

CENG 114 BİLGİSAYAR BİLİMLERİ İÇİN AYRIK YAPILAR

Doç. 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:

Bir dizinin elemanlarını toplama

<pre>top = 0 for (i=0; i<n; i++){ top = top + a[i]} print top</pre>	İşlem sayısı
	1
	$1+n+1+n$
	n
	1
	$3n+4$

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

Tablo Metodu

■ Örnek:

Matris Toplama

a, b, c 'nin $m \times n$ boyutunda matrisler olduğunu varsayalım.

<pre>for (i=0; i<m; i++) { for (j=0; j<n; j++) { c[i,j] = a[i,j] + b[i,j] } }</pre>	işlem	toplam
	$1+m+1+m$	$2m+2$
	$m(1+n+1+n)$	$2mn+2m$
	mn	mn
		$3mn+4m+2$

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

Asimptotik Notasyon Ve Temel Verimlilik Sınıfları

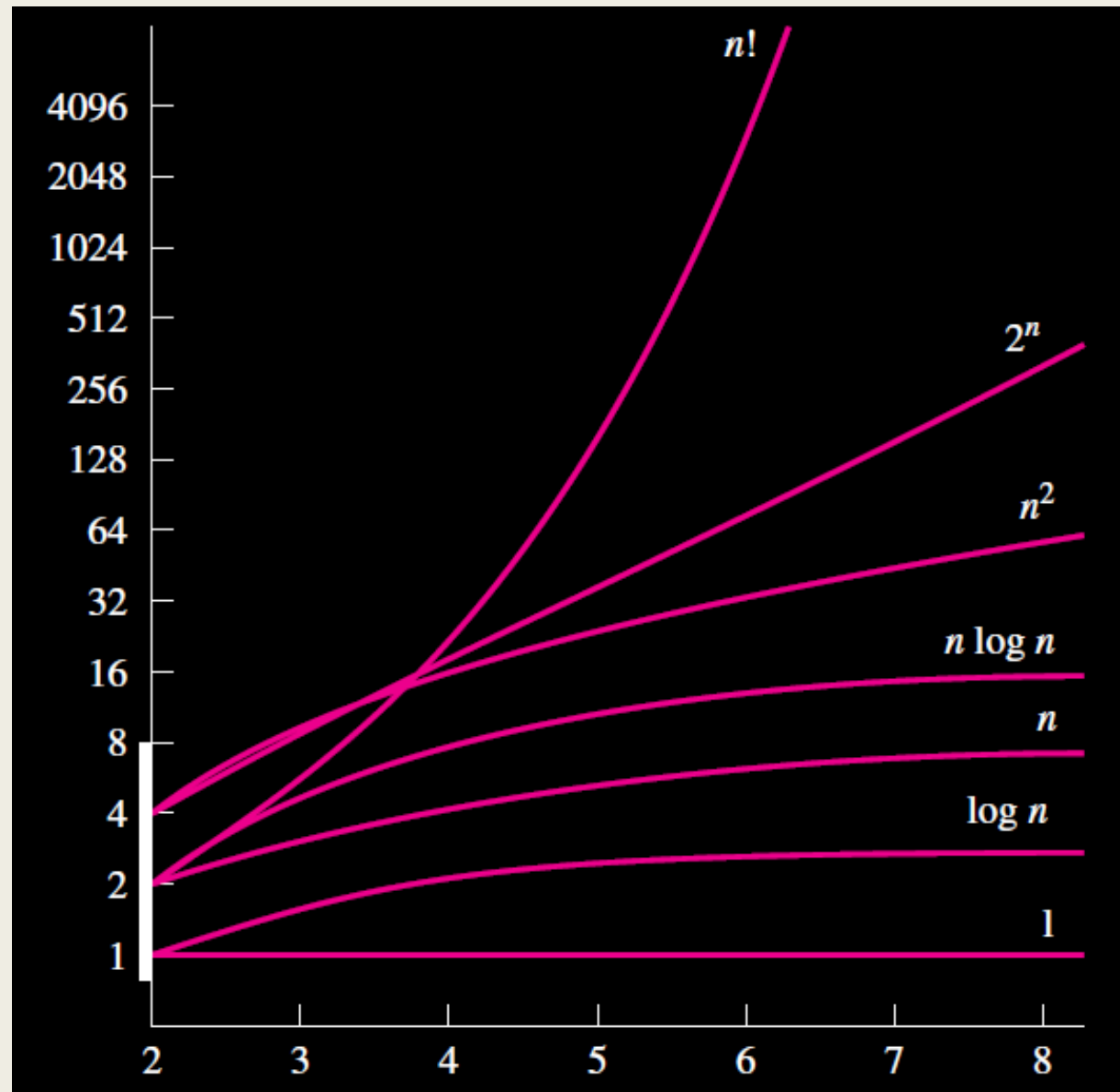
(Büyüme Sırası) Order of growth

- 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?*

$n \rightarrow \infty$ 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}		



Asimptotik Büyüme Dereceleri

Asimptotik Analiz

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

Asimptotik Notasyon

O Büyükle Ω Büyükle Θ Teta

(Big O) (Big Omega) (Theta)

O Notasyonu (Üst Sınır)

Ω Notasyonu (Alt Sınır)

Θ Notasyonu (Sıkı Sınır, Ortalama durum)

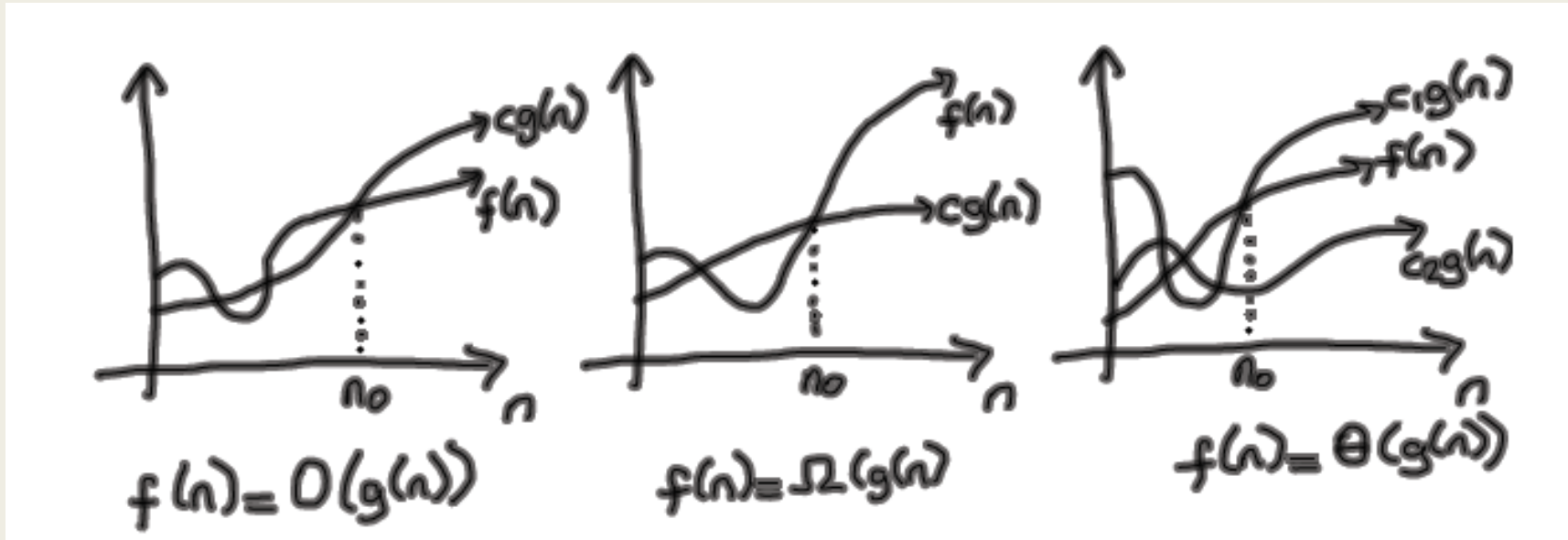
Kısaltmalar ve Anlamları

\leq \geq $=$



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

Asimtotik Notasyonların Grafik Üzerinde Gösterimi



~~*~~) Asimtotik analiz $n \rightarrow \infty$ 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.

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

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.

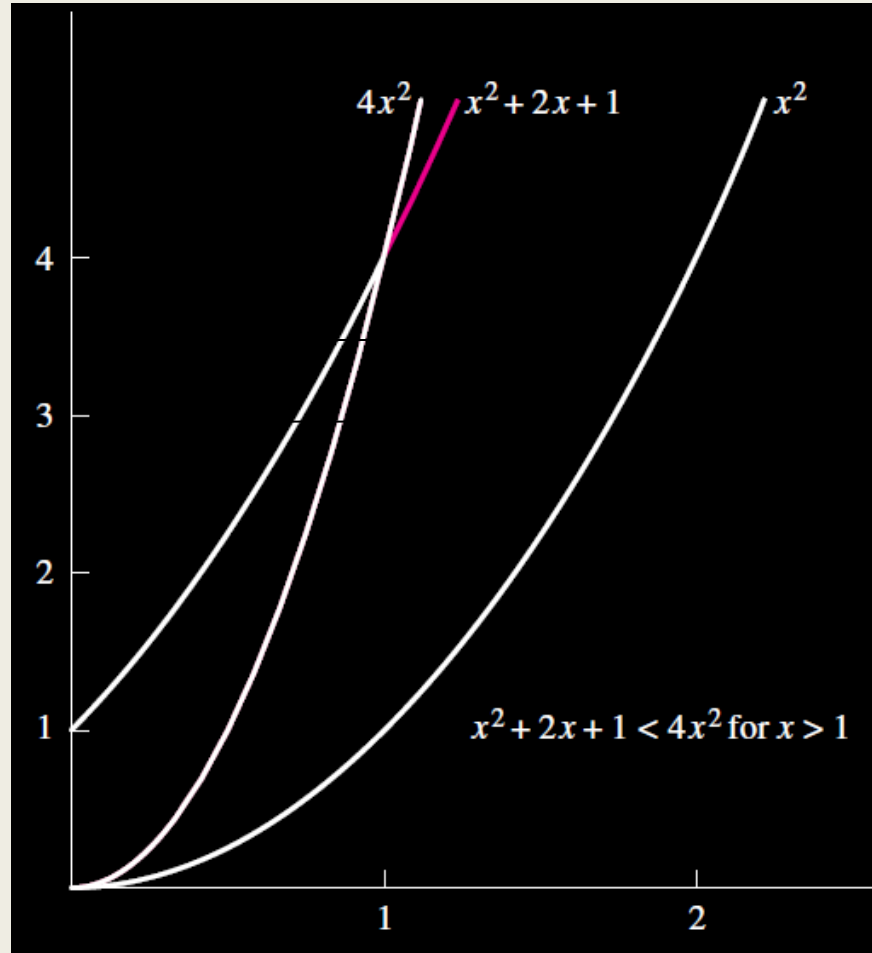
- Çünkü:

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

$x > 1$ 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. $(3n+8 \in O(n))$

$$3n+8 \leq c \cdot n$$

$n \geq 1$ için $3n+8 \leq 3n+8n \leq 11n$ ($c=11, n_0=1$)

$n \geq 4$ için $3n+8 \leq 5n$ ($c=5, n_0=4$)

Diğer c ve n_0 değerleri bulunabilir.

Örnek:

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

$$3n+8 \leq c \cdot n^2$$

$n \geq 5$ için $3n+8 \leq c \cdot n^2$ ($c=1, n_0=5$)

$n \geq 3$ için $3n+8 \leq c \cdot n^2$ ($c=2, n_0=3$)

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. :

- $a_k n^k + a_{k-1} n^{k-1} + \dots + a_0 \in O(n^k)$

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

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

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

Ω - 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.

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

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

Örn: $T(n)=3n+5 \in \Omega(n)$

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

Örnek: $n = \Omega(\lg n)$ 'dir.

$f(n)$ $g(n)$

$$\log_2^n = \lg n$$

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

$$1 \cdot \lg n \leq n$$

$$n \geq 2 \text{ için } 1 \leq 2 \leq n$$

$$c=1 \text{ ve } \forall n \geq n_0 \text{ için, } c \cdot g(n) \leq f(n)$$

$$\text{O halde } f(n) = \Omega(g(n)) \text{ 'dir.}$$

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

Çalışma Sorusu: $n^3 = \Omega(n^2)$?

⊖ 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.

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

★)

Θ notasyonu küçük terimler ihmal edilir

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

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

$$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n) \quad \begin{array}{l} \forall n \geq n_0 \\ c_1, c_2 > 0 \\ c_1 < c_2 \end{array}$$

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

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

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

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

Böylece: $c_2 = \frac{1}{2} > 0$ $n_0 = 8$ için
 $c_1 = \frac{1}{4} > 0$

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

② $n_0 = 9$ 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)

Farklı Problemler ve Bunların Analizi

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)

$A[1] = \text{key}$ ($O(1)$) (İlk elemanın aranan eleman olması)

■ Worst case (En kötü Durum)

$A[i] \neq \text{key}$ 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 $A[1] \dots A[n]$, listesini sıralamak

■ Yaklaşım (j. index açısından bakıldığında)

1. $A[1]$ sıralıdır.
2. $A[1] \leq A[2] \dots \leq A[j-1]$ olduğunu farzedelim.
3. Bir sonraki öğeyi alın ve uygun yerine yerleştirin

Eklemeli Sıralama

■ 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
```

Örnek:

AC[1] AC[2] AC[3] AC[4] AC[5] AC[6]

Girdi: 8 2 4 5 3 6

$j=2$
 $key=2$
 $i=1$
 $f=0$

1) $0 > 2$

8 8 4 5 3 6


2 8 4 5 3 6 $j=2$ için

$j=3$
 $AC[3]=4=key$
 $i=2$
 $i=1$

✓ $2 > 0?$ $A[2] > key$ L

2 8 8 5 3 6

1) $0 > 4$ $A[1] > key$ $2 > 4$ X while den çıktı



$$A[1] = A[2] = \text{key}$$

2 4 8 9 3 6 $j = 3$ ist zu klein.

$j = 4$ km
 $key = 9$
 $i = 3$

$3 > 0 \quad A[3] = 8 > 9 \quad \times \quad \text{while colliding}$

2 4 8 9 **3** 6 $j = 4$ swap

2 3 4 8 9 16]

2 3 4 6 8 9

Eklemeli Sıralama

■ Analizi

<u>Adım</u>	<u>Maliyet</u>	<u>Zaman</u>
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	$n-1$

■ 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
```

$t_j = 0$ satırın j değeri için kaç kez çalıştığını gösterir.

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 ($T(n) = an + b$)
 $= O(n)$

Ekleme Sıralama

En Kötü Durum

A tersten sıralı olduğunda

$$t_j = j \text{ for } j=2, \dots, n$$

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

$T(n)$ = 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:

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

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

$T(n) \rightarrow \text{quadratic}$

2-) Birleştirmeli Sıralama (Merge-Sort)

■ Hedef

- Verilen $A[1] \dots A[n]$, listesini sıralamak

■ Yaklaşım (j. index açısından bakıldığında)

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

①ⁿ) 11 7 8 1 5 6 3 17 12

n=1 dönüşa kodlar Gö1



7 11 | 8

11 | 5
8 | 7
7

6 | 17
3 | 12

1 5 7 8 11 | 3 6 12 17

↙
Gösterme

Sıra göre) 1 3 5 6 7 8 11 12 17

Algoritma Analizi

$T(n)$ Merge-Sort ($A[L, \dots, n]$)
 $\Theta(1)$ if $n=1$ return
 $T(\frac{n}{2})$ merge-Sort ($A[1, \dots, \frac{n}{2}]$)
 $T(\frac{n}{2})$ merge-Sort ($A[\frac{n}{2}+1, \dots, n]$)
 $\Theta(n)$ 2 sıralı listeyi birleştir.

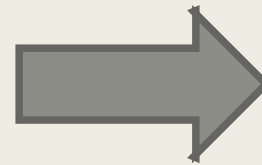
$T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n) + \cancel{\Theta(1)}$

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

özyineleme bağıntısı
recurrence relation

güçlendirme
zamanı

7 Özyineleme ağacı
- Master (Ana) Teorem
Yerine kayna yöntemi



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

4. dönem **Algoritmalar** dersinde nasıl elde edildiği öğretilecek.

Insertion Sort	Merge Sort
$T(n) = \Theta(n^2)$	$T(n) = \Theta(n \cdot \log n)$
* iteratif Algoritma	* Rekürsif Algoritma
* Aynı dizi üzerinde çalışır.	* Geçici dizi kullanır.



n^2 $n \cdot \log n$?
 n=100 için
 10000 660

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 sayısı)

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: $28 \cdot 28 \cdot 28 \cdot 10 \cdot 10 \cdot 10 = 24.389.000$ adet.

Listeler ve Sema

S sırlı bir küme olsun.

En az bir öz alt kümesine birebir eşleme
göksen bir kümeye sırlı küme denir.

Eğer S 'nin elemanları

1. eleman	S_1
2. "	S_2
⋮	⋮
n. "	S_n

Şeklinde izomtlenebiliyorsa S kümesine numaralenebilir
küme denir.

*) Elementlerinin sıralamaya dizelerine liste

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

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

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

(Sıra, Elemanı)

$$\overbrace{(2, 5)} = \overbrace{(x+2, y-3)} \Rightarrow \underbrace{x+2=2}_{x=0} \quad \underbrace{y-3=5}_{y=8}$$

$(x_n)^n \quad S = \{1, 2, 3\}$ ob... ..

2' תי תיכר: יגזעו. ז.
טזחולקו

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

→ g tone 2 bzw. liste vordr.

n elemanlı bir kümenin 2-uzunluk listesi sayısı: n^2
 " " " " 3 " " " = n^3
 ⋮
 " " " " k - " " " = n^k

⊛ Eğer listelerde aynı eleman ydusa (elemanları tekrarsız) kaç tane liste yapılır.

$$n^2 - n = n \cdot (n-1) \Rightarrow 2' \text{ li liste}$$

3' li liste \Rightarrow $\frac{n \cdot (n-1) \cdot (n-2)}{3 \text{ tane}}$

$(1,2,3), (1,3,2)$

$\downarrow \quad \downarrow \quad \downarrow$

$n \cdot (n-1) \cdot (n-2)$

\vdots

\vdots

k' li liste $\Rightarrow \frac{n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdot \dots \cdot (n-k+1)}{k \text{ tane}}$

C_n^k

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

kaçanesini (i) 4 uzunluklu liste sayısı = 6^4

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

$$n=6 \quad k=4$$
$$n-k+1=3$$

$$= 6 \cdot 5 \cdot 4 \cdot 3$$
$$= 360$$

Grupun Listesi:

1. elemanın n

2. elemanın m farklı

ortamdan seçildiği 2'li listelerin sayısı, $n \cdot m$ dir.

Örnek Ad ve soyadların ilk harfleriyle kaç farklı kodlama yapılır? 29 harf (Ğ alınmaz)

$1(28) * 28$ farklı kodlama yapılır.

$28 * 27 \Rightarrow$ harfler tekrarsız.

ör)

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

i)

1 rakam 1 harf şeklinde kaç kodlene = 10×29

ii)

Harf Harf Harf rakam = $29 \times 29 \times 29 \times 10$

ör)

1'den 49'a kadar sayıların bulunduğu bir
ortamda 6 uzunluklu liste sayı nedir? 49^6

rakamları telaffuz = 49.48.47.46.45.44

(10) 11

29 kişi 3 tane isim kullanarak hesaplarına isimler veriyor. Kısaca farklı isim verilir.

$29^3 \Rightarrow$ tekrarl,

$29.28.27 \Rightarrow$ tekrarsız.

(10) 11

11 rakam kullanarak ilk rakam 0 ve 1

Örneğin kaç tane telefon numarası vardır?

8

10 10 10 - - - - - 10

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

10 seçenekli

(10 tane eleman var)



rakamları

112 rakamı

① 9 rakamlı S.S.K numaralarının dağılımı bir antendir;

i) Kaç tane yazılır? 10^9

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

iii) Kaç tane tüm rakamları çifttir : 5^9

iv) Kaç tane en az bir rakamı 8'dir : Tüm durum - 8'in olmayı durumu

{0, 1, 2, 3, 4, 5, 6, 7, ~~8~~, 9} = $10^9 - 9$
9 eleman } 9, sekiz yoktur.
9 uzunlukta

En az bir tane 8 vardır.

② {2, 5, 7} ⇒ kaç tane 3 uzunluklulık yazılır? → 3^9 tane.

★ Teorem: Tekrarlız n elemanı tamenün kulubıgı n 'li listelin sayısı $n!$ dir.

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

Seyna Yolları: 1. dayın n_1 farklı
2. dayın n_2 farklı

seceretden secildiği bir zamanda;

1. keya 2. dayın sayısı $= n_1 + n_2 \Rightarrow$ toplama yoluyla

1. ve 2. " sayısı $= n_1 \cdot n_2 \Rightarrow$ Gorpma "
Seyna dir.

(u) 7 bluz, 5 etek, 9 elbisesi den beyaz kaç farklı şekilde giyilir.

$$7 \times 5 + 9 = 44 \text{ farklı giysi giyilebilir.}$$

(u) 4



A'dan C'ye kaç farklı şekilde gidilir?

$$4 \cdot 3 + 2 = 14$$

(A → B ve B → C) veya

ör) Bir bilgisayar işletim sisteminde A'dan Z'ye
29 harf ve 0'dan 9'a kadar rakamlar kullanılarak

en çok 8 uzunluklu ilk eleman bir harf olan

harflardan veya rakamlardan oluşan ve önce harflar
sonra rakamlar gelecek şekilde kaç farklı başka
ismi yazılır?

cad 0024
3h 4r

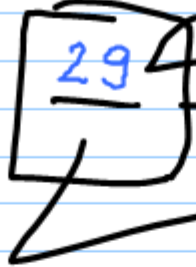
~~d 1296~~
~~a 102d~~

Q 235 ✓

Q t 110 ✓

8 uzunluklu

harf



7 eleman

7 harf	0	eleman	:	$29^7 \cdot 10^0$
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	"	:	$29^0 \cdot 10^7$

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

$$\begin{aligned}
 \text{uzunluk} &= 29^7 + 29^6 \cdot 10^1 + 29^5 \cdot 10^2 + 29^4 \cdot 10^3 + 29^3 \cdot 10^4 \\
 &\quad 29^2 \cdot 10^5 + 29^1 \cdot 10^6 \\
 &\quad \vdots \\
 &\quad \vdots \\
 &\quad \vdots
 \end{aligned}$$

\downarrow
 ilk eleman herf.

$$1 \text{ uzunluk} = 29^1 \text{ (Sadece herf olabilir.)}$$

Tanım (Permutasyon)

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

n elementli bir kümenin birbirinden farklı r elementinin her bir dizilişine A kümesinin r 'li permutasyonu

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

Örnek $A = \{a, b, c\}$ 2'li Permutasyonlarının sayısı

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

$(a, b), (a, c), (b, a), (c, a), (b, c), (c, b)$ ~~7~~
6 tane

4) m)

$A = \{a, b, c, d, e, f\}$ kümesinin 4'lü permütasyonlarını

kaç tane vardır;

i) c bulunur \Rightarrow

1. yol

Tümünün - c'nin olma olasılığı

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

$$= 360 - 120 = 240 \checkmark$$

ii) c bulunmaz
d bulunur

iii) a veya c
bulunur

2. yol

$$\frac{1}{c} \text{ 3 kez bulunur}$$

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

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

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

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

$$\begin{array}{r} 60 \\ 60 \\ 60 \\ \hline 240 \end{array}$$

ii) c bulunmaz?
 d bulunur $\therefore P(5, 4) - P(4, 4)$
 a, b, e, f \nearrow c yok d yok a, b, e, f
 \searrow d olabilir c yok
 $= 120 - 24 = 96$

iii) = Tüm durum - a ve c yok a, b, c, d, f, e
 $= P(6, 4) - P(4, 4)$
 $= 360 - 24 = 336$

4-jane

Tekeleli permütasyon

n_1 tane 1. cinsten, n_2 tane 2. cinsten, ..., n_r tane r ci cinsten
ve $n_1 + n_2 + \dots + n_r = n$ olmak üzere;
 n tane elemanın birbirinden farklı sıralanması; $\frac{n!}{n_1! n_2! \dots n_r!}$

Örn)

111 0052 sayısının rakamları ile 7 basamaklı

- a) kaç sayı yazılır.
- b) kaç çift sayı yazılır.
- c) $2 \cdot 10^6$ da kaç sayı yazılır.

$$a) \frac{7!}{3! \cdot 2! \cdot 1! \cdot 1!} = 420 \rightarrow 0 \text{ ile bölgelebilir.}$$

$$\frac{6!}{3! \cdot 1! \cdot 1! \cdot 1!} = 120 \rightarrow 0 \text{ ile bölgelebilir.}$$

$$420 - 120 = 300 \rightarrow$$

$$\begin{aligned} b) & \text{0 ile bölgelebilir} \quad \frac{6!}{3!} = 120 \\ & \text{0 ile bölgelebilir 0 ile bölgelebilir} \quad \frac{5!}{3!} = 20 \end{aligned} \quad \left. \begin{array}{l} 120 - 20 = \underline{\underline{100}} \\ \text{sonu sıfır olan.} \end{array} \right\}$$

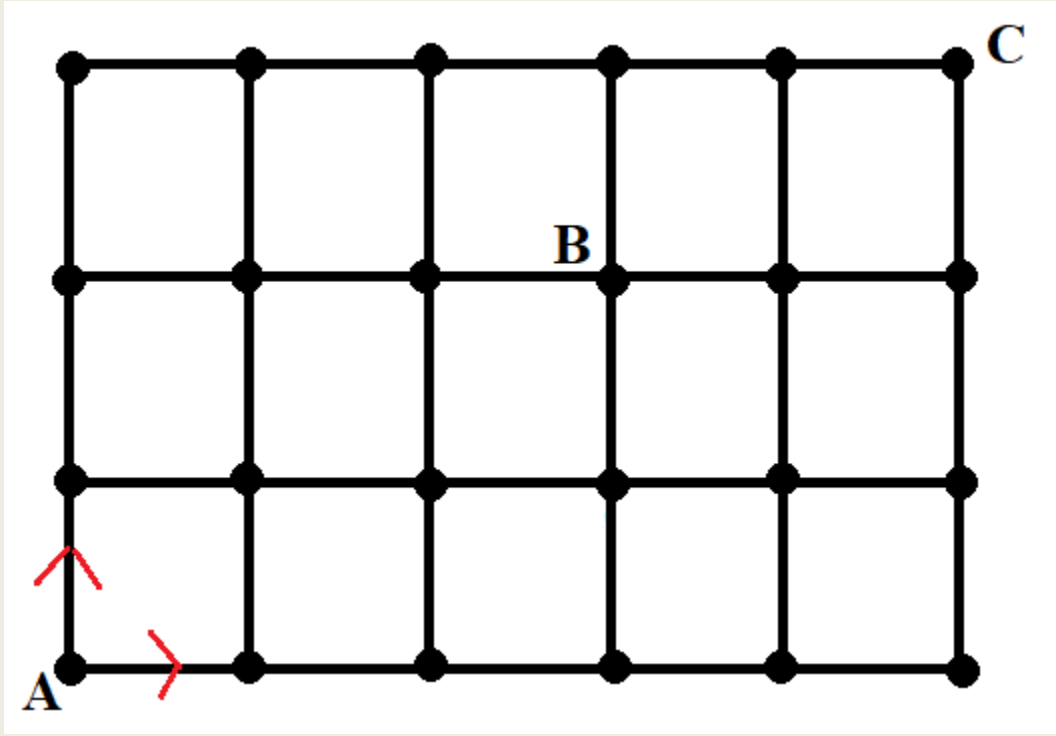
$$\begin{aligned} & \text{2 ile bölgelebilir} \quad \frac{6!}{3! \cdot 2!} = 60 \\ & \text{0 ile bölgelebilir 2 ile bölgelebilir} \quad \frac{5!}{3!} = 20 \end{aligned} \quad \left. \begin{array}{l} 60 - 20 = \underline{\underline{40}} \\ \text{2 ile bölgelebilir} \end{array} \right\}$$

$$\Rightarrow 100 + 40 = 140 \text{ adet olanlar.}$$

c) $2 \cdot 10^6$ 'dan büyük olması için milgenle bölgelebilir 2 veya 5

$$\frac{420 \cdot 2}{7} = 120 \rightarrow$$

Örnek:



Çözüm:

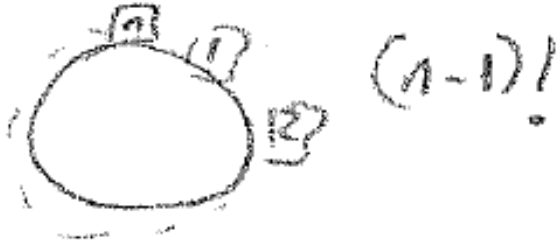
$$a) \frac{8!}{5! \cdot 3!} = \frac{8 \cdot 7 \cdot 6 \cdot 5!}{5! \cdot 3 \cdot 2} = 56$$

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

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?

Dönel Permutasyon



Örnek

5 kut, 3 fiz, 2 kimga öpirtmenin yururde bir masa etrafına,

a) kaa fahhı zellıde? $\rightarrow 7!$

b) aynı branslatık kaa fahhı zellıde? $2! \cdot 5! \cdot 3! \cdot 2!$

Permütasyon Algoritması ve Analizi

```
#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;
}
```

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

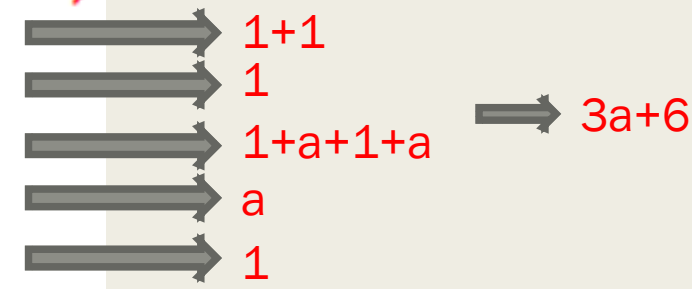


Diagram illustrating the execution of the `faktoriyel` function with arrows indicating the number of operations for each line of code:

- `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

The total number of operations is summarized as $3a+6$.

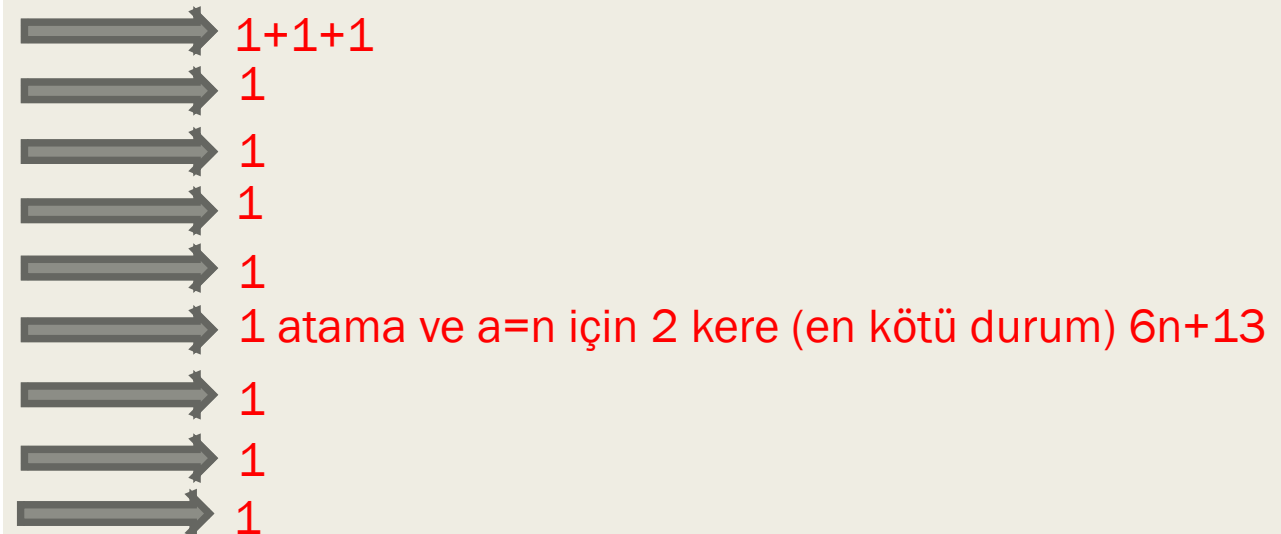


Diagram illustrating the execution of the `main` function with arrows indicating the number of operations for each line of code:

- `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. Tersine eğer n güvercin $n+1$ yuvaya koyulacak olursa en az bir yuva boş kalacaktır.

— **Prensibin daha genel hali ise**, eğer $k \cdot n + 1$ 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
(Ayırık 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.