Nama : Ahmad Haziq Mu'izzaddin Wafiq

Kelas : Pengembangan Perangkat Lunak(PPL)

ALGORITMA PENGURUTAN

Pengurutan atau **"Sorting"** adalah suatu proses menyusun kembali data yang sebelumnya telah disusun dengan suatu pola tertentu, sehingga tersusun secara teratur menurut aturan tertentu (untuk data yang bertipe numerik atau karakter).

Dua Macam Pengurutan

- Ascending (urut naik) merupakan pengurutan dari angka yang nilainya lebih kecil kemudian menuju ke nilainya yang lebih besar.
- Descending (urut turun) adalah sebaliknya, yaitu pengurutan dari nilainya yang lebih besar kemudian menuju ke nilainya yang lebih kecil.

Sebagai contoh misalkan diberikan data berupa bilangan berikut ini:

(3 9 1 4 0 2)

Hasil sorting:

- Ascending adalah 0 1 2 3 4 9
- Descending adalah 9 4 3 2 1 0.

Macam-Macam Sorting

1. Selection Sort

Memindahkan elemen dengan cara membandingkan elemen sekarang dengan elemen yang berikutnya sampai dengan elemen terakhir . Jika ditemukan elemen lain yang lebih kecil / lebih besar dari elemen sekarang maka dicatat posisinya dan kemudian ditukar dan begitu seterusnya.

Proses Selection Sort:

- Ascending → Elemen yang paling besar diletakkan di akhir.
  → Elemen yang paling kecil diletakkan di awal.

- Descending → Elemen yang paling kecil diletakkan di akhir.
  → Elemen yang paling besar diletakkan di awal.

```
# # Selection Sort

# Membuat function yang menerima parameter array
def selection_sort(array):
    # Membuat for perulangan sebanyak isi value dari array
    for i in range(len(array)-1):
        # Membuat variabel index_awal dan mengisi dengan i
        index_awal = i
        # Membuat for perulangan dimana i + 1, dan loop sebanyak isi dari array
        for j in range(i + 1, len(array)):
            # Jika array[j] lebih kecil dari array [index_awal]
            if array [j] < array[index_awal]:
                # Maka index awal = j
                index_awal = j
                # Mengubah value
        array[i], array[index_awal] = array[index_awal], array[i]

angka = [1, 4, 3, 9, 6, 5]
selection_sort(angka)

print(angka)
```

2.Bubble Sort

Memindahkan element sekarang dengan elemen berikutnya , jika elemen sekarang itu lebih besar / lebih kecil dari elemen berikutnya maka di tukar (berpindah posisi).

Proses Bubble Sort

a. Ascending

Data yang paling awal dibandingkan dengan data berikutnya jika ternyata lebih besar maka tukar.

Data yang paling akhir dibandingkan dengan data sebelumya jika ternyata lebih kecil maka tukar.

b. Descending

Data yang paling awal dibandingkan dengan data berikutnya jika ternyata lebih kecil maka tukar.

Data yang paling akhir dibandingkan dengan data sebelumya jika ternyata lebih besar maka tukar.

```
# Bubble Sort

angka = [9, 5, 8, 6, 7, 4, 3, 1, 2]

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

Pengurutan yang dilakukan dengan cara membandingkan dengan data ke-2 sampai data terakhir. Jika ditemukan data yang lebih kecil atau lebih besar maka data tersebut disisipkan kedepan sesuai posisi yang seharusnya.

```
# insert sort

angka = [9, 5, 8, 6, 7, 4, 3, 1, 2]

def insertion_sort(array):
    for i in range(1, len(array)):
        value = array[i]
        j = i - 1
        while j >= 0  and array [j]> value:
            array [j + 1] = array[j]
            j = j - 1
        array [j + 1] = value

insertion_sort(angka)
print (angka)
```

4.Merge Sort

Merupakan proses pengurutan data yang menggunakan merging dua buah vector. Pada proses merge sort, data dibuat sepasang-sepasang, yang terdiri dari dua elemen. Jika N ganjil, maka ada satu vector yang terdiri dari 1 elemen. Lalu kemudian data tersebut di merging sampai terurut.

```
# Merge sort

angka = [9, 5, 8, 6, 7, 4, 3, 1, 2]

def merge_sort(array):
    if len (array) > 1:
        mid = len (array) // 2
        leftarray = array[:mid]
        rightarray = array[mid:]
        merge_sort(leftarray)
        merge_sort(rightarray)
        merge(leftarray, rightarray, array)

def merge(left, right, array):
    i, j, k = 0, 0, 0
    while i < len(left) and j < len (right):
        if left[i] < right[j]:
            array[k] = left[i]
            i = i + 1
        else:
            array[k] = right[j]
            j = j + 1
        k = k + 1

    while i < len(left):
        array[k] = left[i]
        k = k + 1
        i = i + 1

    while j < len(right):
        array[k] = right[j]
        k = k + 1
        j = j + 1

merge_sort=(angka)
print(angka)
```

5. Quick Sort

Merupakan proses penyusunan elemen yang membandingkan suatu elemen (pivot) denan elemen yang lain, dan menyusunnya sedemikian rupa sehingga elemen –elemen lain yang lebih kecil dari pivot terletak disebelah kiri pivot. Dan elemen yang lebih besar dari pivot terletak disebelah kanan pivot.

Dengan demikian akan terbentuk dua sublist, yang terletak disebelah kanan dan kiri pivot. Lalu pada sublist kiri dan kanan itu kita anggap sebuah list baru, dan kita kerjakan proses yang sama seperti yang sebelumnya. Demikian seterusnya sampai tidak terdapat sublist lagi.

```
# Quick Sort

angka = [9, 5, 8, 6, 7, 4, 3, 1, 2]

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)

def partition(array, start, end):
    pivot = array[end]
    i = start
    for j in range(start, end):
        if array[j] <= pivot:
            array[i], array[j] = array[j], array[i]
            i = i + 1
    array[i], array[end] = array[end], array[i]
    return i

quick_sort(angka)
print(angka)
```

Referensi : *http://risasisteminformasi.blogspot.com/2013/02/sorting-pengurutan.html*