1 Konsep

Kerjakan berikut dengan ketentuan, modulus satu digit terakhir NPM kalian dengan 2, hasil modulus adalah kode soal berikut ini. Tulis ringkasan atau penjelasan hal-hal berikut, dengan kata-kata kalian sendiri mengenai :

- **Kode Soal = 0**, Bubble Sort, pengertian, dan fungsi Bubble Sort
- **Kode Soal = 1**, Selection Sort, pengertian, dan fungsi Selection Sort

Konsep_1

Kode Soal 0

Pengartian

Bubble Sort adalah algoritma pengurutan sederhana yang berkerja dengan cara membandingkan dua data yang berdekatan dan menukar data tersebut jika data tersebut tidak berada pada urutan yang benar hingga terurut.

Fungsi

Fungsi Bubble Sort adalah untuk mengurutkan sebuah list/data menjadi data yang terurut secara ascending atau decending namun bubble sort tidak terlalu efisien untuk data yang besar karena membutuhkan waktu yang cukup lama.

2 Implementasi-1

Kerjakan berikut dengan ketentuan, modulus satu digit terakhir NPM kalian dengan 4, hasil modulus adalah kode soal berikut ini.

2.3 Kode Soal=2

Buat fungsi **Modifikasi Algoritma Sorting - Selection Sort**, untuk pengurutan secara **ascending**, dengan ketentuan :

- Seleksi/cari indeks data yang memiliki nilai maksimal dari suatu list (pencarian dilakukan mulai dariindeks-indeks akhir)
- Data yang memiliki nilai besar, diletakkan terlebih dahulu di indek-indeks akhir, sehingga pada akhir iterasi indeks-indeks akhir ditempati data yang memiliki nilai besar.
- Contoh Proses perbandingan yang dilakukan pada modifikasi algoritma selection sort ini dan outputyang harus dihasilkan pada fungsi yang akan dibuat, dapat dilihat pada Gambar 3

```
a=[10,9,1,3,8,7,11,0]
selectionSortAscending(a)
```

```
Data = [10, 9, 1, 3, 8, 7, 11, 0]
data[6] is the max value, swap data[6] and data[7],   New Data : [10, 9, 1, 3, 8, 7, 0, 11]
data[0] is the max value, swap data[0] and data[6],   New Data : [0, 9, 1, 3, 8, 7, 10, 11]
data[1] is the max value, swap data[1] and data[5],   New Data : [0, 7, 1, 3, 8, 9, 10, 11]
data[4] is the max value, swap data[4] and data[4],   New Data : [0, 7, 1, 3, 8, 9, 10, 11]
data[1] is the max value, swap data[1] and data[3],   New Data : [0, 3, 1, 7, 8, 9, 10, 11]
data[1] is the max value, swap data[1] and data[2],   New Data : [0, 1, 3, 7, 8, 9, 10, 11]
data[1] is the max value, swap data[1] and data[1],   New Data : [0, 1, 3, 7, 8, 9, 10, 11]
data[0] is the max value, swap data[0] and data[0],   New Data : [0, 1, 3, 7, 8, 9, 10, 11]
Sorted Data= [0, 1, 3, 7, 8, 9, 10, 11]
```

```
a=[10,7,6,5,3,1,0,-5]
selectionSortAscending(a)
```

```
Data = [10, 7, 6, 5, 3, 1, 0, -5]
data[0] is the max value, swap data[0] and data[7],   New Data : [-5, 7, 6, 5, 3, 1, 0, 10]
data[1] is the max value, swap data[1] and data[6],   New Data : [-5, 0, 6, 5, 3, 1, 7, 10]
data[2] is the max value, swap data[2] and data[5],   New Data : [-5, 0, 1, 5, 3, 6, 7, 10]
data[3] is the max value, swap data[3] and data[4],   New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[3] is the max value, swap data[3] and data[3],   New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[2] is the max value, swap data[2] and data[2],   New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[1] is the max value, swap data[1] and data[1],   New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[0] is the max value, swap data[0] and data[0],   New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
Sorted Data= [-5, 0, 1, 3, 5, 6, 7, 10]
```

Gambar 3: Selection Sort - Ascending

```
# 2 Implementasi-1
# Kode Soal 2

def selectionSortAscending(data):
    print('Data = ',data)
    for i in range(len(data)-1,-1,-1):
        idxMax = i
        for j in range(i-1,-1,-1):
            if data[idxMax] < data[j]:
                idxMax=j
        data[i],data[idxMax]=data[idxMax],data[i]
        print(f"data[{idxMax}] is the max value, swap data[{idxMax}] and data[{i}],  New Data : {data}")
```

```
    print('Sorted Data =',data)

a = [10,7,6,5,3,1,0,-5]
selectionSortAscending(a)
```

```
Data =  [10, 7, 6, 5, 3, 1, 0, -5]
data[0] is the max value, swap data[0] and data[7],  New Data : [-5, 7, 6, 5, 3, 1, 0, 10]
data[1] is the max value, swap data[1] and data[6],  New Data : [-5, 0, 6, 5, 3, 1, 7, 10]
data[2] is the max value, swap data[2] and data[5],  New Data : [-5, 0, 1, 5, 3, 6, 7, 10]
data[3] is the max value, swap data[3] and data[4],  New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[3] is the max value, swap data[3] and data[3],  New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[2] is the max value, swap data[2] and data[2],  New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[1] is the max value, swap data[1] and data[1],  New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
data[0] is the max value, swap data[0] and data[0],  New Data : [-5, 0, 1, 3, 5, 6, 7, 10]
Sorted Data = [-5, 0, 1, 3, 5, 6, 7, 10]
```

3 Implementasi-2

Kerjakan untuk semua mahasiswa.
Modifikasi algoritma Bubble Sort agar iterasi terhenti ketika tidak ada lagi data yang akan ditukar posisinya (semua data sudah berada di tempat yang tepat). Contoh Bubble Sort tanpa modifikasi dapat dilihat pada 5a, sedangkan modifikasi bubble sort dapat dilihat pada 5b.

```
b=[10,2,5,8,1,20,2,2,4]
sorting.modifiedBubbleSort(b)
```

```
iterasi ke- 1 jumlah iterasi- 8
1 =  [2, 10, 5, 8, 1, 20, 2, 2, 4]
2 =  [2, 5, 10, 8, 1, 20, 2, 2, 4]
3 =  [2, 5, 8, 10, 1, 20, 2, 2, 4]
4 =  [2, 5, 8, 1, 10, 20, 2, 2, 4]
5 =  [2, 5, 8, 1, 10, 20, 2, 2, 4]
6 =  [2, 5, 8, 1, 10, 2, 20, 2, 4]
7 =  [2, 5, 8, 1, 10, 2, 2, 20, 4]
8 =  [2, 5, 8, 1, 10, 2, 2, 4, 20]
iterasi ke- 2 jumlah iterasi- 7
1 =  [2, 5, 8, 1, 10, 2, 2, 4, 20]
2 =  [2, 5, 8, 1, 10, 2, 2, 4, 20]
3 =  [2, 5, 1, 8, 10, 2, 2, 4, 20]
4 =  [2, 5, 1, 8, 10, 2, 2, 4, 20]
5 =  [2, 5, 1, 8, 2, 10, 2, 4, 20]
6 =  [2, 5, 1, 8, 2, 2, 10, 4, 20]
7 =  [2, 5, 1, 8, 2, 2, 4, 10, 20]
iterasi ke- 3 jumlah iterasi- 6
1 =  [2, 5, 1, 8, 2, 2, 4, 10, 20]
2 =  [2, 1, 5, 8, 2, 2, 4, 10, 20]
3 =  [2, 1, 5, 8, 2, 2, 4, 10, 20]
4 =  [2, 1, 5, 2, 8, 2, 4, 10, 20]
5 =  [2, 1, 5, 2, 2, 8, 4, 10, 20]
6 =  [2, 1, 5, 2, 2, 4, 8, 10, 20]
iterasi ke- 4 jumlah iterasi- 5
1 =  [1, 2, 5, 2, 2, 4, 8, 10, 20]
2 =  [1, 2, 5, 2, 2, 4, 8, 10, 20]
3 =  [1, 2, 2, 5, 2, 4, 8, 10, 20]
4 =  [1, 2, 2, 2, 5, 4, 8, 10, 20]
5 =  [1, 2, 2, 2, 4, 5, 8, 10, 20]
iterasi ke- 5 jumlah iterasi- 4
1 =  [1, 2, 2, 2, 4, 5, 8, 10, 20]
2 =  [1, 2, 2, 2, 4, 5, 8, 10, 20]
3 =  [1, 2, 2, 2, 4, 5, 8, 10, 20]
4 =  [1, 2, 2, 2, 4, 5, 8, 10, 20]
```

(a) Bubble Sort Modifikasi

```
# 3 Implementasi-2

def modifiedBubbleSort(data):
    i = len(data)-1
    while i > 0 :
        print('Iterasi ke-',len(data)-i,'Jumlah Iterasi-',i)
        temp=0
        for j in range(i):
            if data[j+1] < data [j] :
                data[j],data[j+1]=data[j+1],data[j]
                temp+=1
            print(j+1,'=',data)
        if temp == 0 :
            i = 0
        i-=1

```

```
b = [10,2,5,8,1,20,2,2,4]
modifiedBubbleSort(b)
```

```
Iterasi ke- 1 Jumlah Iterasi- 8
1 = [2, 10, 5, 8, 1, 20, 2, 2, 4]
2 = [2, 5, 10, 8, 1, 20, 2, 2, 4]
3 = [2, 5, 8, 10, 1, 20, 2, 2, 4]
4 = [2, 5, 8, 1, 10, 20, 2, 2, 4]
5 = [2, 5, 8, 1, 10, 20, 2, 2, 4]
6 = [2, 5, 8, 1, 10, 2, 20, 2, 4]
7 = [2, 5, 8, 1, 10, 2, 2, 20, 4]
8 = [2, 5, 8, 1, 10, 2, 2, 4, 20]
Iterasi ke- 2 Jumlah Iterasi- 7
1 = [2, 5, 8, 1, 10, 2, 2, 4, 20]
2 = [2, 5, 8, 1, 10, 2, 2, 4, 20]
3 = [2, 5, 1, 8, 10, 2, 2, 4, 20]
4 = [2, 5, 1, 8, 10, 2, 2, 4, 20]
5 = [2, 5, 1, 8, 2, 10, 2, 4, 20]
6 = [2, 5, 1, 8, 2, 2, 10, 4, 20]
7 = [2, 5, 1, 8, 2, 2, 4, 10, 20]
Iterasi ke- 3 Jumlah Iterasi- 6
1 = [2, 5, 1, 8, 2, 2, 4, 10, 20]
2 = [2, 1, 5, 8, 2, 2, 4, 10, 20]
3 = [2, 1, 5, 8, 2, 2, 4, 10, 20]
4 = [2, 1, 5, 2, 8, 2, 4, 10, 20]
5 = [2, 1, 5, 2, 2, 8, 4, 10, 20]
6 = [2, 1, 5, 2, 2, 4, 8, 10, 20]
Iterasi ke- 4 Jumlah Iterasi- 5
1 = [1, 2, 5, 2, 2, 4, 8, 10, 20]
2 = [1, 2, 5, 2, 2, 4, 8, 10, 20]
3 = [1, 2, 2, 5, 2, 4, 8, 10, 20]
4 = [1, 2, 2, 2, 5, 4, 8, 10, 20]
5 = [1, 2, 2, 2, 4, 5, 8, 10, 20]
Iterasi ke- 5 Jumlah Iterasi- 4
1 = [1, 2, 2, 2, 4, 5, 8, 10, 20]
2 = [1, 2, 2, 2, 4, 5, 8, 10, 20]
3 = [1, 2, 2, 2, 4, 5, 8, 10, 20]
4 = [1, 2, 2, 2, 4, 5, 8, 10, 20]
```