

Algoritma Analizi ve Tasarımı

GİRİŞ

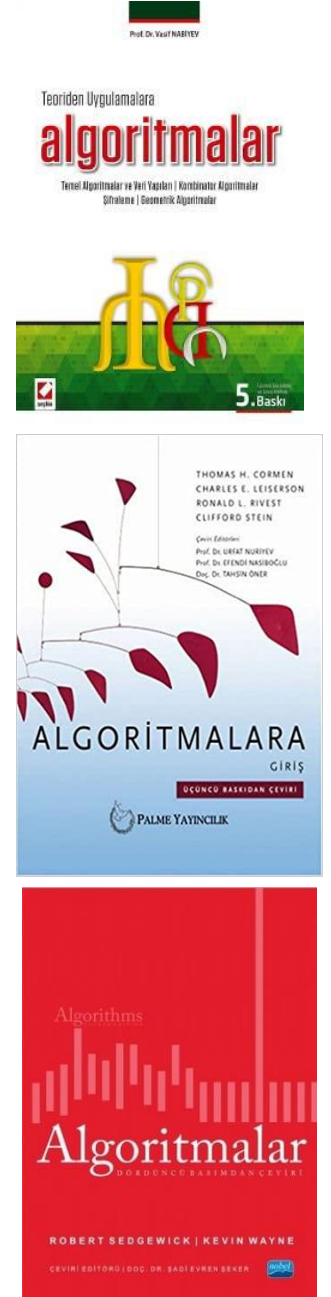
Ders Hakkında

► Ders Kaynakları

- **Algoritmalar** : Prof. Dr. Vasif NABİYEV, Seçkin Yayıncılık
- **Algoritmalar Giriş** : Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Palme YAYINCILIK
- **Algoritmalar** : Robert Sedgewick , Kevin Wayne, Nobel Akademik Yayıncılık

► Değerlendirme

- Araştırma Konuları- Program Geliştirme
- Yazılı Sınav



Temel Kavramlar

❖ Algoritma

- Bir problemin çözümünü belirli bir zamanda çözmek için sonlu sayıdaki adım-adım birbirini takip eden işlemlerdir.

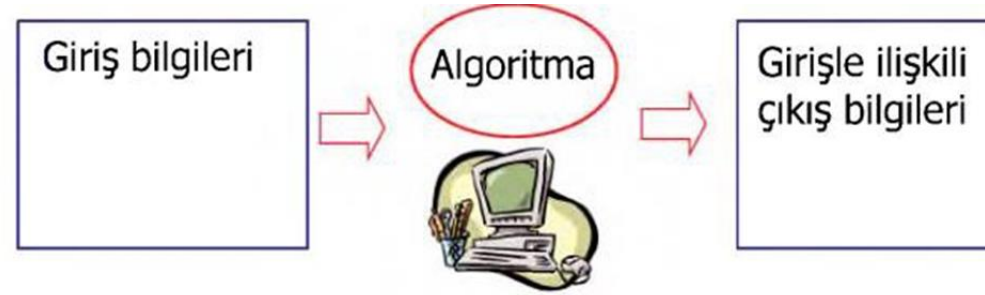
❖ Program

- Bir programlama dilinde bir algoritmanın gerçekleştirilmesidir.

❖ Veri Yapısı

- Problemin çözümü için gereken verinin organize edilmesidir.

Algoritmik çözüm



- Algoritma, **giriş örnekleri üzerindeki işlemleri tanımlar ve uygun çıkış üretir.**
- Aynı algoritmik problem için farklı algoritmalar olabilir.

Geliştirilecek Algoritma Nasıl Olmalıdır?



Geliştirilen Algoritmalar Hangisi Seçilmelidir?

- Üretilen algoritmalar**dan en hızlısı ve en az hafıza kullanan algoritma tercih edilir.**



Geliştirilen Algoritma nasıl olmalıdır?



Program

- Üretilen algoritmaya göre kullanılacak alana uygun programlama dilleri seçilerek program geliştirilir.



Programlama Dillerinin Sınıflandırılması

1. Genel Sınıflandırma

- ❖ **Temel (Imperative) Programlama Dilleri**
 - Fortran, C, Cobol, Basic, Pascal
- ❖ **Veriye Yönelik Programlama Dilleri**
 - Lisp, Apl, Snobol, Icon
- ❖ **Nesneye Yönelik (Object Oriented) Programlama Dilleri**
 - C++, Java, C#, PHP, Python
- ❖ **Fonksiyonel Programlama Dilleri**
 - Haskell, F# vb.

2. Uygulama Alanlarına Göre Sınıflandırma

❖ **Bilimsel ve Mühendislik Dilleri**

- Fortran, C, Pascal

❖ **Sistem Programlama Dilleri**

- C, Assembler

❖ **Veri Tabanı Dilleri**

- Dbase, Clipper

❖ **Yapay Zeka Dilleri**

- Prolog, LISP, Python

❖ **Genel Amaçlı Programlama Dilleri**

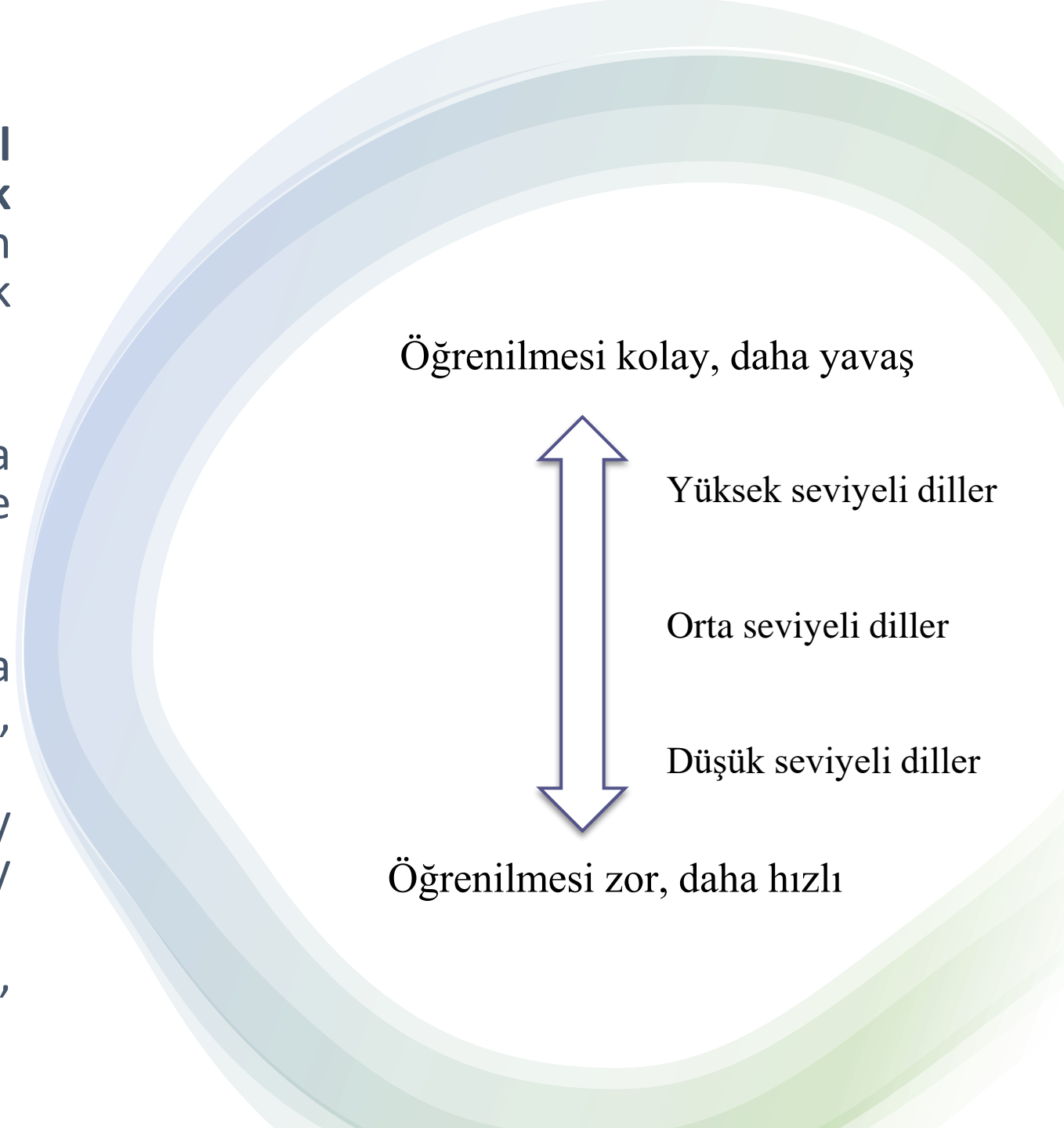
- C, Pascal, Basic, C++

❖ **Web Programlama Dilleri**

- PHP, Asp.Net, Python, JSP





















❖ Bir programlama dili **konuştuğumuz doğal dile ne kadar yakın** ise o kadar **yüksek seviyeli dil**, makine diline ne kadar yakın ise o kadar **düşük seviyeli dil** olarak sınıflandırılır.

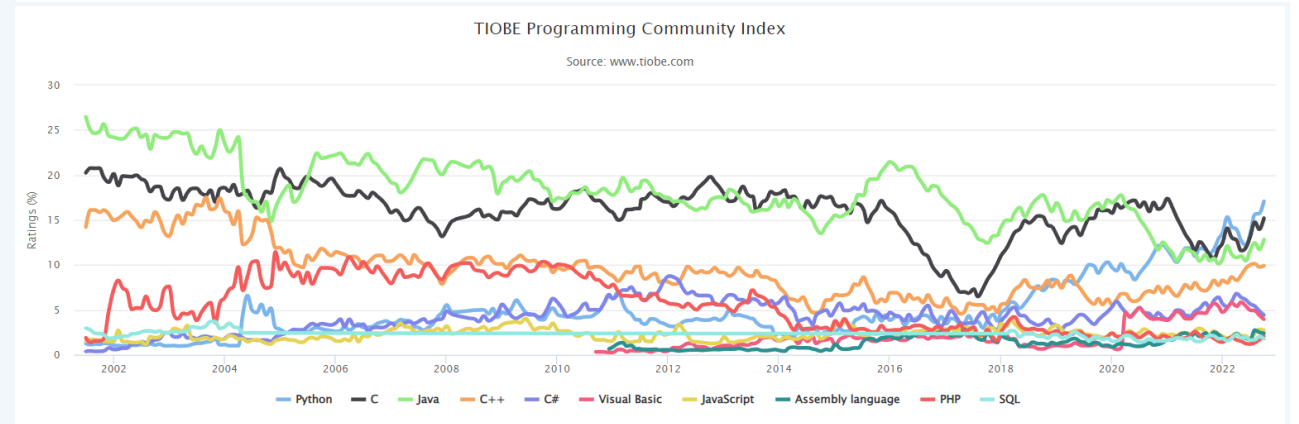
- **Düşük seviyeli:** Bilgisayar donanımına direkt erişim ve tam hakimiyete sahiptir.
 - Makine Dili, Assembly
 - **Orta seviyeli:** Belleğe tam erişim, kısa ve anlaşılır program koduna sahiptir. C, C++ vb.
 - **Yüksek seviyeli:** Veritabanına kolay erişim, hazır modüller sayesinde kolay programlama yapışan sahiptir.
- Fortran, Visual Basic, Pascal, Cobol, C#, PHP, Java, Python vb.



Programlama Dilleri

Yazılım kalitesinin **değerlendirilmesi** ve **izlenmesi** konusunda çalışmalar yapan ve grafikler sunan **Tiobe Index**'ine göre Ekim 2022 yılında sıralaması aşağıda verilmiştir.

Oct 2022	Oct 2021	Change	Programming Language	Ratings	Change
1	1		 Python	17.08%	+5.81%
2	2		 C	15.21%	+4.05%
3	3		 Java	12.84%	+2.38%
4	4		 C++	9.92%	+2.42%
5	5		 C#	4.42%	-0.84%
6	6		 Visual Basic	3.95%	-1.29%
7	7		 JavaScript	2.74%	+0.55%
8	10	▲	 Assembly language	2.39%	+0.33%
9	9		 PHP	2.04%	-0.06%
10	8	▼	 SQL	1.78%	-0.39%
11	12	▲	 Go	1.27%	-0.01%
12	14	▲	 R	1.22%	+0.03%
13	29	▲▲	 Objective-C	1.21%	+0.76%
14	13	▼	 MATLAB	1.18%	-0.02%
15	17	▲	 Swift	1.05%	-0.06%
16	16		 Ruby	0.88%	-0.24%
17	11	▼▼	 Classic Visual Basic	0.87%	-0.96%
18	20	▲	 Delphi/Object Pascal	0.85%	-0.09%
19	18	▼	 Fortran	0.79%	-0.29%
20	26	▲▲	 Rust	0.70%	+0.17%



Veri Yapısı

- **Veriler (Data)**, işlenebilir duruma getirilmiş *sayısal* ya da *sayısal* olmayan niceliklerdir.
- Günümüzde internet üzerinden yaptığımız *her işlem, her paylaşım bir veri oluşturmakta ve dijital ayak izi bırakmaktadır.*



Büyük Veri (Big Data)

- Günümüzde Pandemi dönemi ile birlikte teknolojik dönüşüm (dijital dönüşüm) oldukça hızlanmıştır.
- Dolayısıyla işlenebilecek veri miktarı çok daha fazla artmıştır.
- **Büyük veri:** yeni veri kaynaklarından gelen *daha büyük, daha karmaşık* veri kümeleridir.



B y k Veri (Big Data)

B y k veri, zamanda    V olarak da bilinir.

1. Volume (Hacim)
2. Velocity (Hız)
3. Variety ( e itlilik)



1. Volume (Hacim)

Yüksek hacimli düşük yoğunluklu, yapılandırılmamış verileri işlemeniz gerekir.

Bir Twitter, LinkedIn gibi sosyal medya uygulamalarından gelen içerikler

Bir web sayfasındaki veya bir mobil uygulamadaki tıklama akışları

Bir sensör özellikli ekipmanlardaki bilinmeyen değerli veriler olabilir.

2. Velocity (Hız)

Verilerin alındığı ve (belki de) işlem yapıldığı hız oranıdır.

Normalde, en yüksek veri akışları hızı, diske yazılmak yerine doğrudan belleğe aktarılır.

İnternet özellikli bazı akıllı ürünler gerçek zamanlı veya neredeyse gerçek zamanlı olarak çalışır.

3. Variety (Çeşitlilik)

Mevcut olan birçok veri türünü ifade eder.

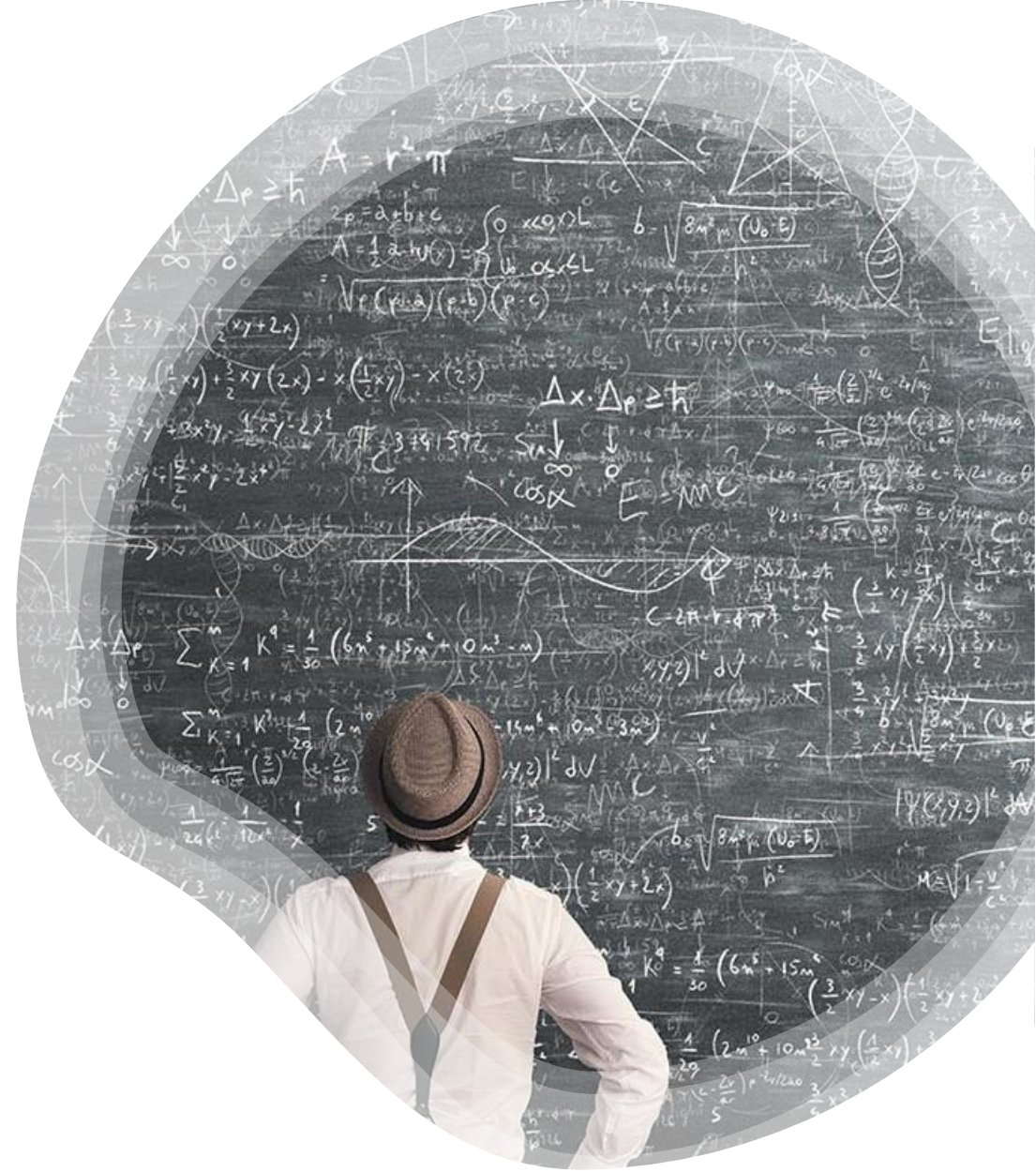
Geleneksel veri türleri bir ilişkisel veritabanında yapılandırılır ve yerleştirilir.

Büyük verinin yükselişiyle, **veriler** yeni yapılandırılmamış veri türleri (metin, ses ve video gibi) biçiminde gelir.

Yapılandırılmamış ve yarı yapılandırılmış veri türleri, anlam türetmek ve meta verileri desteklemek için ek ön işleme süreci gerektirir.

ÖNEMLİ PROBLEM TÜRLERİ

- **Hesaplama** karşılaşılan sınırsız problem denizinde, araştırmacıların özellikle dikkatini **çeken önemli problem türleri vardır.**



ÖNEMLİ PROBLEM TÜRLERİ

Sıralama (Sorting)

Arama (Searching)

Sözel bilgi işleme (String processing)

Graf problem (Graph problems)

Kombinatöryel problemler (Combinatorial problems)

Geometrik problemler (Geometric problems)

Küme teorisi ile ilgili problemler

Sayı teorisi veya sayısal problemler

SIRALAMA PROBLEMLERİ



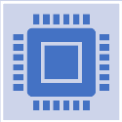
Kayıtlar söz konusu olduğunda, sıralamayı yönlendirmek için bir bilgi parçası seçmemiz gerekir.



Örneğin, öğrenci kayıtlarını *alfabetik ad sırasına göre* veya *öğrenci numarasına* veya *öğrenci not ortalamasına* göre sıralamayı seçebiliriz.



Özel olarak seçilmiş bir bilgi parçasına **anahtar (key)** denir.



Bilgisayar bilimcileri, listedeki öğeler **kayıt değil de**, örneğin **tamsayılar olsa bile**, genellikle bir anahtar listesini sıralamaktan bahseder.



Sıralama, listeye ilgili birçok soruyu yanıtlamayı kolaylaştırır.



Bunlardan en önemlisi aramadır: ***sözlüklerin, telefon rehberlerinin, sınıf listelerinin*** vb. sıralanmasının nedeni budur.



Ayrıca sıralama, **diğer alanlardaki birkaç önemli algorithma**, örneğin **geometrik algorithmalarda** yardımcı bir adım olarak kullanılır.

Neden
sıralanmış bir
liste isteyelim?



Sıralama algoritmaları gerçekten **biri diğerinden daha iyi olsa da**, her durumda en iyi çözüm olacak bir algoritma yoktur.

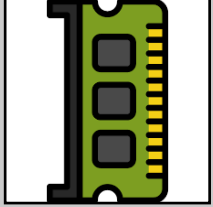


Algoritmalarından bazıları **basit** fakat **nispeten yavaşken diğerleri daha hızlı fakat daha karmaşıktır.**



Bazıları **rastgele sıralanmış girdilerde daha iyi çalışırken** diğerleri ise neredeyse **sıralanmış listelerde daha iyi sonuç verir.**

En iyi çözümü veren sıralama algoritması hangisidir?



Bazıları yalnızca **hızlı**, *bellekte bulunan listeler için uygundur.*



Diğerleri ise bir **diskte depolanan büyük dosyaları sıralamak için** uyarlanabilir.

En iyi çözümü
veren sıralama
algoritması
hangisidir?

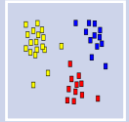
Arama (Searching) Problemleri

- Belirli bir kümede (veya birkaç öğenin aynı değere sahip olmasına izin veren bir çoklu kümede) **arama anahtarı** adı verilen belirli bir değeri bulmakla ilgilenir.
 - Aralarından **seçim yapabileceğiniz çok sayıda arama algoritması vardır.**
-

Arama (Searching) Problemleri



Basit sıralı aramalar yapabildiği gibi olağanüstü verimli ancak sınırlı ikili aramalar yapılabilir.



Temeldeki kümeyi aramaya daha elverişli farklı bir biçimde temsil etmeye dayanan farklı arama algoritmaları da vardır.



Büyük veritabanlarından *bilgi aramaları* da yapacağımız arama algoritmaları vardır.

Arama (Searching) Problemleri

Dolayısıyla arama için de **tüm durumlara en iyi uyan tek bir algoritma yoktur.**

Bazı algoritmalar diğerlerinden daha ***hızlı çalışır*** ancak ***daha fazla bellek gerektirir.***

Bazıları **çok hızlıdır** ancak yalnızca **sıralanmış dizilere uygulanabilir.**

Arama (Searching) Problemleri

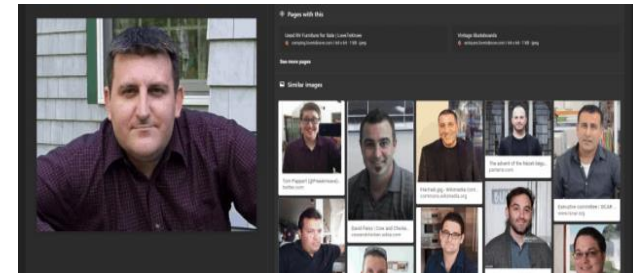
Sıralama algoritmalarının aksine, **kararlılık sorunu yoktur**, ancak farklı sorunlar ortaya çıkar.

Spesifik olarak, temel verilerin ***arama sayısına göre sıklıkla değişebildiği*** uygulamalarda, aramanın diğer iki işlemle birlikte düşünülmesi gerekir: Bir öğenin veri kümesine **ekleme** ve **silme**.

Arama (Searching) Problemleri

Bu gibi durumlarda, **veri yapıları** ve **algoritmalar**, her işlemin gereksinimleri arasında bir denge sağlayacak şekilde seçilmelidir.

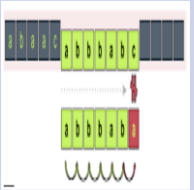
Ayrıca, **verimli arama için çok büyük veri kümelerinin düzenlenmesi**, gerçek hayattaki uygulamalar için önemli sonuçları olan özel zorluklar doğurur.



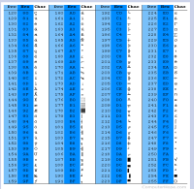
Dizi (String) Problemler



Son yıllarda, **sayısal olmayan verilerle uğraşan uygulamalar** hızla çoğalmıştır.



Araştırmacıların dizi işleme algoritmalarına olan ilgisini daha da artmıştır.



Özellikle ilgi çekici dizeler:

Harfler

Sayılar

Dizi (String) Problemler



Özellikle ilgi çekici dizeler (devam):

Özel karakterlerden oluşan metin dizeleridir.

Sıfırlar ve birlerden oluşan bit dizileri

Dört karakterli alfabeden {A, C, G, T} karakter dizileriyle modellenebilen *gen dizileri*.



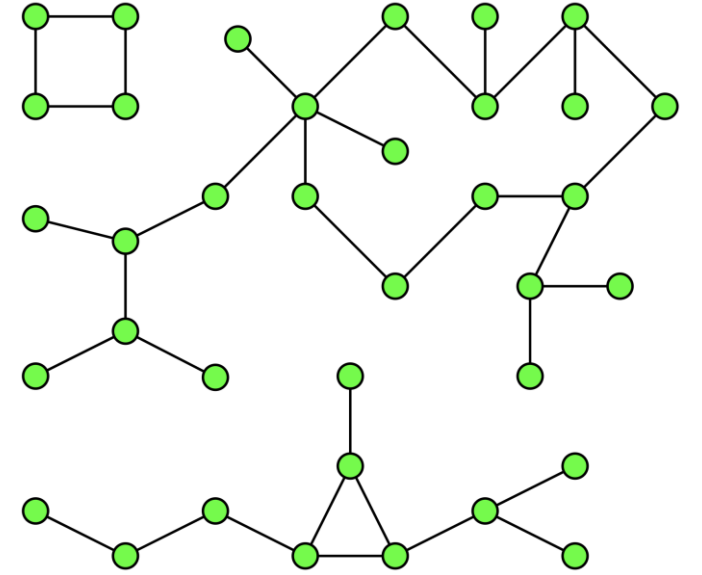
Belirli bir problemde *bir metinde* belirli *bir kelimeyi aramak* araştırmacılardan özel ilgi görmüştür.



Buna dize eşleştirme denmektedir.

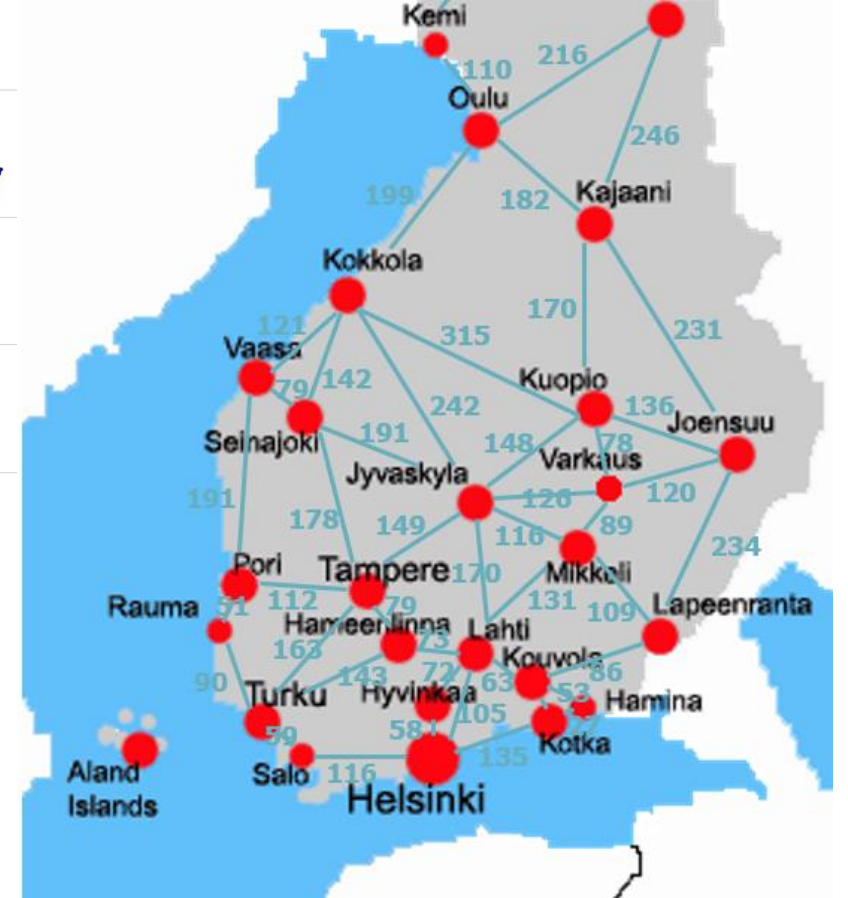
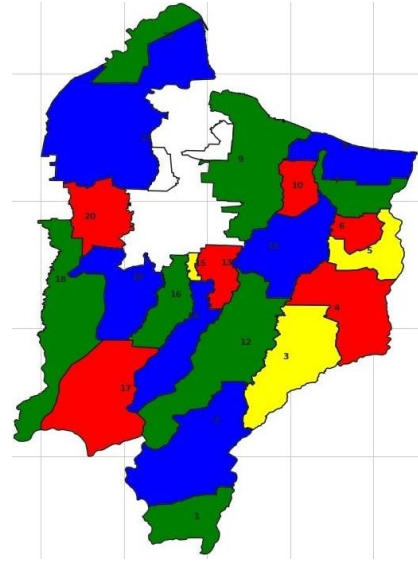
Graf (Graph) Problemler

- ❑ Graflar, ulaşım ve iletişim ağları, proje çizelgeleme ve oyunlar dahil olmak üzere çok çeşitli gerçek hayat uygulamalarını modellemek için kullanılabilir.
- ❑ Örneğin bir kişinin bir web sayfasına en doğru yoldan ulaşmak için takip etmesi gereken maksimum bağlantı sayısının bulması problemi graf kullanarak belirlenir.



Bazı graf problemleri hesaplama açısından çok zordur.

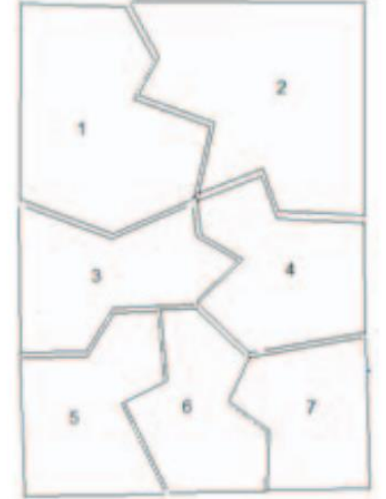
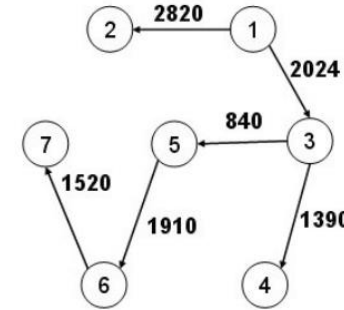
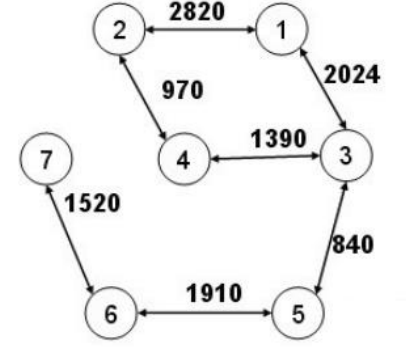
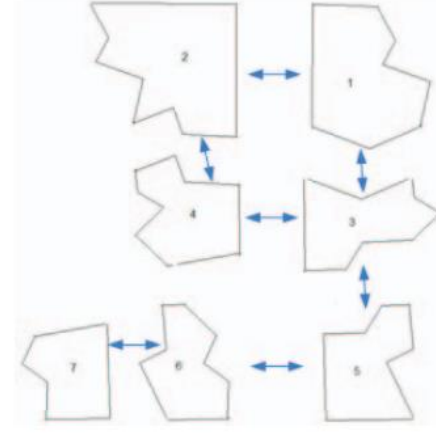
- Gezgin satıcı problemi (TSP)
- Rota planlamak
- Devre kartı ve VLSI çip üretimi
- Renk problemleri



Şehirlerin birbirine bağlanması

Graf (Graph) Problemler

- ❑ Algoritma önermede bakış açısı önemlidir.
- ❑ Örneğin parçalanmış bir belgenin bir araya getirilmesinde graf yapısı kullanılabilir.



Kombinatöryel (Combinatorial) Problemler

Daha soyut bir perspektiften bakıldığında, *gezgin satıcı problemi* ve *graf boyama problemi* **kombinatoryal problemlere** örnektir.

Bunlar, **belirli kısıtlamaları karşılayan** ve **istenen bazı özelliklere** (örneğin, *bir değeri maksimize eden* veya *bir maliyeti minimuma indiren*) sahip **bir permütasyon**, **bir kombinasyon** veya **bir altküme gibi bir kombinatoryal nesne bulmayı** (açık veya dolaylı olarak) isteyen problemlerdir.

Kombinatöryel (Combinatorial) Problemler

Genel olarak kombinatöryel problemler **hem teorik** hem de **pratik** açıdan hesaplamadaki en zor problemlerdir.

Birleştirici nesnelerin sayısı problemin boyutuyla birlikte tipik olarak son derece hızlı bir şekilde büyümektedir.

Bu tür problemlerin **çoğunu tam olarak kabul edilebilir bir sürede çözmek için bir algoritma yoktur**.

Kombinatöryel (Combinatorial) Problemler

SUDOKU

2		9				6		
	4		8	7			1	2
8				1	9		4	
	3		7			8		1
	6	5			8		3	
1				3				7
			6	5		7		9
6		4						2
	8		3		1	4	5	

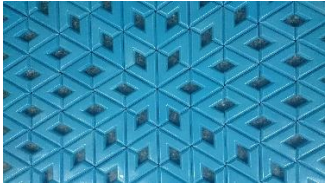
ANSWER:

2	1	9	5	4	3	6	7	8
5	4	3	8	7	6	9	1	2
8	7	6	2	1	9	3	4	5
4	3	2	7	6	5	8	9	1
7	6	5	1	9	8	2	3	4
1	9	8	4	3	2	5	6	7
3	2	1	6	5	4	7	8	9
6	5	4	9	8	7	1	2	3
9	8	7	3	2	1	4	5	6

Poliomino Yapbozlar



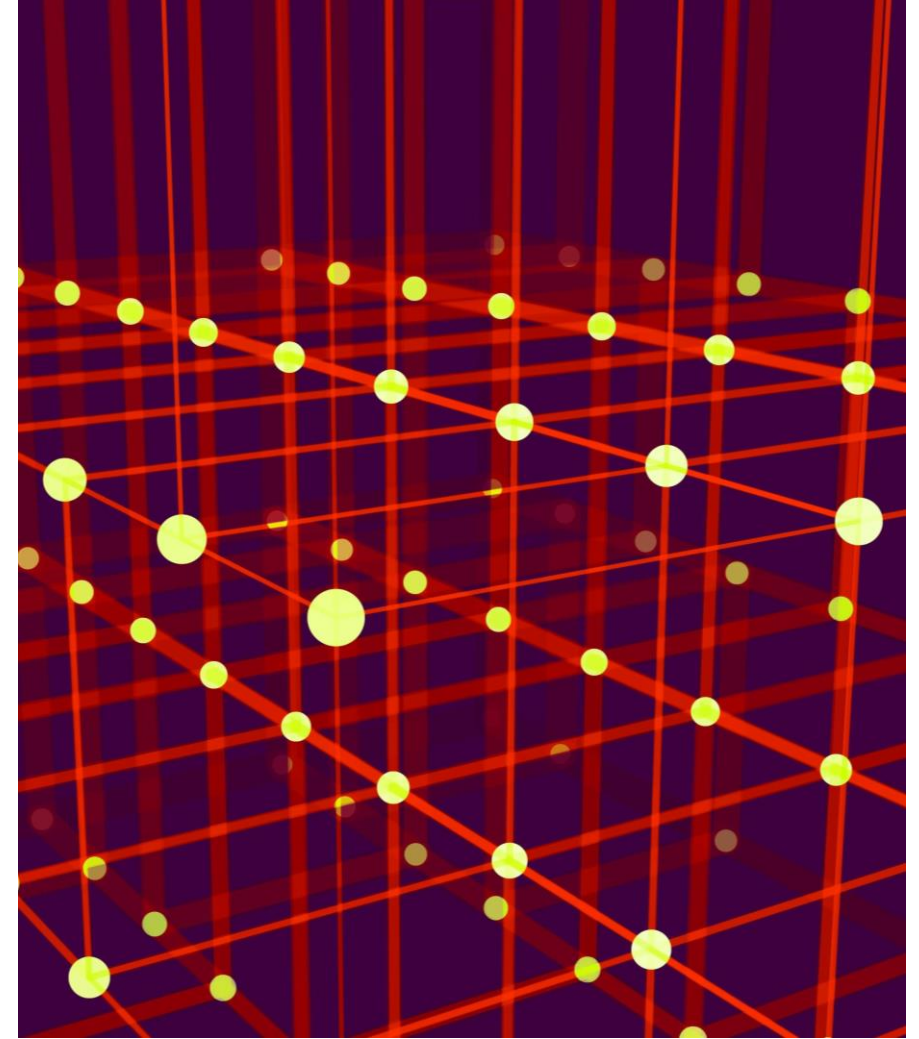
V harfi ile bir alanın farklı şekillerde kaplanması



...

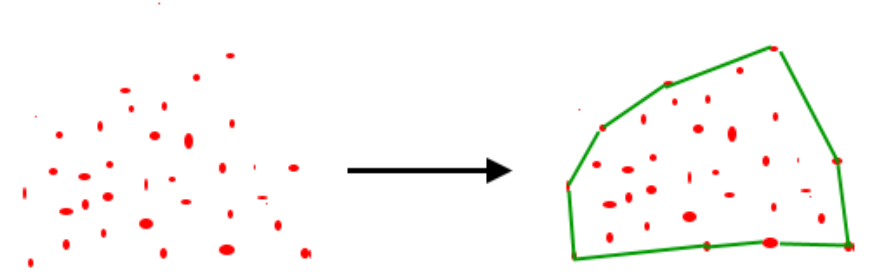
Geometrik (Geometric) Problemler

- Geometrik algoritmalar **noktalar**, **çizgiler** ve **çokgenler** gibi geometrik nesnelerle ilgilenir.
- Hesaplamalı geometri ile ilgili bilinen iki klasik problem **en yakın çift (closest pair) problemi** ve **dışbükey gövde (Convex Hull) problemi**dir.



Geometrik (Geometric) Problemler

En yakın çift problemi: düzlemde verilen n nokta, aralarındaki en yakın çifti bulmaktır.



Dışbükey gövde problemi

Dışbükey gövde problemi: verilen bir kümenin tüm noktalarını içeren en küçük dışbükey çokgeni bulmayı ister.

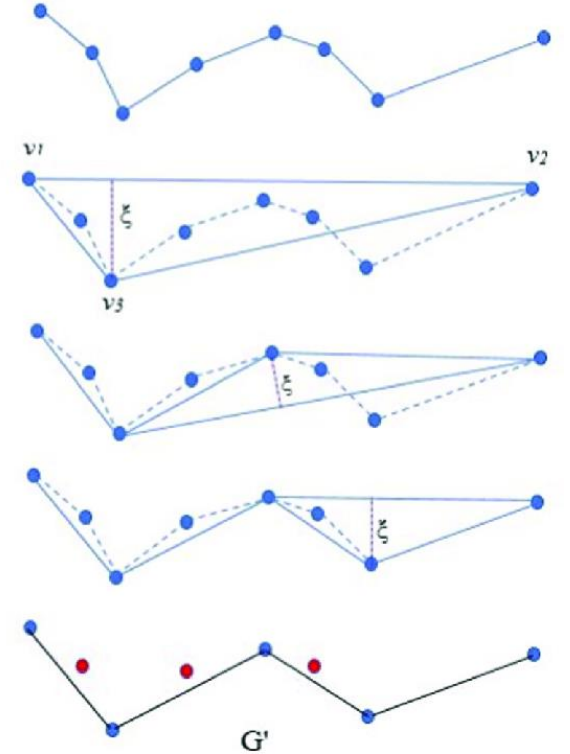
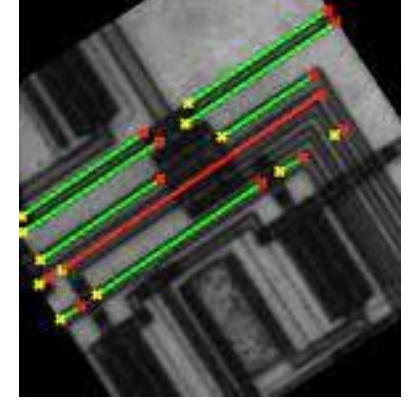
Geometrik (Geometric) Problemler

❖ Bunların dışında *düz çizgi* veya *bir çember* bulma problemleri vardır.

- Bu problem çözmek için *Hough Dönüşüm* gibi popüler algoritmalar geliştirilmiştir.

❖ Karmaşık bir şeklin yapısını basitleştirme problemi de örnek olarak gösterilebilir.

- Bu problemi çözmek için *Douglas Pecuker* gibi popüler algoritmalar geliştirilmiştir.





Kümeleme Algoritmaları

- ❑ Küme teorisine dayanan bu tarz problemlere birkaç örnek aşağıda verilmiştir.
 - ❑ **Nesnelerin kutuya konması**
 - M farklı nesnenin N kutuya farklı yerleştirilme problemleri
-

Kümeleme Algoritmaları

❖Güvercin yuvası ilkesi , Dirichlet çekmecesini veya Dirichlet ilkesi


Bu problem m nesnenin n kutuya yerleştirilmesi şeklinde yorumlanabilir.

Eğer m **güvercin** n sayıda **yuva**ı zapt ederse ve $m > n$ ise en azından bir yuvada 2 veya daha çok güvercin tünemektedir.

Eğer n nesne k kutuya dağıtılsa ve $n > k$ ise en az bir kutu minimum $\left\lceil \frac{n}{k} \right\rceil$ nesne içermektedir.



Şekil 2. Güvercin yuvası ilkesi

İlkenin adının esin kaynağı: 
Deliklerdeki Güvercinler. Burada $n = 7$ ve $m = 9$, buradan en az iki güvercin deliği boş kalacağını söyleyebiliriz. (Eğer iki kuş bir deliği paylaşırsaydı üç boş delik olacaktı.)

Kümeleme Algoritmaları

- Bu tarz probleme bir örnek vermek gerekirse 400 kişi arasından en az ikisi aynı günde doğmuştur.
- **Başka bir örnek ise bilgisayarda yazılan bir programda en çok 4 harften oluşan rastgele 800.000 harf dizisi oluşturulduğunu düşünelim.**
- **Bu kelimelerin hepsi birbirinden farklı olabilir mi?**
- **Çözüm: $29^4 + 29^3 + 29^2 + 29^1 = 732.540$**

Dolayısıyla en az bir kelime 2 kez tekrarlanacaktır.

$$\left[\begin{array}{r} 800000 \\ \hline 732540 \end{array} \right]$$

Kümeleme Algoritmaları

- Güvercin yuvası ilkesi kombinatör hesaplamalarda ve sayı teorisinde sıkça kullanılmaktadır.
- Bu ilke basit olmasına rağmen bazen güvercin ve yuva ilkesinin eşleştirilmesinde zorluklar oluşabilmektedir.
- Burada sonuç kestirimi yapılarak yuvaların en az birisinde 2 güvercin vardır denebilmektedir.

Kümeleme Algoritmaları

- Fakat bu güvercinlerin hangi yuvada yer aldığını söyleyemeyiz.
- Ancak bunlara bakmayarak güvercin yuvası ilkesinde sayı teorisinde ele alınan **Fermant'ın küçük teoremi**, **kalanlarla ilgili Çin teoremi** ispatlanabilmektedir.

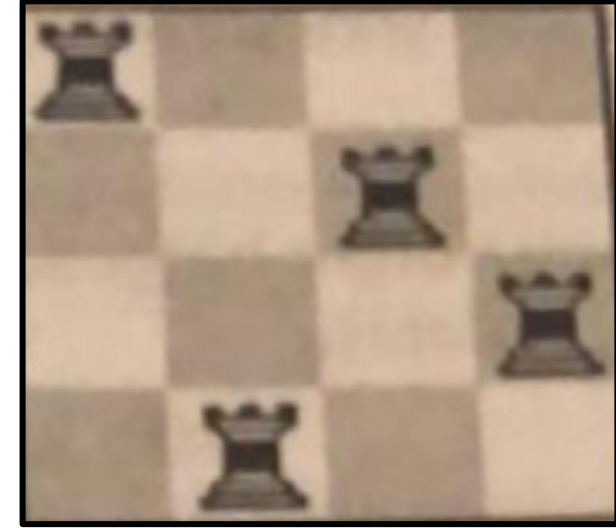
□ Akıl Ustası (Master Mind) Oyunu :

- Kombinasyonlar ve kümelenendirme ile ilgili güzel bir örnektir.
- İki kişilik oynanan bir oyun olup ipuçlar yardımıyla saklı tutulan renklerin rakip tarafından belirlenmesine dayanmaktadır.

Kümeleme Algoritmaları

□Kale Polinomları :

$n*n$ bir alanda birbirini yemeden n sayıda kale kaç farklı şekilde yerleştirilebilir.

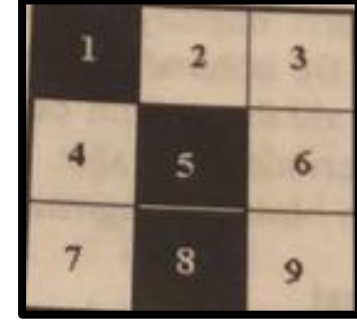


n – kale problemi

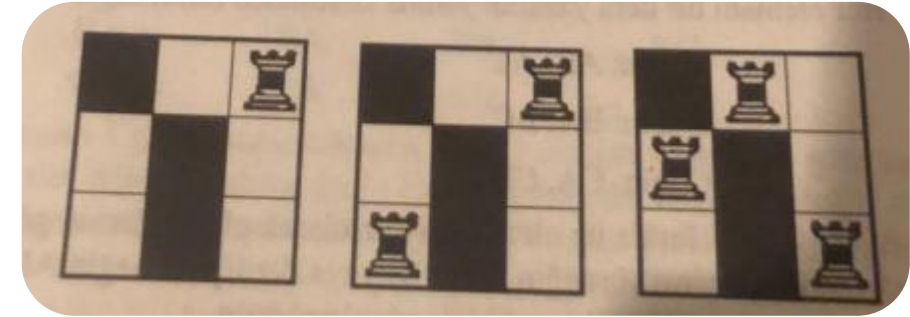
Kümeleme Algoritmaları

□Kale Polinomları :

- Örnek olarak yandaki şekilde **yasaklı bölgeler** 1., 5. ve 8. karelere kalelerin yerleştirilmesine izin verilmemektedir.
- Şekil 5.a'da yasaklı durumlar göz önüne alınarak **1 kale 6 farklı** şekilde yerleştirilir. Benzer şekilde Şekil 5.b'de **2 kale 8 farklı** (her iki kalede aynı bölgede) şekilde ve Şekil 5.c'de **3 kale** (2,4,9) ve (2,6,7) gibi **2 farklı** şekilde yerleştirilir.



Şekil 4. n – kale problemi örnek durum



(a) (b) (c)

Şekil 5. n – kale problemi örnek çözüm

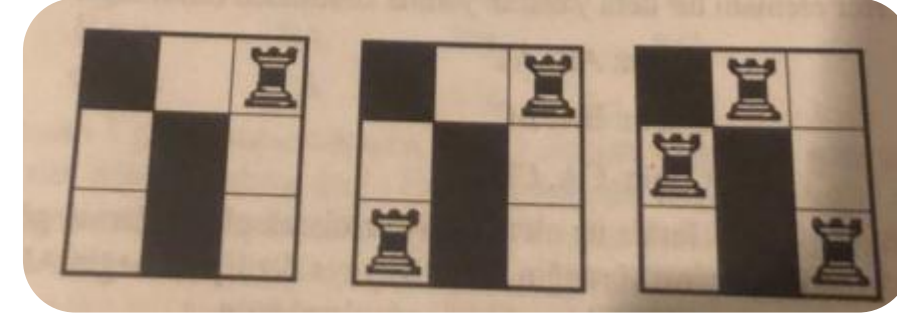
Kümeleme Algoritmaları

□Kale Polinomları :

- Alanın her P parçası için çözüm bulmak için $r_k(P)$ değerlendirilmesi yapılabilir.
- $k=1,2,3$ için örnek durumlar şekilde gösterilmiştir.
- Eğer $r_0=1$ kabul edilirse $r_k(P,x)$ kale polinomu aşağıdaki gibi ifade edilebilir.

$$r(P,x)=1 + 6x + 8x^2+ 2x^3$$

- Burada x^k k sayıdaki kalelerin birbirini yemeden yerleştirilebilecek mümkün sayıyı ifade etmektedir.



(a)

(b)

(c)

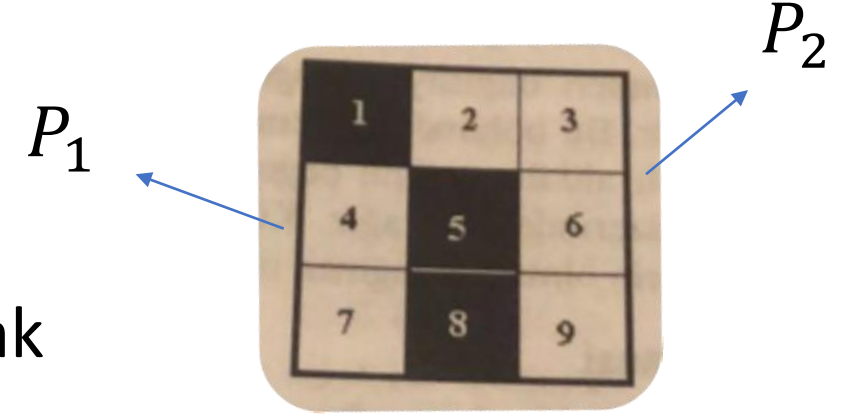
Şekil 5. $n -$ kale problemi
örnek çözüm

Kümeleme Algoritmaları

□Kale Polinomları :

$$r(P,x)=1 + 6x + 8x^2+ 2x^3$$

- Buradaki hesap küme teorisine dayandırarak yapılabilir.
- Düzlemi P_1 ve P_2 olmak üzere iki bölgeye ayırırsak kaleler **yatay** ve **dikey** yönden birbirini yemeseydi bu durumda mümkün yerleştirme sayısı aşağıdaki gibi olacaktır.
- $r(P, x) = r(P_1, x) * r(P_2, x)$
- $r(P_1, x)=1+2x$ ve $r(P_2, x)=1 + 4x + 2x^2$ olur.



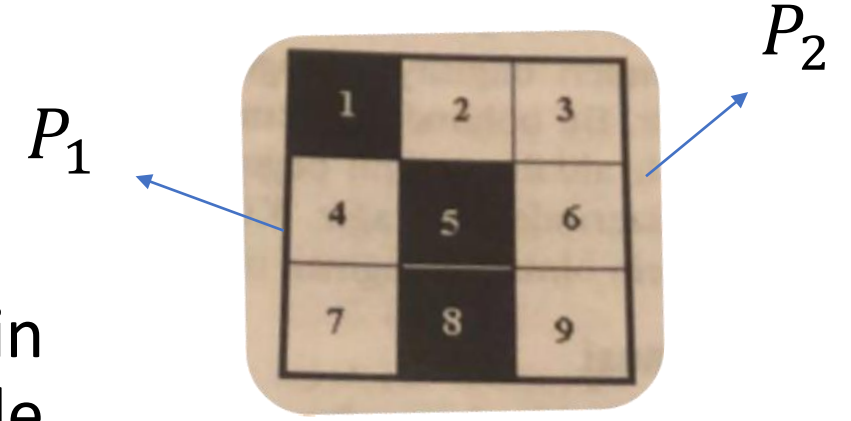
Şekil 4. n – kale problemi örnek durum

Not: P_2 bölgeye 3 kale yerleştirilemez.

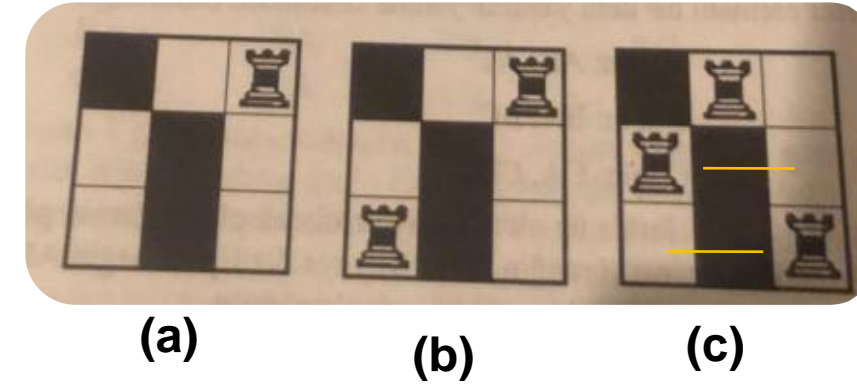
Kümeleme Algoritmaları

❑Kale Polinomları :

- $r(P, x) = r(P_1, x) * r(P_2, x)$
- Buradan toplam çözüm bu alt çözümlerin çarpımından **örtüşen M durumların çıkarılması** ile oluşacaktır.
- $r(P, x) = r(P_1, x) * r(P_2, x) - M$
- $r(P, x) = (1+2x) * (1 + 4x + 2x^2) - (2x^2 + 2x^3)$
- $r(P, x) = 1 + 6x + 8x^2 + 2x^3$ elde edilir.



Şekil 4. n – kale problemi örnek durum



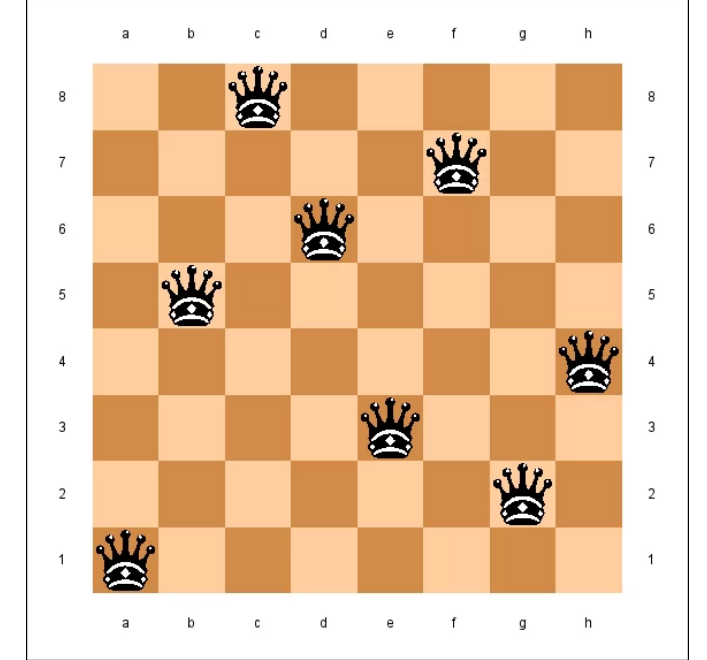
Şekil 5. n – kale problemi örnek çözüm

Kümeleme Algoritmaları

❑ Kale polinomu problemi, yinelemeli şekilde de çözülebilir.

- Herhangi bir kale haneye yerleştirildiğinde ilgili **satır ve sütun silinir ve araştırma alanı küçültülür.**
- Eğer seçili kare kullanılamıyorsa bu durumda geriye kalan alt bölge incelenir.

❑ Bu probleme benzer bir örnek ise bir satranç tahtasına birbirini yemeyecek şekilde **8 vezirin yerleştirilmesine** bakılabilir.



Şekil 6. 8 vezir problemi

Kümeleme Algoritmaları

□Kümelerin alt kümelere parçalanması

N elemanlı kümenin boş olmayan k adet alt kümelere parçalanması problemi olabilir.

❖ 2. Türden Stirling ve Bell Sayıları incelenebilir.

Sayı Teorisi ve Sayısal (Numerical) Problemler



- Sayısal problemler, sürekli nitelikteki matematiksel nesneleri içeren problemlerdir.
- Denklemleri ve denklem sistemlerini çözme,
 - Belirli integralleri hesaplama,
 - Fonksiyonları değerlendirme vb.

Matematiksel problemlerin bazıları ancak yaklaşık olarak çözülebilir.

Sayısal (Numerical) Problemler

Bu alanda yıllar içinde **birçok karmaşık algoritma geliştirilmiştir** ve **birçok bilimsel ve mühendislik uygulamasında** kritik bir rol oynamaya devam etmektedir.

Ancak son 25 yılda bilgisayar endüstrisi odağını iş uygulamalarına kaydırmıştır.

Bu yeni uygulamalar, öncelikle **bilgi depolama, alma, ağlar aracılığıyla taşıma** ve **kullanıcılara sunum** için algoritmalar gerekmiştir.



- **Sayı teorisi ile ilgili örnek problem:**
Asal sayıların bulunması
- **Rasgele ve asal sayı üretimi** pek çok bilim dalı tarafından kullanılmakta olup özellikle *ağ güvenlik protokolleri*, *şifreleme* ve *benzetim problemlerinde* yararlanılmaktadır.
- **Peki neden asal sayılar?**
- Asal sayıların özellikleri ve tahmin edilmesi güçlüğü ön plana çıkarmaktadır.

Asal sayıların bulunması ile ilgili yaklaşımlar

❑Asal sayıların bulunması - Eratosthenes Eleği

- Asal sayıların belirleneceği üst sınırı olan **N** değeri belirlenir.
- İşlemin yapılacağı maksimum sınır değeri olan $k = \sqrt{N}$ hesaplanır.
- $2 \leq i \leq k$ aralığındaki 0'dan farklı **i** değeri için i^2 . pozisyonundan itibaren her i. Değer 0'a eşitlenir.
- Elde edilen 0'dan farklı değerler asal sayılar olarak elde edilir.

❑ **Örnek:** 1..100 arasındaki asal sayılar aşağıdaki gibi hesaplanır.

1. $K=\sqrt{N}=\sqrt{100}=10$ olur. Sırasıyla 10'a kadar olan asal sayılara bakılır.
2. Dizinin ilk elemanı olan 2 asal sayı olarak belirlenir ve $2^2=4$ 'ten itibaren her 2. pozisyondaki sayı 0'lanır.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

	<u>2</u>	<u>3</u>	4	<u>5</u>	6	<u>7</u>	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

	<u>2</u>	<u>3</u>	4	<u>5</u>	6	<u>7</u>	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Sayı Teorisi ve Sayılarla İlgili Algoritmalar

3. 2'den sonra 0'dan farklı ilk eleman 3 bulunur.
 $3^2=9$ 'dan itibaren her **3. pozisyondaki sayı 0'lanır.**

4. 3'ten sonra 0'dan farklı ilk sayı olan sayı 5'tir.
 $5^2=25$ 'ten itibaren her **5. pozisyondaki sayı 0'lanır.**

5. Son seçilecektir. $7^2=49$ 'daşamada asal sayı olarak 7 an itibaren her **7. pozisyondaki sayı 0'lanır.**
(Üst sınır 10 olduğu için 11 sayısına devam edilmez.)

Sonuçta yandaki şekildeki gibi sayılar elde edilir.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

□ Sezgisel Yöntem

- Bu yöntemin en önemli özelliği, **sadece tek sayılar üzerinde işlem yapmasıdır.**
- Çift sayılar asal olamayacağından direk olarak elenmektedir.
- $i=1,2,..,n$ sayılarından tek sayıları elde etmek için **$2i+1$** formülü kullanılır.
- 2 asal sayı bulunamadığından ayrıca eklenir.

□ Sezgisel Yöntem –Algoritması

- Asal sayıların üst sınırı N değeri belirlenir.
- İşlemlerin yapılacağı alt aralığın maksimum olan $max = \lfloor (N - 1)/2 \rfloor$ değeri hesaplanır.
- $n=2i+1$ adımıyla $\lfloor n^2/2 \rfloor$ pozisyonundan başlanarak max değerine kadar ilerlenerek uygun değerler 0'lanır.
- Bu yinelenmeler $i = \sqrt{max}$ değerine kadar devam ettirilir.
- Elde edilen 0'dan farklı değerler için **$2i+1$ 'e göre asal sayılar üretilir.**

□ Sezgisel Yöntem –Örnek

- 1,...,100 arasındaki asal sayıların bulunması için sezgisel yöntemde öncelikle $\max=(N-1)/2 = 49$ olur.
- $k=\text{sqrt}(\max)=7$ bulunarak aşağıdaki adımlar izlenir.
 1. Birinci aşamada $i=1$ için $n=2i+1=3$ olur. $\lfloor n^2/2 \rfloor=4$ ten itibaren , $\max=49$ 'a kadar her 3. pozisyondaki değer 0'a eşitlenir.
 2. $i=2$ için $n=5$ olur. $\lfloor n^2/2 \rfloor=12$ den itibaren her 5. pozisyondaki değer 0'a eşitlenir.
 3. $i=3$ için $n=7$ olur. $\lfloor n^2/2 \rfloor=24$ den itibaren her 7. pozisyondaki değer 0'a eşitlenir.
 4. $i=4$ için $n=9$ olur. $\lfloor n^2/2 \rfloor=40$ tan itibaren her 9. pozisyondaki değer 0'a eşitlenir.
 5. $i=5$ için $n=11$ olur. $\lfloor n^2/2 \rfloor=60$ tan itibaren her 11. pozisyondaki değer 0'a eşitlenir. Bu sayı max değerinden büyük olduğu için işlem sonlandırılır.

❑ Sonuç olarak **Eratosthenes Eleği** ve **Sezgisel Yöntemi karşılaştıracak** olursak **Sezgisel yöntem %60 daha hızlı** bir şekilde asal sayı bulmaktadır.

❑ KRIPTOLOJİ

❖ **RSA Genel Anahtar Kriptosistemi** : SSL, S-HTTP, S-MIME, S/WAN, STT ve PCT

- RSA için bir **ortak anahtar (Public Key)** bir de **özel anahtar (Private Key)** gerekir.
- **Ortak anahtar herkes tarafından bilinir ve mesajı şifrelemek için kullanılır.**
- **Bir ortak anahtarla şifrelenmiş mesaj sadece özel anahtarla çözülebilir.**



1. Yeterince büyük iki adet asal sayı seçilir.

Bu sayılar p ve q olsunlar.

2. Genel/ortak (public) ve özel (private) key elde etmek için n sayısı hesaplanır.

$n=p*q$ şeklinde hesaplanır.

3. $\phi(n) = (p-1)*(q-1)$ değeri hesaplanır.

- Totient sayılar teorisinde, bir tam sayının o sayıdan daha küçük ve o sayı ile aralarında asal olan sayma sayısını belirten fonksiyondur.

4. Şifrelemede kullanılacak anahtarı oluşturmak için e değeri hesaplanır.

$1 < e < \phi$ aralığında aralarında asal olan öyle bir e sayısı rastgele seçilir. (EBOB fonksiyonundan faydalanılır.)

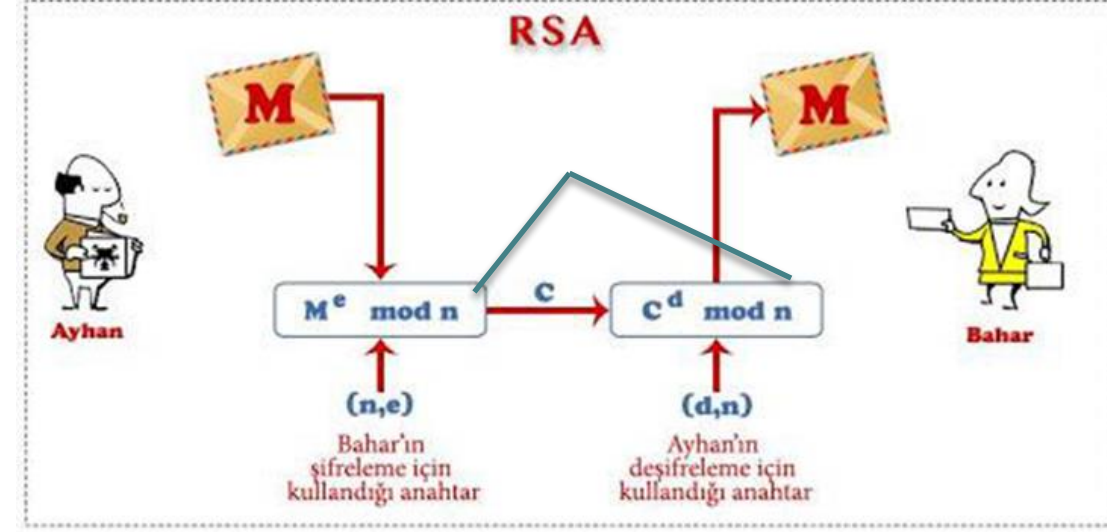
5. Deşifrelemede kullanılacak anahtarı oluşturmak için d değeri hesaplanır.

$1 < d < \phi$ aralığında $d*e \equiv 1 \mod (\phi)$ şartını sağlayan d sayısı seçilir.

Sonuç olarak

❖ Şifreleme için kullanılan anahtar (n,e)

❖ Deşifreleme için kullanılan anahtar (d,n) olur.

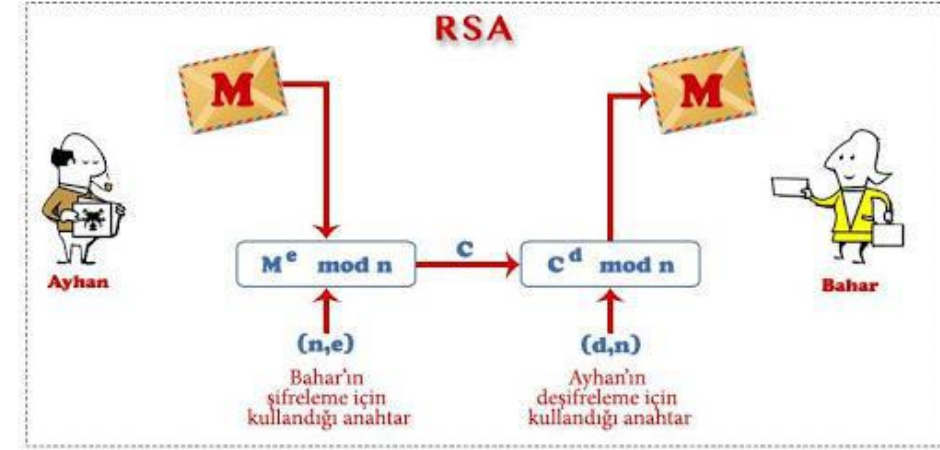


RSA ile mesajın şifrelenmesi:

1. Mesajın gönderileceği kişinin genel anahtarı (n,e) elde edilir.
2. Şifrelenecek mesaj, $[0,n-1]$ aralığındaki m sayısına dönüştürülür.
3. $c=m^e \bmod n$ hesaplanır.
4. Oluşturulan c şifreli mesaj alıcıya gönderilir.

RSA ile mesajın deşifrelenmesi:

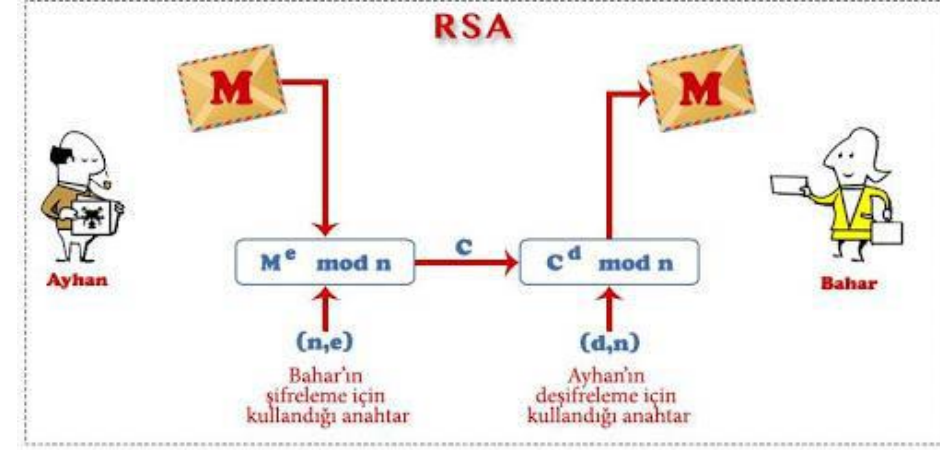
1. d özel anahtarı ile $m=c^d \bmod n$ hesaplanır ve orijinal mesaj elde edilir.



Örnek

1. $p = 61$ ve $q = 53$ olarak seçilsin.
2. n değeri hesaplanır $n = p * q$ şeklinde $n = 61 * 53 = 3233$
3. $\phi(n) = (p-1)(q-1)$, $\phi(n) = (61-1)(53-1) = 3120$
4. $1 < e < \phi(3120)$, $e = 17$ olarak seçilir.
5. **Özel anahtar** olması için bir d sayısı seçilir.
 $d * e \equiv 1 \pmod{n}$ olacak şekilde $d = 2753$ olarak hesaplanır.

❖ Sonuç olarak şifreleme ve deşifreleme için e ve d değerleri hesaplanmış oldu.



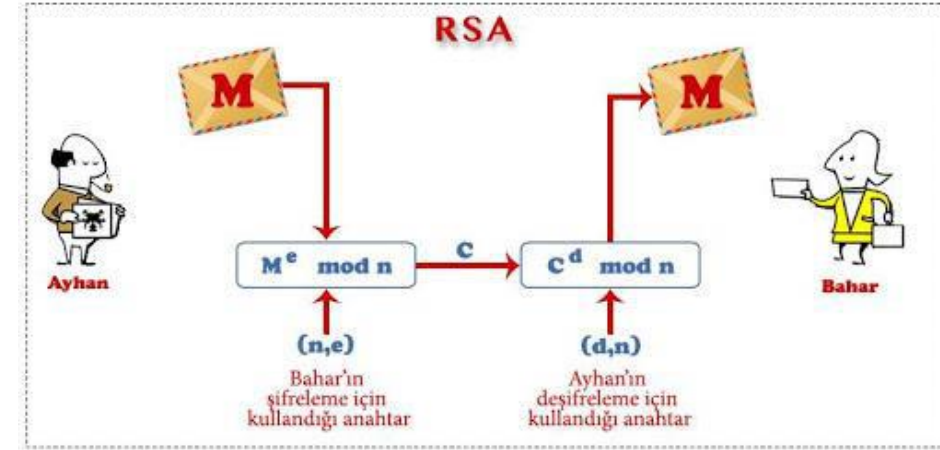
Örneğin mesaj olarak **123** gönderilecek olsun:

Şifreleme:

$$c = m^e \bmod n = 123^{17} \bmod 3233 = 855$$

Deşifreleme:

$$m = c^d \bmod n \rightarrow m = 855^{2753} \bmod 3223 = 123$$



Şifreleme işlemini Armstrong sayıları ile yapabilir miyiz?

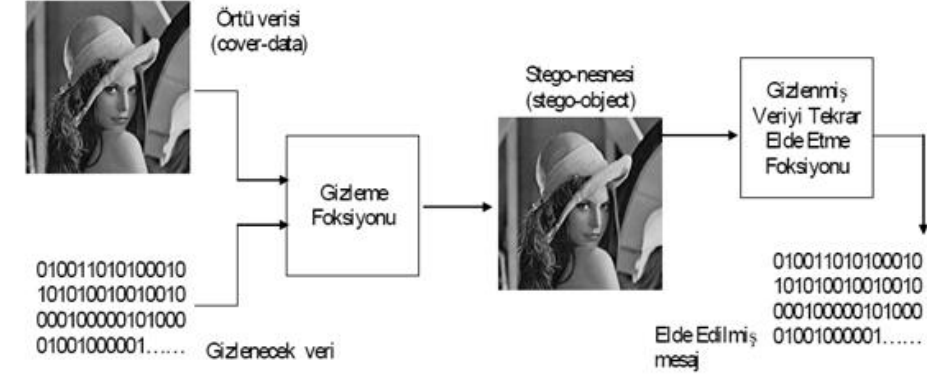
Yaparsak RSA'ya göre avantajı veya dezavantajı ne olur?



$$1634 = 1^4 + 6^4 + 3^4 + 4^4$$

**Armstrong
Number**

□ Steganografi (Sırörtme)

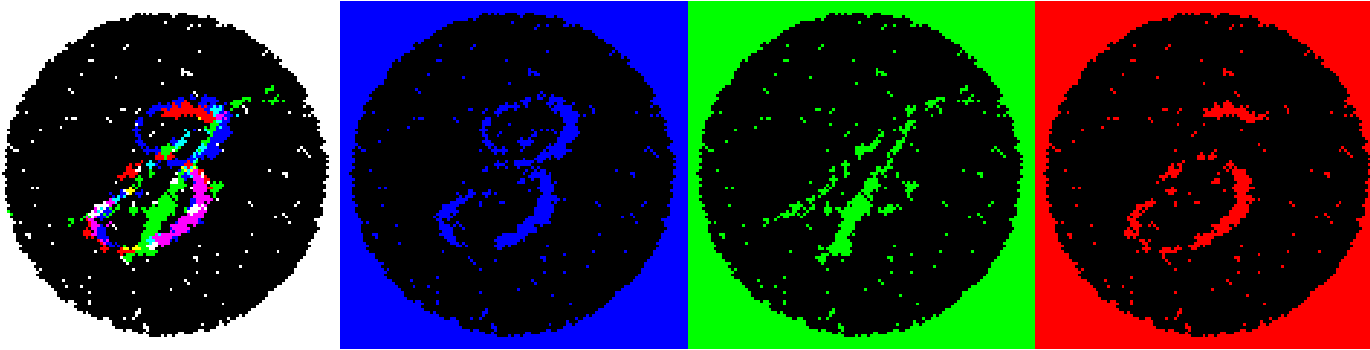
- Örneğin orijinal görüntü içerisine güvenliğe hassas veriler oluşturulup görüntü içerisine gizlenerek eklenir.
- Bu eklenen veriler daha sonra görüntüyü doğrulamak için kullanılır.
- Böylece dijital medyanın gerçekliğini doğrulamak için kullanılır.



- 
- **Steganografi görüntü kaynağının korunması amacının dışında da kullanılmaktadır.**
 - **Örneğin Devletlerin istihbarat teşkilatlarının, iletişim kanallarının dinlenme olasılığına karşı haberleşmelerin gizliliğini korumak için Steganografi tekniğinden faydalanmaktadır.**
 - **Terör örgütleri de ile güvenlik güçlerinin haberleşmelerinin farkına varmaması için Steganografi tekniğini kullanmaktadır.**
- 

❑ Örnek

- (Son yıllarda sosyal medya üzerinde bu tür haberleşmeler artmıştır.) **Bu teknik ile bir görüntü içerisine metin, imge gibi öğeler gizlenebilmektedir.**



En sondaki görüntü farklı renk kanalları ile görüntülenince (mavi, yeşil ve kırmızı) farklı gizli sayıları ortaya çıkarmaktadır.

UYGULAMALAR

❖ Eratosthenes Eleği ve Sezgisel Yöntemi herhangi bir programlama dili kullanarak kodlayınız.

Her iki yöntemin performansını büyük asal sayılar girerek karşılaştırınız.

❖ RSA algoritmasını herhangi bir programlama dili kullanarak kodlayınız.

Örneğin şifrelenecek içerik, 547 gibi sayısal veri olsun. RSA da gerekli parametreler ise klavyeden girilsin.

Kaynakça

- Algoritmalar : Prof. Dr. Vasif NABİYEV, Seçkin Yayıncılık
- Algoritmalara Giriş : Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Palme YAYINCILIK
- Algoritmalar : Robert Sedgewick , Kevin Wayne, Nobel Akademik Yayıncılık
- M.Ali Akcayol, Gazi Üniversitesi, Algoritma Analizi Ders Notları
- Doç. Dr. Erkan TANYILDIZI, Fırat Üniversitesi, Algoritma Analizi Ders Notları
- <http://www.bilgisayarkavramlari.com>