LAPORAN PRAKTIKUM ALGORITMA DAN PEMROGRAMAN

MODUL 05

"PENGURUTAN"

Nama : Aiza Fravy Qanza

NIM : L200170144

Kelas : D

Soal-soal Mahasiswa

1. Buatlah suatu program untuk mengurutkan array mahasiswa berdasarkan nim, yang elemennya terbuat dari class MTIF, yang telah kamu buat sebelumnya

```
class mhsTIF():
    def __init__(self, nama, nim, asal, saku):
            self.nama = nama
            self.nim = nim
            self.asal = asal
            self.saku = saku


m1 = mhsTIF('Aiza','L200170144','Samarinda',240000)
m2 = mhsTIF('Bella','L200170135','Jakarta',290000)
m3 = mhsTIF('Chiara','L200170128','Balikpapan',250000)
m4 = mhsTIF('Devia','L200170100','Bandung',230000)
m5 = mhsTIF('Elvira','L200170156','Surabaya',235000)
m6 = mhsTIF('Farah','L200170119','Bandung',220000)
m7 = mhsTIF('Gege','L200170148','Tarakan',260000)

mhs = [m1, m2, m3, m4, m5, m6, m7]

def urutkan(A):
    baru = {}
    for i in range(len(A)):
            baru[A[i].nama] = A[i].nim
    listofTuples = sorted(baru.items(), key=lambda x: x[1])
    for elemen in listofTuples :
            print(elemen[0] , ":" , elemen[1])
```

```
>>> urutkan(mhs)
Devia : L200170100
Farah : L200170119
Chiara : L200170128
Bella : L200170135
Aiza : L200170144
Gege : L200170148
Elvira : L200170156
```

2. Misal terdapat dua buah array yang sudah urut A dan B. Buatlah suatu program untuk menggabungkan, secara efisien, kedua array itu menjadi suatu array C yang urut.

```
def bubbleSort(arr):
    n = len(arr)
    for i in range(n):
            for j in range(0, n-i-1):
                    if arr[j] > arr[j+1] :
                            arr[j], arr[j+1] = arr[j+1], arr[j]
    return arr

def gabung(a,b):
    c = []
    c = a+b
    n = len(c)
    for i in range(n):
            for j in range(0, n-i-1):
                    if c[j] > c[j+1] :
                            c[j], c[j+1] = c[j+1], c[j]
    return c
```

```
>>> a = [28, 3, 19, 99, 31, 5, 98]
>>> b = [9, 12, 20, 2, 74, 73]
>>> a, b = bubbleSort(a), bubbleSort(b)
>>> gabung(a, b)
[2, 3, 5, 9, 12, 19, 20, 28, 31, 73, 74, 98, 99]
```

3. Manakah yang lebih cepat antara selection sort dan insertion sort? Untuk memulai menyelidikinya, kamu bisa membandingkan waktu yang diperlukan untuk mengurutkan sebuah array yang besar, misal sepanjang 6000 elemen

```
from time import time as detak
from random import shuffle as kocok
k = [i for i in range(1,6001)]
kocok(k)
def u_bub(arr):
    n = len (arr)
    for i in range(n):
            for j in range(0, n-i-1):
                    if arr[j] > arr[j+1]:
                            arr[j], arr[j+1] = arr[j+1], arr[j]


def u_sel(A):
    for i in range(len(A)):
            min_in = i
            for j in range(i+1, len(A)):
                    if A[min_in] > A[j]:
                            min_in = j
            A[i], A[min_in] = A[min_in], A[i]


def u_ins(arr):
    for i in range(1, len(arr)):
            key = arr[i]
            j = i-1
            while j >= 0 and key < arr[j]:
                    arr[j+1] = arr[j]
                    j -= 1
            arr[j+1] = key
```

```
>>> bub = k[:]
>>> sel = k[:]
>>> ins = k[:]
>>>
>>> aw=detak();u_bub(bub);ak=detak();print('bubble : %g detik' %(ak-aw));
bubble : 5.80593 detik
>>> aw=detak();u_sel(sel);ak=detak();print('bubble : %g detik' %(ak-aw));
bubble : 2.91651 detik
>>> aw=detak();u_ins(ins);ak=detak();print('bubble : %g detik' %(ak-aw));
bubble : 2.48562 detik
```