Kabarcık/Baloncuk Sıralaması

→ Bubble Sort algoritması, bir dizi elemanını sıralamak için kullanılan bir karşılaştırmalı sıralama algoritmasıdır. Adını, sıralama işlemi sırasında elemanların yukarı doğru hareket etmeleri ve bir baloncuk gibi kabarcıklar oluşturması nedeniyle almıştır.

1- İlk Adım: Başlangıçta, elemanları sıralanacak olan diziyi alırız.

2- İkinci Adım: İki ardışık elemanı karşılaştırırız. Eğer elemanların sırası yanlışsa, yani büyük olan eleman küçük olan elemandan önce geliyorsa, bu iki elemanın yerini değiştiririz.

3- Üçüncü Adım: Bu işlemi dizinin başından sonuna kadar tekrar ederiz. Böylece, en büyük eleman dizinin en sonuna yerleşir.

4- Dördüncü Adım: Şimdi, dizinin son elemanını dikkate almayız ve aynı işlemi dizinin geri kalan elemanları için tekrar ederiz.

5- Beşinci Adım: Bu adımları dizide sıralama yapılmayana kadar tekrar ederiz. Yani, hiçbir elemanın yer değiştirmesi gerekmeyecek duruma gelene kadar işlemi devam ettiririz.

Zaman Karmaşıklığı: Bubble Sort algoritmasının zaman karmaşıklığı O(n^2) olarak bilinir, çünkü en kötü durumda (dizi tersten sıralı olduğunda) tüm elemanları karşılaştırmak ve yer değiştirmek için işlemler gerçekleştirir. Ancak, en iyi durumda (dizi zaten sıralı olduğunda), elemanların yer değiştirmesi gerektirmediği için zaman karmaşıklığı O(n) olabilir.

Not: Bubble Sort algoritması, diğer sıralama algoritmalarına göre daha yavaş çalışır ve büyük veri kümeleri için pek tercih edilmez. Ancak, nispeten basit bir algoritma olduğu için öğrenme aşamasında sıklıkla kullanılır. Ayrıca, küçük veri kümeleri için hala etkili bir seçenek olabilir.

Örnek:

Dizi: [5, 3, 8, 4, 2]

1. Adım:

İlk eleman (5) ile ikinci eleman (3) karşılaştırılır. 5, 3'ten büyük olduğu için elemanlar yer değiştirilir ve dizi şöyle olur:

[5, 3, 8, 4, 2]

↑ ↑

1 2

→ Yeni hali [3, 5, 8, 4, 2].

2. İkinci adım:

İkinci eleman (5) ile üçüncü eleman (8) karşılaştırılır. 5, 8'den küçük olduğu için elemanlar yer değiştirmez ve dizi şöyle kalır.

[3, 5, 8, 4, 2]

↑ ↑

2 3

→ Yeni hali : [3, 5, 8, 4, 2].

3. Üçüncü Adım

Üçüncü eleman (8) ile dördüncü eleman (4) karşılaştırılır. 8, 4'ten büyük olduğu için elemanlar yer değiştirilir ve dizi şöyle olur.

[3, 5, 8, 4, 2]

↑ ↑

3 4

→ Yeni hali: [3, 5, 4, 8, 2].

4. Dördüncü Adım

Dördüncü eleman (8) ile beşinci eleman (2) karşılaştırılır. 8, 2'den büyük olduğu için elemanlar yer değiştirilir ve dizi şöyle olur.

[3, 5, 4, 8, 2]

↑ ↑

4 5

Yeni hali: [3, 5, 4, 2, 8].

Hatırlatma: İç döngü tamamlandıktan sonra, en büyük eleman (8) doğru konumuna yerleşmiştir. İç döngü bir kez daha çalıştırılır ve bu kez son eleman hariç ([3, 5, 4, 2]) elemanlar karşılaştırılır.

Tekrar:

Tüm elemanlar doğru konumlarına yerleşene kadar devam eder. Sonuç olarak, sıralanmış dizi şöyle olur: [2, 3, 4, 5, 8].

Pseudocode Kod:

```
fonksiyon bubbleSort(dizi):
  n = uzunluk(dizi)
  for i = 0 to n-1
    for j = 0 to n-i-1
      if arr[j] > arr[j+1]
        swap(arr[j], arr[j+1])
```

Python ile Yazarsak:

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n-1):
        for j in range(n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
    return arr
```