Algoritma pengurutan

1.Selection short

Selection Sort algoritma adalah algoritma yang digunakan untuk melakukan sortir nilai pada array/list dengan membagi array menjadi dua bagian. Bagian pertama adalah sorted value (bagian yang telah disortir) dan kedua adalah unsorted value. Sorted value akan diinisialisasi dengan nilai kosong. Selanjutnya kita akan mencari nilai minimum pada bagian unsorted untuk selanjutnya disimpan pada bagian sorted. Contoh code pada python:

angka = [1, 4, 3, 9, 6, 5]

#selection sort

def selection\_short(array):

    for i in range(len(array)-1):

        min\_idx = 1

        for j in range(i + 1, len(array)):

            if array[j] > array[min\_idx]:

               min\_idx = j

    array[i], array[min\_idx] = array[min\_idx], array[i]

selection\_short(angka)

print(angka)

Pada gambar di atas kita memiliki array/list [3, 4, 1, 6, 2].

Nilai pertama pada array akan kita jadikan nilai minimum yang akan kita bandingkan dengan nilai pada index selanjutnya. Jika kondisi nilai pada array lebih kecil dari nilai minimum maka tukar nilainya/swap.

Looping ini akan terus dilakukan sampai range pada looping terpenuhi.

2.Bubble sort

Bubblesort adalah mengurutkan baris angka/string dari urutan terkecil sampai yang terbesar atau sebaliknya.Misal kita memiliki list/baris angka sebagai berikut: [1, 4, 3, 9, 6, 5].

Dengan bubblesort kita ingin mengubah baris angka kita menjadi berurutan dari angka yang terkecil ke angka terbesar seperti berikut: [1, 3, 4, 5, 6, 9].

Kita memiliki 4 angka dalam baris, langkah pertama adalah membandingkan angka pertama dengan angka kedua dan apabila angka pertama lebih besar dari angka kedua, tukar posisi angka pertama menjadi angka kedua dan angka kedua menjadi angka pertama "swap".

Lalu kita bandingkan angka kedua dengan angka ke tiga, jika angka kedua lebih besar dari angka ketiga swap.

Langkah ini akan terus berulang sampai baris angka berurutan dari yang terkecil ke terbesar. Contoh code pada python:

#bubble sort

angka = [1, 4, 3, 9, 6, 5]

def bubble\_sort(array):

    for i in range(len(array)-1):

        for j in range(len(array)-1-i):

            if array[j] > array[j+1]:

               array[j], array[j+1] = array[j+1], array[j]

bubble\_sort(angka)

print(angka)

3.Insertion sort

Adalah algoritma yang melakukan pengurutan dengan membandingkan elemen satu dengan elemen lainnya dalam sebuah list. Elemen yang dibandingkan akan ditempatkan ke posisi yang sesuai (urut) pada list.

Analoginya seperti mengurutkan kumpulan kartu. Setiap kartu yang kalian ambil, kalian bandingkan terlebih dahulu ke kumpulan kartu yang sudah diurutkan. Dan ketika tahu urutan ke berapa, kalian selipkan kartu itu ke tumpukan kartu agar urut. Contoh code pada python:

#insertion short

angka = [1, 4, 3, 9, 6, 5]

def insertion\_sort(array):

    for i in range(1, len(array)):

        key = array[i]

        j = i - 1

        while j >= 0 and array[j] > key:

            array[j + 1] = array[j]

            j = j - 1

        array[j+1] = key

insertion\_sort(angka)

print(angka)

4.Merge sort

Algoritma merge sort merupakan salah satu pengurutan dengan metode memecah data kemudian mengolah untuk diselesaikan pada setiap bagian dan menggabungkan kembali sehingga data tersebut berhasil tersusun. Merge sort dalam menyelesaikan pengurutan membutuhkan fungsi rekursif.

Contoh code pada python:

#merge sort

angka = [1, 4, 3, 9, 6, 5]

def merge(left, right, arr):

    i, j, k = 0, 0, 0

    while i < len(left) and j < len(right):

        if left[i] < right[j]:

            arr[k] = left[i]

            i = i + 1

        else:

            arr[k] = right[j]

            j = j + 1

        k = k + 1

    while i < len(left):

        arr[k] = left[i]

        k = k + 1

        i = i + 1

    while j < len(right):

        arr[k] = right[j]

        k = k + 1

        j = j + 1

def merge\_sort(array):

    if len(array) > 1:

        mid = len(array) // 2

        leftarr = array[:mid]

        rightarr = array[mid:]

        merge\_sort(leftarr)

        merge\_sort(rightarr)

        merge(leftarr, rightarr, array)

merge\_sort(angka)

print(angka)

5.Quick sort

Algoritma quick sort ini cara kerjanya berprinsip pada penekatan divide and conquer yakni dengan memilih satu elemen sebagai elemen pivot dan mempartisi array sehingga sisi kiri pada pivot mempunyai semua elemen dengan nilai yang lebih kecil dibandingkan dengan elmen pivot dan pada sisi kanan mempunyai semua elemen dengan nilai yang lebih besar dibandingkan dengan nilai elemen pivot.

Contoh code pada python:

#quick sort

angka = [1, 4, 3, 9, 6, 5]

def quick\_sort(array):

    quick\_sort\_rec(array, 0, len(array)-1)

def quick\_sort\_rec(array, start, end):

    if start < end:

        pivot\_idx = partition(array, start, end)

        quick\_sort\_rec(array, start, pivot\_idx-1)

        quick\_sort\_rec(array, pivot\_idx+1, end)

quick\_sort(angka)

print(angka)