BTK AKADEMİ

Programlama

Doç. Dr. Zafer CÖMERT

BTK AKADEMİ

Bölüm 14

Sıralama ve Arama Algoritmaları ve Rekürsif Fonksiyon kullanımı

Giriş

İçerik

- Özyinelemeli (Recursion) Fonksiyonlar
- Özyinelemeli Fonksiyonlar ile Problem Çözümü
- Sıralama Algoritmaları
  - Kabarcık Sıralama
  - Seçmeli Sıralama
  - Eklemeli Sıralama
  - Birleştirmeli Sıralama
- Arama Algoritmaları
  - Doğrusal Arama
  - İkili Arama

Özyinelemeli (Recursion) Fonksiyonlar

- Kendini çağıran herhangi bir fonskiyon rekürsif (recursive) olarak adlandırılır.

- Özyinelemeli bir yöntem, daha küçük bir sorun üzerinde çalışmak için kendisinin bir kopyasını çağırarak bir sorunu çözer. Bu rekürsif adım (recursion step) olarak tanımlanır.

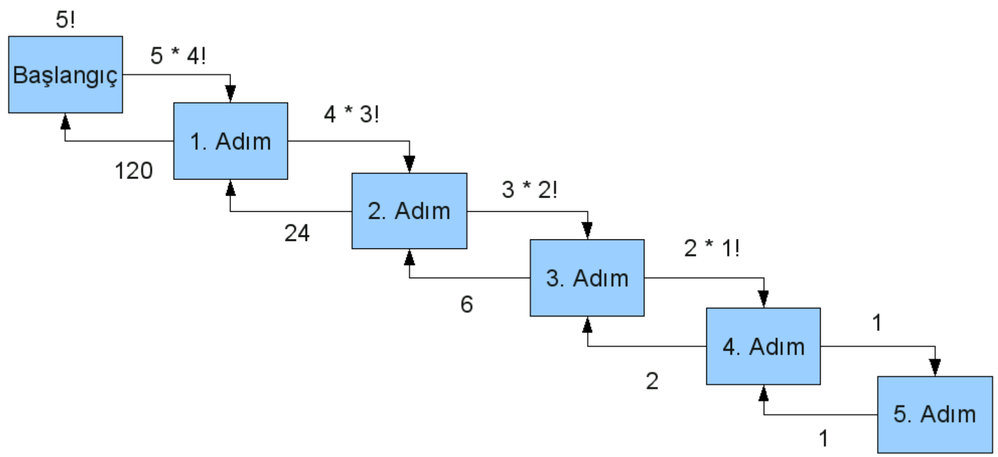
- Rekürsif adım çok daha fazla rekürsif çağrı ile sonuçlanır.

Özyinelemeli (Recursion) Fonksiyonlar

- Durma koşulu olmalıdır.
- Küçük problemlerin daha küçük dizileri temel duruma (base case) yakınsamalıdır.
- Çoğu zaman iteratif kod yazmaktan daha kısa ve kolaydır.
- Benzer alt görevlerin kullanımında daha kullanışlı olurlar. Sıralama, arama ve gezinme problemleri bu duruma örnek olarak gösterilebilir.

Özyinelemeli (Recursion) Fonksiyonlar

- $n! = \begin{cases} 1 & n \leq 1 \\ n.(n-1)! & n > 1 \end{cases}$



Özyinelemeli (Recursion) Fonksiyonlar

Rekürsif Yaklaşım

- Temel durum (base case) ulaşınca durur.
- Her rekürsif çağrı ekstra bellek alanı kullanır.
- Eğer sonsuz rekürsif çağrı yapılırsa; bellek taşma hatası alınır (stack overflow).
- Bazı problemlerin çözümü rekürsif olarak daha kolay ifade edilebilir.

İteratif Yaklaşım

- Bir koşulun yanlış olması durumunda durur.
- Her bir iterasyon ekstra bellek alanı gerektirmez.
- Ekstra bellek alanı gerektirmediğinden sonsuz döngüler sonsuza kadar devam eder.
- İteratif çözümler rekürsif çözümler kadar açık olmayabilir.

Sıralama Algoritmaları

- Sıralama algoritmaları, verilerin belirli bir düzene (genellikle küçükten büyüğe veya büyükten küçüğe) göre yerleştirilmesini sağlar.

- Arama işlemlerinin daha verimli yapılabilmesi için genellikle verinin sıralı olması tercih edilir.

Kabarcık Sıralama	Seçmeli Sıralama
Eklemeli Sıralama	Birleştirmeli Sıralama
Hızlı Sıralama	Yığın Sıralama

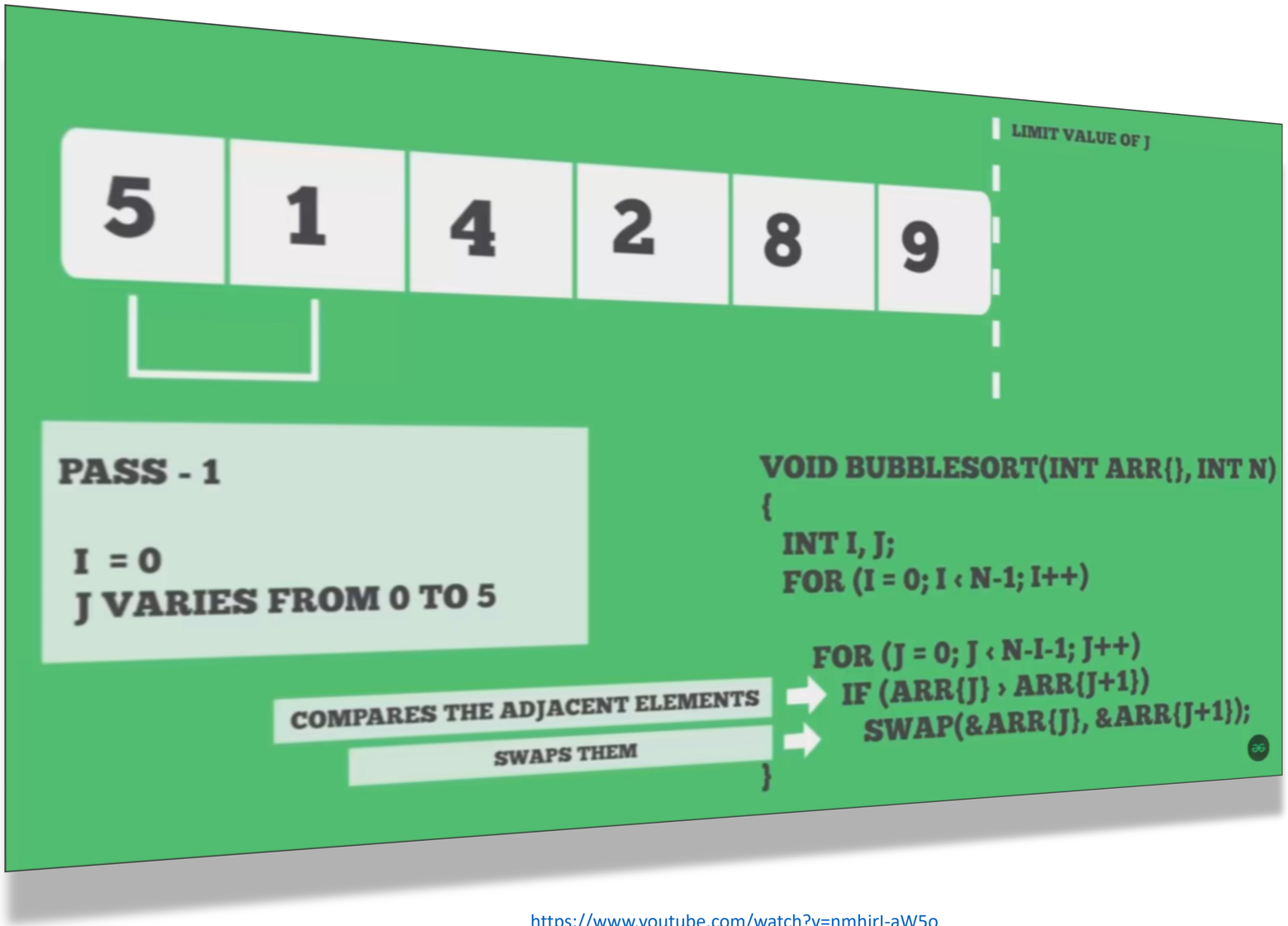
Kabarcık Sıralama

- Sıralama işlemi için komşu elemanı kullanan bir algoritma yapısına sahiptir.

- Eğer komşular doğru sırada değillerse yer değiştirme işleminin yapılması şeklinde sıralama yapılır.

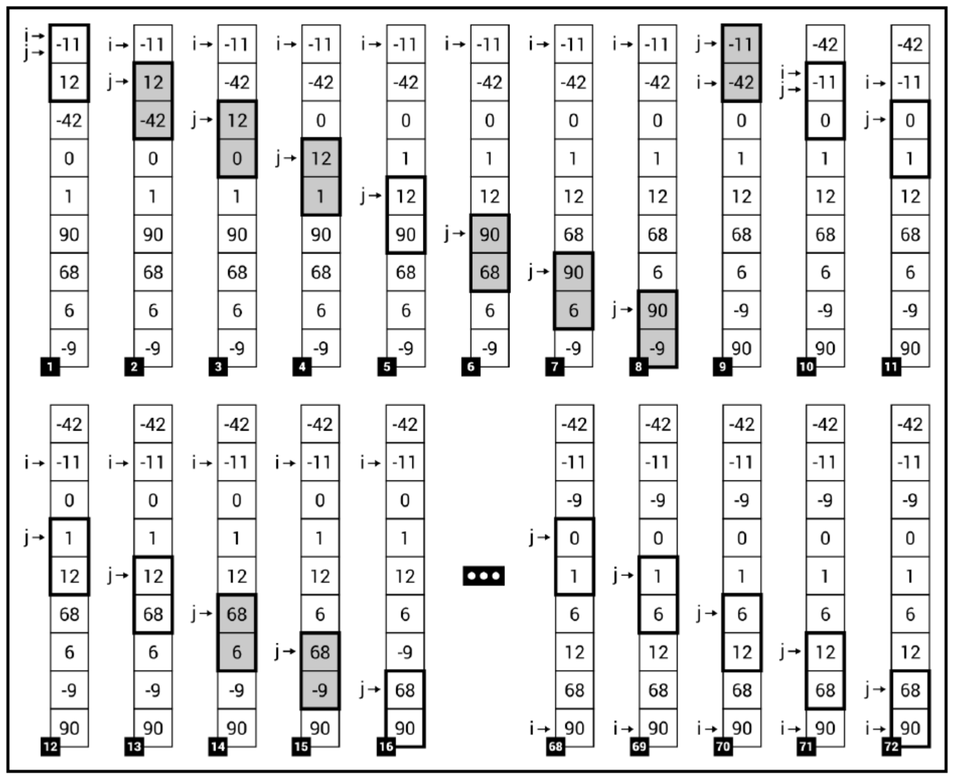
6 5 3 1 8 7 2 4

Kabarcık Sıralama



https://www.youtube.com/watch?v=nmhjrI-aW5o

Kabarcık Sıralama

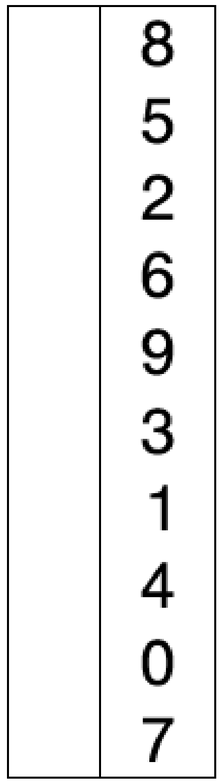


Kabarcık Sıralama

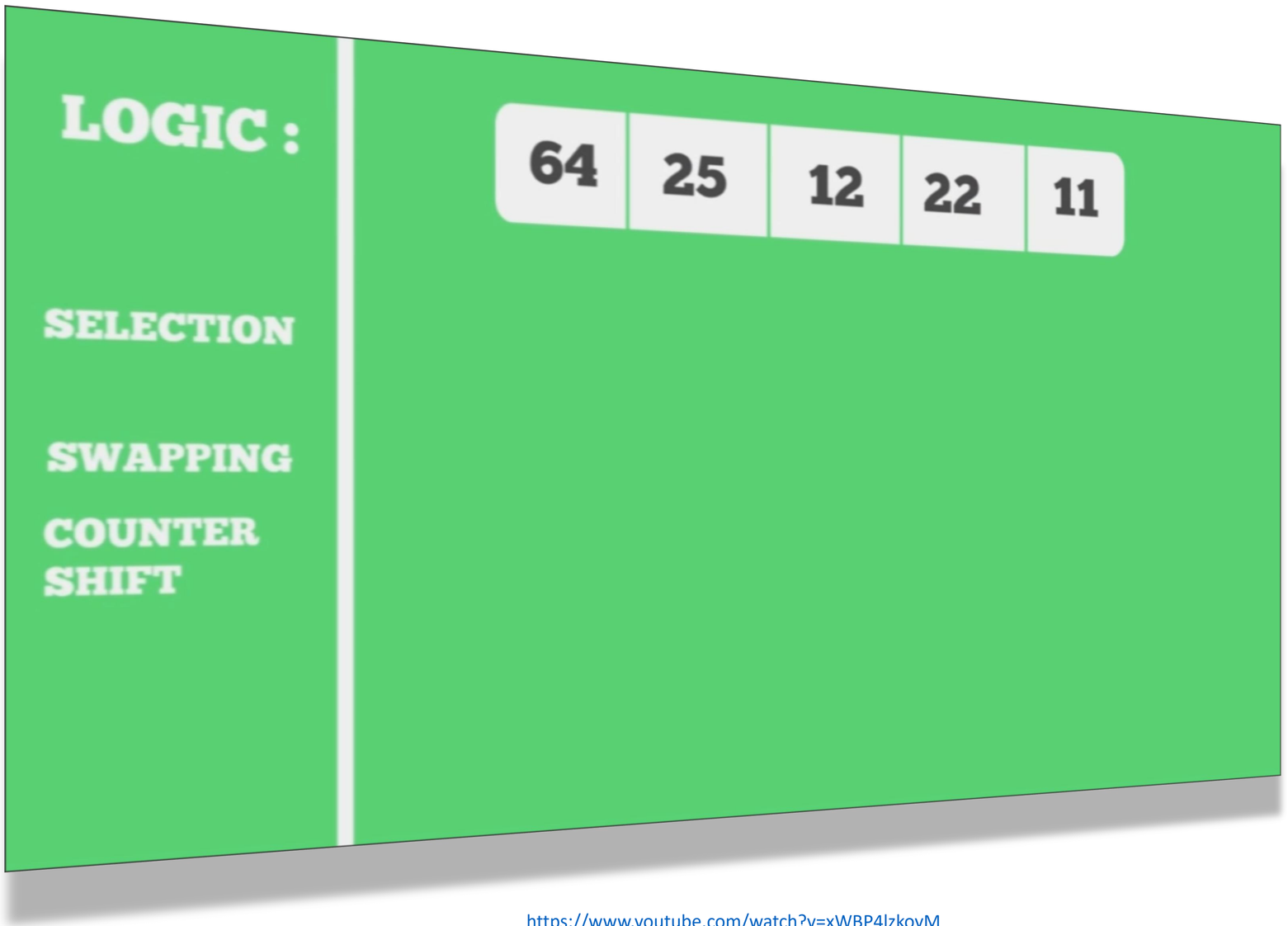
```
procedure bubbleSort(A : list of sortable items)
    n := length(A)
    repeat
        swapped := false
        for i := 1 to n-1 inclusive do
            /* if this pair is out of order */
            if A[i-1] > A[i] then
                /* swap them and remember something changed */
                swap(A[i-1], A[i])
                swapped := true
            end if
        end for
    until not swapped
end procedure
```

Seçmeli Sıralama

- En basit sıralama algoritmalarından bir tanesidir.
- Algoritma diziyi sıralı ve sırasız olmak üzere iki parçaya böler.
- Devam eden iterasyonlarda, algoritma sırasız parçadaki en küçük elemanı bulur ve ilgili elemanı sırasız parçanın ilk elemanı olarak atar.

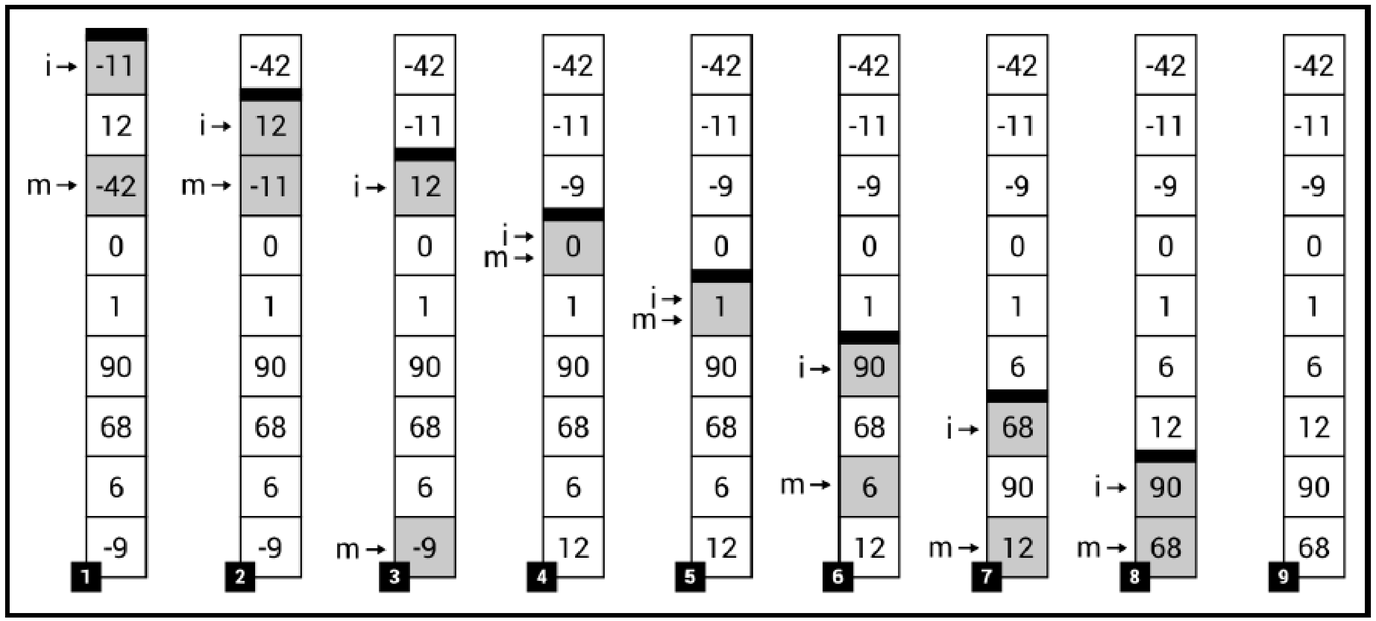


Seçmeli Sıralama



https://www.youtube.com/watch?v=xWBP4lzkoyM

Seçmeli Sıralama



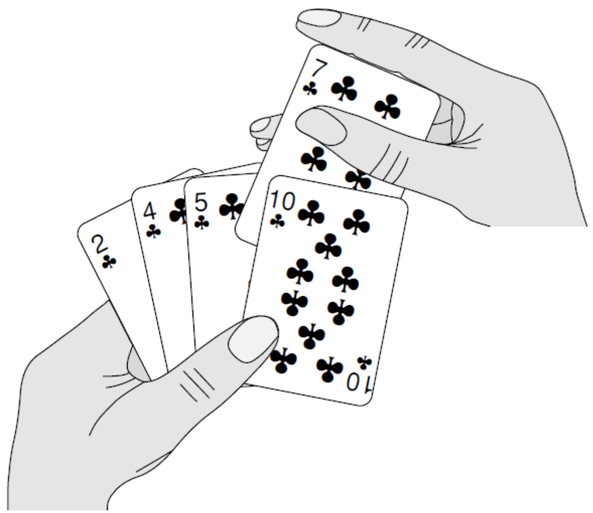
Seçmeli Sıralama

```
selectionSort(array, size)
  repeat (size - 1) times
  set the first unsorted element as the minimum
  for each of the unsorted elements
    if element < currentMinimum
      set element as new minimum
  swap minimum with first unsorted position
end selectionSort
```

Eklemeli Sıralama

- Tek boyutlu dizileri sıralamak üzere kullanılan bir diğer algoritma Insertion sort (eklemeli ya da sokuşturma sıralama algoritması) algoritmasıdır.

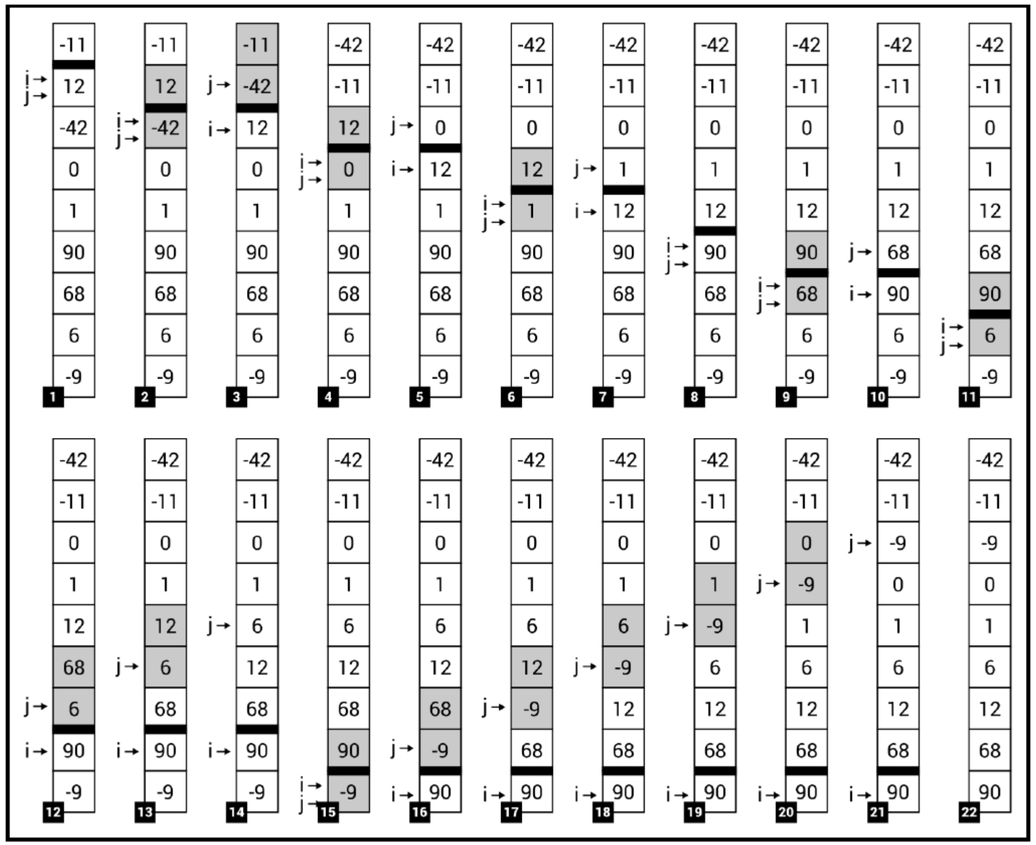
- Selection sort algoritmasında olduğu gibi dizi sıralı ve sırasız olmak üzere iki parçaya bölünür.



Eklemeli Sıralama

https://www.youtube.com/watch?v=OGzPmgsI-pQ

Eklemeli Sıralama



Eklemeli Sıralama

```
i ← 1
while i < length(A)
    j ← i
    while j > 0 and A[j-1] > A[j]
        swap A[j] and A[j-1]
        j ← j - 1
    end while
    i ← i + 1
end while
```

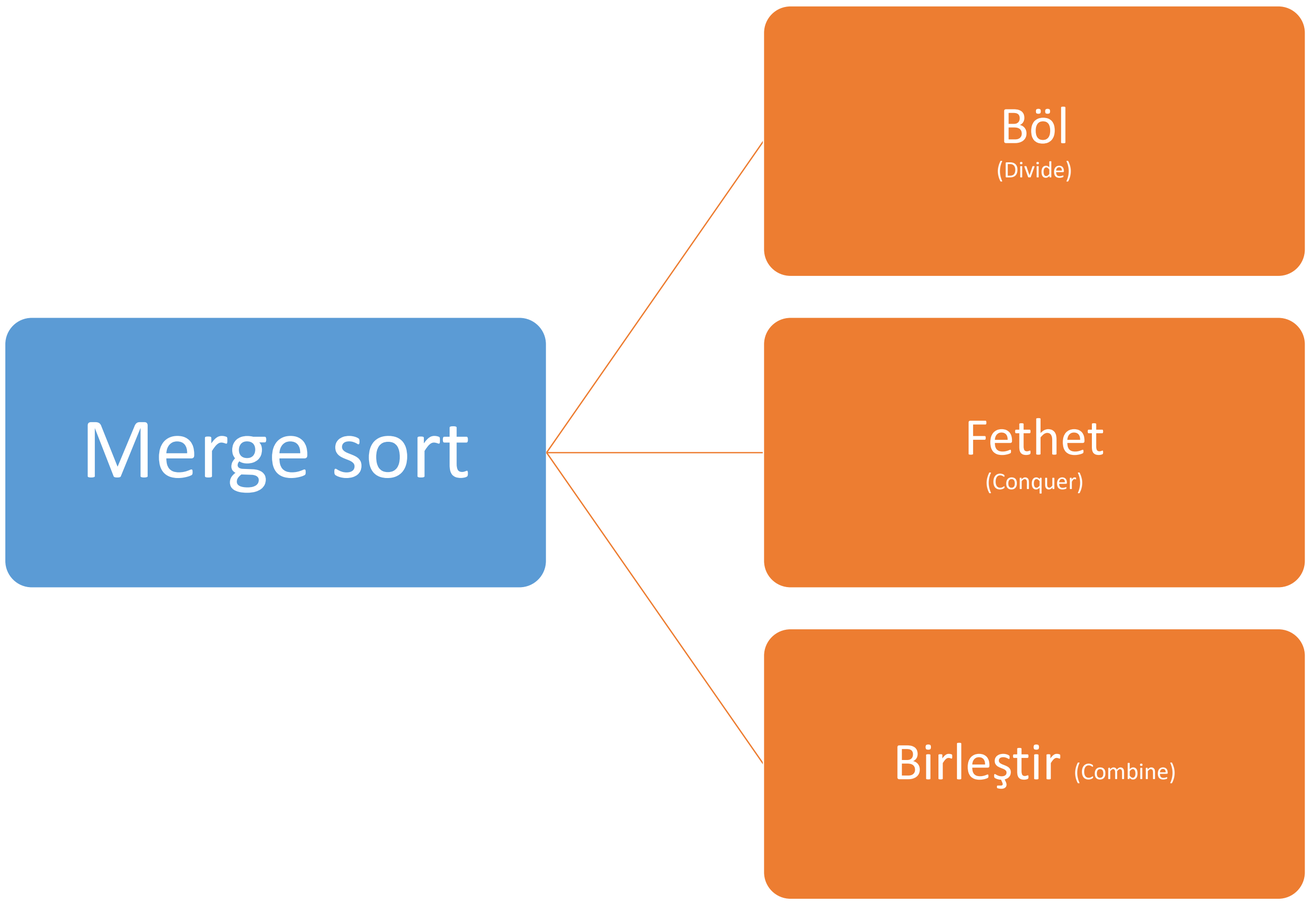
Birleştirmeli Sıralama

- Merge sort (birleştirme sıralama) algoritması böl ve fethet stratejisini uygular.
- Merge sort algoritması önce diziyi eşit parçalara böler ve daha sonra bu parçaları sıralı bir şekilde birleştirir.
- Parçaları sürekli yarıya bölecek şekilde çalışan özyinelemeli bir algoritmadır.

Birleştirmeli Sıralama

- Bölme işlemi eleman kalmayınca ya da tek bir eleman kalıncaya kadar devam eder.
- Eğer dizi boş ya da tek bir eleman var ise, ilgili parça temel koşul (base case) dikkate alınarak sıralanır.
- Eğer dizi birden fazla elemana sahipse, dizi parçalara bölünür ve rekürsif olarak merge sort çağrısı yapılır.

Birleştirmeli Sıralama



Birleştirmeli Sıralama

Birleştirmeli Sıralama

- Bölme işlemi için orta (mid) elemanın bulunması gerekir ve bu elemanda aşağıdaki gibi bulunur:

$$mid = low + \frac{(high - low)}{2} \; veya \; \frac{low + high}{2}$$

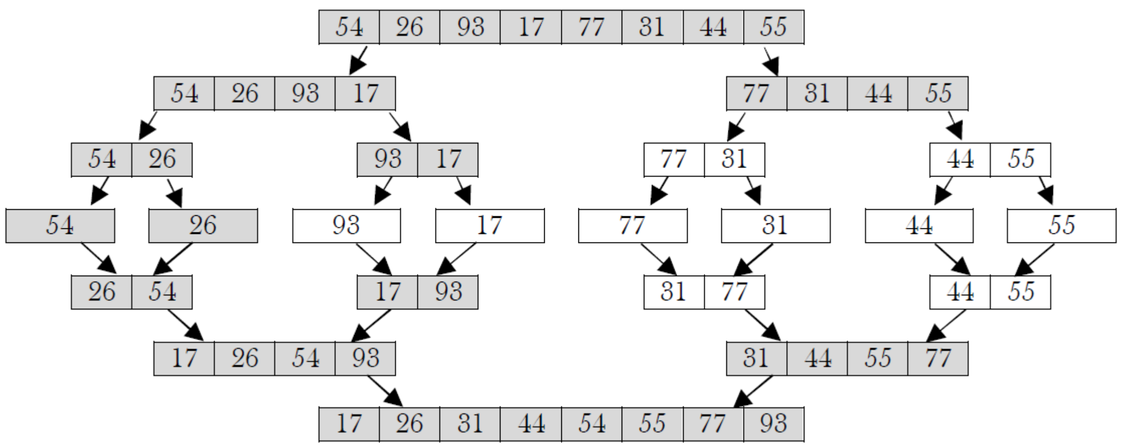
Birleştirmeli Sıralama

Fethet (Conquer): Bölünen alt diziler rekürsif bir şekilde sıralanır.

Birleştir (Combine): Sıralanmış iki alt dizinin birleştirilmesi sürecidir.

Bu noktada A[left … right] şeklide dizinin ilk elemanları dikkate alınarak sıralama işlemi gerçekleştirilir.

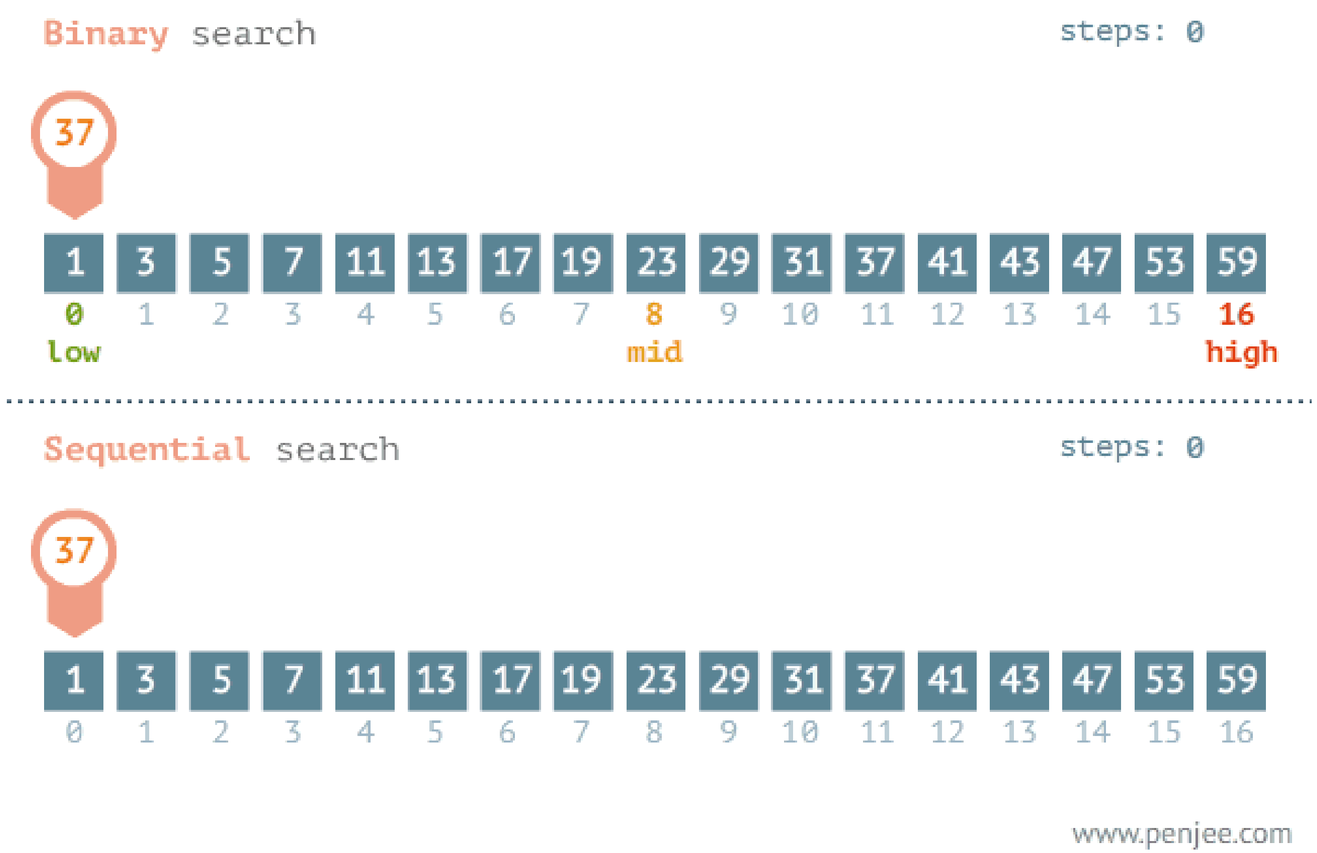
Birleştirmeli Sıralama



Birleştirmeli Sıralama

```
MergeSort(arr, left, right):
  if left > right
     return
  mid = (left+right)/2
  mergeSort(arr, left, mid)
  mergeSort(arr, mid+1, right)
  merge(arr, left, mid, right)
end
```

Arama Algoritmaları



Teşekkürler

ZAFER CÖMERT
Öğretim Üyesi