Hızlı Sıralama Algoritması

→ Quick Sort, yaygın olarak kullanılan hızlı bir sıralama algoritmasıdır. Bu algoritma, böl ve fethet (divide and conquer) stratejisini kullanır. Diziyi parçalara ayırarak sıralama işlemini gerçekleştirir.

1- İlk Adım: Bir pivot elemanı seçmemiz gerekiyor. Genellikle, dizinin ortasındaki bir elemanı veya rastgele bir elemanı pivot olarak seçebiliriz. Pivot elemanı, diziyi ikiye bölmek için bir referans noktası olarak kullanılır.

2- İkinci Adım: Seçilen pivot elemanıyla diziyi ikiye böleriz. Pivot elemanından daha küçük olanlar sol tarafa, pivot elemanından daha büyük olanlar ise sağ tarafa yerleştirilir. Bu işlem, dizinin pivot elemanı etrafında bir bölünme (partition) yapılmasını sağlar.

3- Üçüncü Adım: Bu aşamada, sol ve sağ tarafa bölünmüş olan alt diziler üzerinde aynı işlemi tekrarlarız. Yani, her bir alt diziyi ayrı ayrı ele alırız ve pivot elemanını seçerek alt diziyi tekrar bölerek sıralama işlemini yaparız.

4- Dördüncü Adım: Bölme işlemi, alt diziler belirli bir boyuta ulaşana kadar veya sıralanmış bir dizi elde edene kadar devam eder. Genellikle alt dizilerin boyutu 1 veya 0 olduğunda işlem sona erer. Bu durumda, alt diziler sıralanmış haldedir ve birleştirme aşamasına geçilir.

5- Beşinci Adım: İşlem sonunda, tüm alt diziler sıralanmış olur. Birleştirme (merge) adımı, sıralanmış alt dizileri pivot elemanıyla birleştirerek tamamen sıralanmış bir dizi elde etmemizi sağlar.

6- Son Adım: Bu adımlar, tüm alt diziler sıralanana kadar ve nihayetinde tüm dizi sıralanmış hale gelene kadar tekrarlanır. Sonunda, başlangıçta verilen dizinin tüm elemanları sıralanmış olur.

Zaman Karmaşıklığı: Quick Sort algoritmasının ortalama zaman karmaşıklığı O(n log n) olarak bilinir. Burada n, sıralanacak dizinin eleman sayısıdır. En iyi durumda, yani pivot elemanı her seferinde dizinin tam ortasında seçildiğinde, zaman karmaşıklığı O(n log n) olarak gerçekleşir. Ancak en kötü durumda, yani pivot elemanı her seferinde dizinin en küçük veya en büyük elemanı olarak seçildiğinde, zaman karmaşıklığı O(n^2) olabilir.

Not: Quick Sort, diğer sıralama algoritmalarına göre genellikle daha hızlıdır ve pratikte iyi bir performans sergiler. Ancak en kötü durumdaki zaman karmaşıklığına dikkat etmek önemlidir. Bu nedenle, pivot elemanının seçimi ve bölme işleminin dengeli bir şekilde yapılması, Quick Sort algoritmasının etkin bir şekilde kullanılması için önemlidir.

Örnek:

Dizi: [3, 8, 1, 9, 6, 5, 2, 7, 4]

1. Adım:

Pivot(Son Eleman) 4

Sol Alt Dizi(Pivottan Küçük Elemanlar): 3, 1, 2

Sağ Alt Dizi(Pivottan Büyük Elemanlar): 8, 9, 6, 5, 7

2. Adım:

Sol Alt Dizi: [3, 1, 2]

Pivot(Son Eleman): 2

Sol Alt Dizi(Pivottan Küçük Elemanlar): 1

Sağ Alt Dizi(Pivottan Büyük Elemanlar): 3

→ İlk seçilen pivotun sol kısmındaki alt diziler 1 eleman uzunluğuna ulaştığı için burayla işimiz bitti sıra sağ alt dizilerde.

3. Adım:

Sağ Alt Dizi: 8, 9, 6, 5, 7

Pivot(Son Eleman): 7

Sol Alt Dizi(Pivottan Küçük Elemanlar): 6, 5

Sağ Alt Dizi(Pivottan Büyük Elemanlar): 8, 9

4. Adım:

Sol Alt Dizi: 6, 5

Pivot(Son Eleman): 5

Sol Alt Dizi(Pivottan Küçük Elemanlar):

Sağ Alt Dizi(Pivottan Büyük Elemanlar): 6

5. Adım:

Sağ Alt Dizi: 8, 9

Pivot(Son Eleman): 9

Sol Alt Dizi(Pivottan Küçük Elemanlar): 8

Sağ Alt Dizi(Pivottan Büyük Elemanlar):

→ İlk seçilen pivotun sağ kısmındaki alt diziler 1 eleman uzunluğuna ulaştığı için burayla da işimiz bitti.

6. Adım:

→ En sol alt diziden başlayarak aralara pivotları yerleştirmek şartıyla sağ alt dizilere doğru ilerlemek.

Dizi: [1, 2, 3, 4, 5, 6, 7, 8, 9]

Pseudocode Code:

```
function quickSort(arr, low, high)
    if low < high then
        pivotIndex = partition(arr, low, high)  // Diziyi bölme işlemi
        quickSort(arr, low, pivotIndex - 1)  // Sol alt diziyi sıralama
        quickSort(arr, pivotIndex + 1, high)  // Sağ alt diziyi sıralama


function partition(arr, low, high)
    pivot = arr[high]  // Pivot elemanını seçme
    i = low - 1  // Index belirleme
    for j = low to high - 1
        if arr[j] <= pivot then
            i = i + 1
            swap(arr[i], arr[j])  // Elemanları yer değiştirme
    swap(arr[i + 1], arr[high])  // Pivot elemanını doğru konuma yerleştirme
    return i + 1
```

Python Kodu:

```
def quickSort(arr):
    if len(arr) <= 1:
        return arr
    else:
        pivot = arr[0]  # Pivot elemanını seçme
        less = [x for x in arr[1:] if x <= pivot]  # Pivot'tan küçük elemanlar
        greater = [x for x in arr[1:] if x > pivot]  # Pivot'tan büyük elemanlar
        return quickSort(less) + [pivot] + quickSort(greater)  # Sıralama ve birleştirme

arr = [5, 9, 3, 2, 8, 1, 6, 4, 7]
sorted_arr = quickSort(arr)
print(sorted_arr)
```