)
Çözüm: 28.28.28.10.10.10 = 24.389.000 adet.

Listeler ve Sayma

S sonlu bir küme olsun.

En az bir özalt kümesine birebir eşleme yoksa bu kümeye sonlu küme denir.

Eğer S'nin elemanları

1. eleman $S_1$
2. " $S_2$
⋮ ⋮
n. " $S_n$

şeklinde işaretlenebiliyorsa S kümesine numaralanabilir küme denir.

Elemanlarının sıralanmış dizilerine **liste**

Örn. $(1,2,7) \rightarrow 3$ uzunluklu liste

$(3,7,2,6) \rightarrow 4$ " "

$(3,5) \rightarrow 2$ " "

(Sıralı ikili)

$$(2,5) = (x+2,\ y-3) \Rightarrow \quad x+2=2,\ \boxed{x=0} \qquad 5=y-3,\ \boxed{y=8}$$

Ex) $S = \{1, 2, 3\}$ olsun.

2'li listeleri yazınız.
uzunluklu

$\{(1,2), (1,3), (1,1), (2,1), (2,2), (2,3), (3,1),$
$(3,2), (3,3)\}$

9 tane 2 uzun. liste vardır.

n elemanlı bir kümenin 2-uzunluk liste sayısı : $n^2$

" " " " 3 " " " $= n^3$

$\vdots$

n " " " k - " " " $= n^k$

(B) Eğer listelerde aynı eleman yoksa (elemanları tekrarsız) kaç tane liste yazılır.

$n^2 - n = n.(n-1) \Rightarrow$ 2'li liste

$(1,2,3), (\cancel{1,3},2)$

3'lü liste $\Rightarrow \underbrace{n.(n-1).(n-2)}_{3 \text{ tane}}$

⋮

k'lı liste $\Rightarrow \underbrace{n.(n-1).(n-2).(n-3).\,---(n-k+1)}_{k \text{ tane}}$

$A = \{1,2,3,4,5,6\}$

kümesini (i) 4 uzunluklu liste sayısı = $6^4$

(ii) 4 " elemanları tekrarsız = $n.(n-1).(n-2).(n-3)$

$n=6 \quad k=4$

$n-k+1 = 3$

$= 6.5.4.3$

$= 360$

## Çarpım Listesi:

1. elemanın $n$

2. elemanının $m$ farklı

ortamdan seçildiği 2'li listelerin sayısı $\boxed{n \cdot m}$ dir.

Örn: Ad ve soyadların ilk harfleriyle kaç farklı kodlama yapılır? 29 harf (Ğ almayalım)

$\overline{|(28 * 28)|}$ farklı kodlama yapılır.

28 * 27 => harfler tekrarsız.

Örn: 0'dan 9'a kadar rakamların ve 29 harfin kullanıldığı bir ortamda;

i) 1 rakam 1 harf şeklinde kaç kodlama $= 10 \times 29$

ii) harf harf harf rakam $= 29 \times 29 \times 29 \times 10$

Örn: 1'den 49'a kadar sayıların bulunduğu bir ortamda 6 uzunluklu liste sayısı nedir? $49^6$

rakamları tekrarsız $= \boxed{49 \cdot 48 \cdot 47 \cdot 46 \cdot 45 \cdot 44}$

(ör) 11

29 harfin 3 tanesini kullanarak havaalanlarına isimler veriliyor. Kaç farklı isim verilir.

$29^3 \Rightarrow$ tekrarlı

$29 \cdot 28 \cdot 27 \Rightarrow$ tekrarsız.

(ör) 11

11 rakam kullanılarak ilk rakamı 0 ve 1 olmayan kaç tane telefon numarası vardır?

$$8 \; \underbrace{10 \; 10 \; 10 \; \ldots \; 10}_{10 \text{ uzunluklu}} = \boxed{8 \cdot 10^{10}}$$

2,3,4,5,6, 7,8,9

10 uzunluklu (10 tane eleman var)

liste uzunluğu

rakamları

v) 9 rakamlı S.S.K numaralarının oluşturulduğu bir ortamda;

i) Kaç tane sayı yazılır: $10^9$

ii) " " çift sayı yazılır: $10^8 \cdot \boxed{5} \rightarrow \{0,2,4,6,8\}$

iii) Kaç tanesinin tüm rakamları çifttir: $5^9$

iv) Kaç tanesinin en az bir rakamı 8'dir: Tüm durum − 8'in olmadığı durumları

$\{0,1,2,3,4,5,6,7,\cancel{8},9\}$ $= \underbrace{10^9 - 9^9}$

9 eleman, 9 uzunluklu } $9^9$, sekiz yoktur.

En az bir tane 8 vardır.

vi) $\{2,5,7\} \Rightarrow$ kaç tane 5 uzunluklu liste yazılır? → $3^9$ tane.

**Teorem:** Tekrarsız $n$ elemanın tümünün kullanıldığı $n$'li listelerin sayısı $n!$'dir.

$$\underbrace{n.(n-1).(n-2)\ ----\ .(\cancel{n-n}+1)}_{n \text{ tane}} = n!$$

**Sayma Yolları:** 1. dalın $n_1$ farklı

2. dalın $n_2$ farklı

seçenekten seçildiği bir ortamda;

1. veya 2. dalın sayısı $= n_1+n_2 \Rightarrow$ toplama yoluyla

1. ve 2. " sayısı $= n_1 \cdot n_2 \Rightarrow$ çarpma "

sayma denir.

1. a) 7 bluz, 5 etek, 9 elbisesi olan bayan kaç farklı şekilde giyinir.

$7 \cdot 5 + 9 = 44$ farklı giysi giyebilir.

1. b)


A'dan C'ye kaç farklı şekilde gidilir?

$4 \cdot 3 + 2 = 14$

$(A \to B$ ve $B \to C)$ veya

Örn) Bir bilgisayar işletim sisteminde A'dan Z'ye 29 harf ve 0'dan 9'a kadar rakamlar kullanılarak en çok 8 uzunluklu ilk elemanı bir harf olan harflerden veya rakamlardan oluşan ve önce harfler sonra rakamlar gelecek şekilde kaç farklı dosya ismi yazılır?

$c\,\underbrace{adt}_{3h}\,\underbrace{0024}_{4r}$

~~d1296~~

~~a1b2d~~

a235 ✓

at110 ✓

8 uzunluklu

harf

$\boxed{29}$ _ _ _ _ _ _ _

$\underbrace{\qquad\qquad\qquad\qquad}_{7 \text{ eleman}}$

7 harf 0 rakam : $29^7 \cdot 10^0$
6 " 1 " : $29^6 \cdot 10^1$
5 " 2 " : $29^5 \cdot 10^2$
4 " 3 " : $29^4 \cdot 10^3$
3 " 4 " : $29^3 \cdot 10^4$
2 " 5 " : $29^2 \cdot 10^5$
1 " 6 " : $29^1 \cdot 10^6$
0 " 7 " : $29^0 \cdot 10^7$

8 uzunluklu $= 29^8 + 29^7 \cdot 10^1 + 29^6 \cdot 10^2 + 29^5 \cdot 10^3 + 29^4 \cdot 10^4$
$29^3 \cdot 10^5 + 29^2 \cdot 10^6 + 29^1 \cdot 10^7$

7 uzunluklu $= 29^7 + 29^6 \cdot 10^1 + 29^5 \cdot 10^2 + 29^4 \cdot 10^3 + 29^3 \cdot 10^4$

$29^2 \cdot 10^5 + 29^1 \cdot 10^6$

ilk eleman harf.

⋮

1 uzunluk $= 29^1$ (Sadece harf olabilir.)

Tanım (Permütasyon)

$r, n \in \mathbb{N}$ ve $r \leq n$ olmak üzere;

$n$ elemanlı bir kümenin birbirinden farklı $r$ elemanının her bir dizilişine $A$ kümesinin $r$'li permütasyonu

denir, ve $P(n,r) = \dfrac{n!}{(n-r)!}$ olarak tanımlanır.

Örn) $A = \{a, b, c\}$ 2'li permütasyonlarının sayısı

$$P(3,2) = \frac{3!}{(3-2)!} = 3! = 6$$

$(a,b), (a,c), (b,a), (c,a), (b,c), (c,b)$

6 tane

(e) iii)

$A = \{a, b, c, d, e, f\}$ kümesinin 4'lü permütasyonlarının kaç tanesinde;

(i) c bulunur $\Rightarrow$

(ii) c bulunmaz, d bulunur

(iii) a veya c bulunur

1. yol

Tümdurum − c'nin olmadıkları

$= P(6,4) - P(5,4)$

$= 360 - 120 = 240$

2. yol

$\frac{1}{c}$ _ _ _ (3 uzunluklu)

$P(5,3) = \frac{5!}{2!} = 60$

_ $\frac{1}{c}$ _ _ $\Rightarrow 60$

_ _ $\frac{1}{c}$ _ $\Rightarrow 60$

_ _ _ $\frac{1}{c}$ $\Rightarrow 60$

240

ii) c bulunmaz ?

d bulunur $= P(5,4) - P(4,4)$

a, b, ~~c~~, d, e, f → c yok, d olabilir; d yok, c yok; a, b, ~~c~~, ~~d~~, e, f

$= 120 - 24 = 96$

iii) = Tüm durum − a ve c yok ~~a~~, b, ~~c~~, d, f, e

$= P(6,4) - P(4,4)$ 4 tane

$= 360 - 24 = 336$

## Tekrarlı permütasyon

$n_1$ tanesi 1. cinsten, $n_2$ tanesi 2. cinsten, ---, $n_r$ tanesi r. cinsten

ve $n_1+n_2+\dots+n_r=n$ olmak üzere;

n tane elemanın birbirinden farklı sıralanışı; $\dfrac{n!}{n_1!\,n_2!\cdots n_r!}$

Ör: 111 0052 sayısının rakamları ile 7 basamaklı

a) kaç sayı yazılır.

b) kaç çift sayı yazılır.

c) $2 \cdot 10^6$'dan büyük kaç sayı yazılır.

a) $\dfrac{7!}{3!\cdot 2!\cdot 1!\cdot 1!} = 420$ → 0 ile başlayabilirler.

$\dfrac{6!}{3!\ 1!\cdot 1!\cdot 1!} = 120$ → 0 ile başlıyorlar.

$420 - 120 = 300$

b) 0 ile bitenler $\dfrac{6!}{3!} = 120$

0 ile başlayıp 0 ile biten $\dfrac{5!}{3!} = 20$

$120 - 20 = \underline{\underline{100}}$ sonu sıfır olan.

2 ile bitenler $\dfrac{6!}{3!\cdot 2!} = 60$

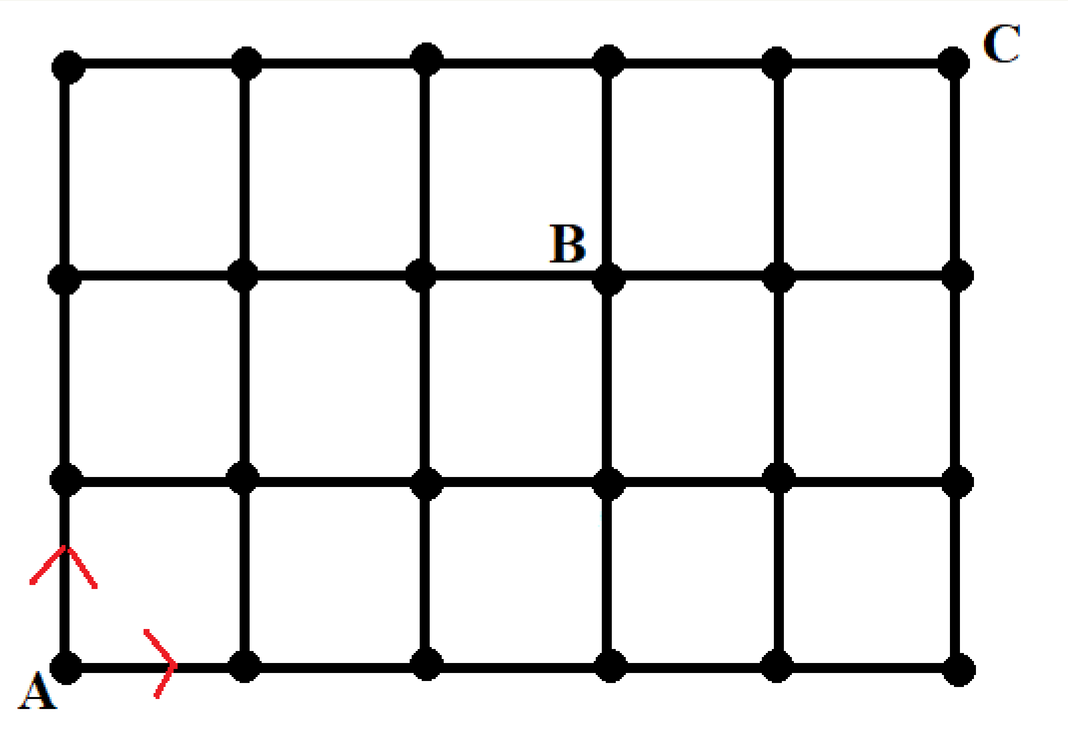
0 ile başlayıp 2 ile biten $\dfrac{5!}{3!} = 20$

$60 - 20 = \underline{\underline{40}}$ 2 ile biten

$\Rightarrow 100 + 40 = 140$ çift olanlar.

c) $2.10^6$ 'dan büyük olması için milyonlar basamağında 2 veya 5 olmalıdır. $420 \cdot \dfrac{2}{7} = 120$

## Örnek:




A dan başlayıp yukarı veya sağa giderek:

a) A dan C ye kaç farklı yol vardır?

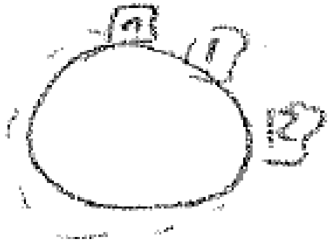
b) A dan C ye, B ye uğrayarak kaç farklı yol vardır?

## Çözüm:

a) $\dfrac{8!}{5!\cdot 3!} = \dfrac{8\cdot 7\cdot \cancel{6}\cdot \cancel{5!}}{\cancel{5!}\cdot \cancel{3}\cdot\cancel{2}} = 56$

b) $\dfrac{5!}{3!\cdot 2!} \cdot \dfrac{3!}{2!\cdot 1!} = 30$

## Dönel Permütasyon



$(n-1)!$

Örn: 5 mat, 3 fiz, 2 kimya öğretmeni yuvarlak bir masa etrafına,

a) kaç farklı şekilde? ⟶ $9!$

b) aynı branştakiler kaç farklı şekilde? $2! \cdot 5! \cdot 3! \cdot 2!$

# Permütasyon Algoritması ve Analizi

```c
#include<stdio.h>
#include<conio.h>
int faktoriyel(int a)
{ int i,carp;
carp=1;
for(i=1;i<=a;i++)
carp=carp*i;
return carp;
}
int main()
{ int n,r,sonuc;
printf("n degerini giriniz: ");
scanf("%d",&n);
printf("r degerini giriniz: ");
scanf("%d",&r);
sonuc=faktoriyel(n)/faktoriyel(n-r);
printf("permutasyon degeri= %d",sonuc);
getch ();
return 0;
}
```

```c
int faktoriyel(int a)
{ int i,carp;          → 1+1
carp=1;                → 1
for(i=1;i<=a;i++)      → 1+a+1+a
carp=carp*i;           → a
return carp;           → 1
}
```

→ 3a+6

main:

- { int n,r,sonuc; → 1+1+1
- printf("n degerini giriniz: "); → 1
- scanf("%d",&n); → 1
- printf("r degerini giriniz: "); → 1
- scanf("%d",&r); → 1
- sonuc=faktoriyel(n)/faktoriyel(n-r); → 1 atama ve a=n için 2 kere (en kötü durum) 6n+13
- printf("permutasyon degeri= %d",sonuc); → 1
- getch (); → 1
- return 0; → 1

$T(n)=6n+23 \in O(n)$

## Güvercin yuvası(Pigeonhole) prensibi

--- Basit ve açık bir prensip olan güvercin yuvası prensibi, kombinatorik teoride oldukça çok kullanılır. Güvercin yuvası problemine en fazla bilgisayar bilimlerinde karşılaşılır. Örneğin, olası anahtar sayısı dizideki indis sayısını geçtiği için çarpışmalar özet fonksiyonlarında kaçınılmazdır. Bu çarpışmalardan kaçınabilen akıllı bir özet fonksiyonu yoktur. Bu prensip aynı zamanda en az bir dosyayı kısaltırken diğer bir dosyayı uzatan kayıpsız sıkıştırma algoritmasını sağlar.

--- 10 fakülte elemanının yerleşeceği 9 oda olduğu durum da yerleşim, 19 fakülte elemanının olduğu durumdaki oda paylaşımı vs. gibi durumlardaki çözümler güvercin yuvası prensibi ile bulunur.

— **Prensip basit şekliyle,** Dirichlet güvercin yuvası prensibi olarak bilinir ve eğer n+1 güvercin n adet yuvaya yerleştirilecek olursa en az bir yuvada birden fazla güvercin olacaktır. Tersi eğer n güvercin n+1 yuvaya koyulacak olursa en az bir yuva boş kalacaktır.

— **Prensibin daha genel hali ise**, eğer k.n+1 veya daha fazla güvercin, n yuvaya konulacak olursa, en az bir yuvada k dan fazla güvercin olacaktır.

Örnek: Bir kampüste 18 adet oturma salonu bulunmaktadır. Öğrenci sorumlusu, bir salondaki bilgisayar kullanımını anlamak için anket yapmak amacıyla seçeceği salondan 5 kişilik öğrenci komitesi oluşturmak ister ve salonlara duyurular asar. En az kaç kişi bu ankete cevap vermelidir ki, sorumlu bir salon seçip komite oluşturabilsin?

Çözüm: Genel güvercin yuvası prensibi gereği, $k=4$ olur ve $k.n+1 = 4.18 +1= 73$

# Kaynaklar


- *Discrete Mathematics and Its Applications*, Kennet H. Rosen

(Ayrık Matematik ve Uygulamaları, Kennet H. Rosen (Türkçe çeviri), Palme yayıncılık)

- *Discrete Mathematics: Elementary and Beyond*, L. Lovász, J. Pelikán, K. Vesztergombi, 2003.

- *Introduction to Algorithms*, T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, 2009.

- *Introduction To Design And Analysis Of Algorithms*, A. Levitin, 2008.