Insertion Sort

Algoritma pengurutan insertion dapat di ilustrasikan seperti pengurutan pada kartu remi. Data di cek satu persatu mulai data ke dua sampai data terakhir. Jika di temukan data yang lebih kecil dari data sebelumnya , maka data tersebut di sisipakan (insert) pada posisi yang sesuai.

```
Algoritma Insertion Sort (A : array [1..N] of integer)
{Diberikan N data kemudian diurutkan naik } Deklarasi
i, j, x : integer
Deskripsi
for i 2toNdo
        x : A[i]
        j : i-1
        while j > 0 and x < A[j] do
                A[j+1] =A[j]
                j =j-1
        end
        A[j+1]= x
end for
```

Percobaan :

```
def insertion_sort(sort_list):
   for i in range(1, len(sort_list)):
      key = sort_list[i]
      j = i - 1
      while j >= 0 and key < sort_list[j]:
         sort_list[j + 1] = sort_list[j]
         j -= 1
      sort_list[j + 1] = key
   print('iterasi', i-1 ,sort_list)


A=[4,3,5,6,2,78,98]
insertion_sort(A)
```

Tugas :

1. Lakukan modifikasi fungsi pada percobaan untuk melakukan pengurutan data secara Descending.
2. Modifikasi fungsi tersebut sehingga menampilkan hasil tiap langkah pengurutan seperti contoh berikut :

```
A=[4,3,5,6,2,78,98]
insertion_sort(A)
```

```
iterasi 0 [3, 4, 5, 6, 2, 78, 98]
iterasi 1 [3, 4, 5, 6, 2, 78, 98]
iterasi 2 [3, 4, 5, 6, 2, 78, 98]
iterasi 3 [2, 3, 4, 5, 6, 78, 98]
iterasi 4 [2, 3, 4, 5, 6, 78, 98]
iterasi 5 [2, 3, 4, 5, 6, 78, 98]
```

Atau lebih lengkapnya (lebih baik)

```
A=[4,3,5,6,2,78,98]
insertion_sort(A)
```

```
pergeseran pada iterasi ke  0 j ke : 0 [4, 4, 5, 6, 2, 78, 98]

iterasi 0 [3, 4, 5, 6, 2, 78, 98]
iterasi 1 [3, 4, 5, 6, 2, 78, 98]
iterasi 2 [3, 4, 5, 6, 2, 78, 98]
pergeseran pada iterasi ke  3 j ke : 3 [3, 4, 5, 6, 6, 78, 98]

pergeseran pada iterasi ke  3 j ke : 2 [3, 4, 5, 5, 6, 78, 98]

pergeseran pada iterasi ke  3 j ke : 1 [3, 4, 4, 5, 6, 78, 98]

pergeseran pada iterasi ke  3 j ke : 0 [3, 3, 4, 5, 6, 78, 98]

iterasi 3 [2, 3, 4, 5, 6, 78, 98]
iterasi 4 [2, 3, 4, 5, 6, 78, 98]
iterasi 5 [2, 3, 4, 5, 6, 78, 98]
```