

LAPORAN PRAKTIKUM MODUL 5

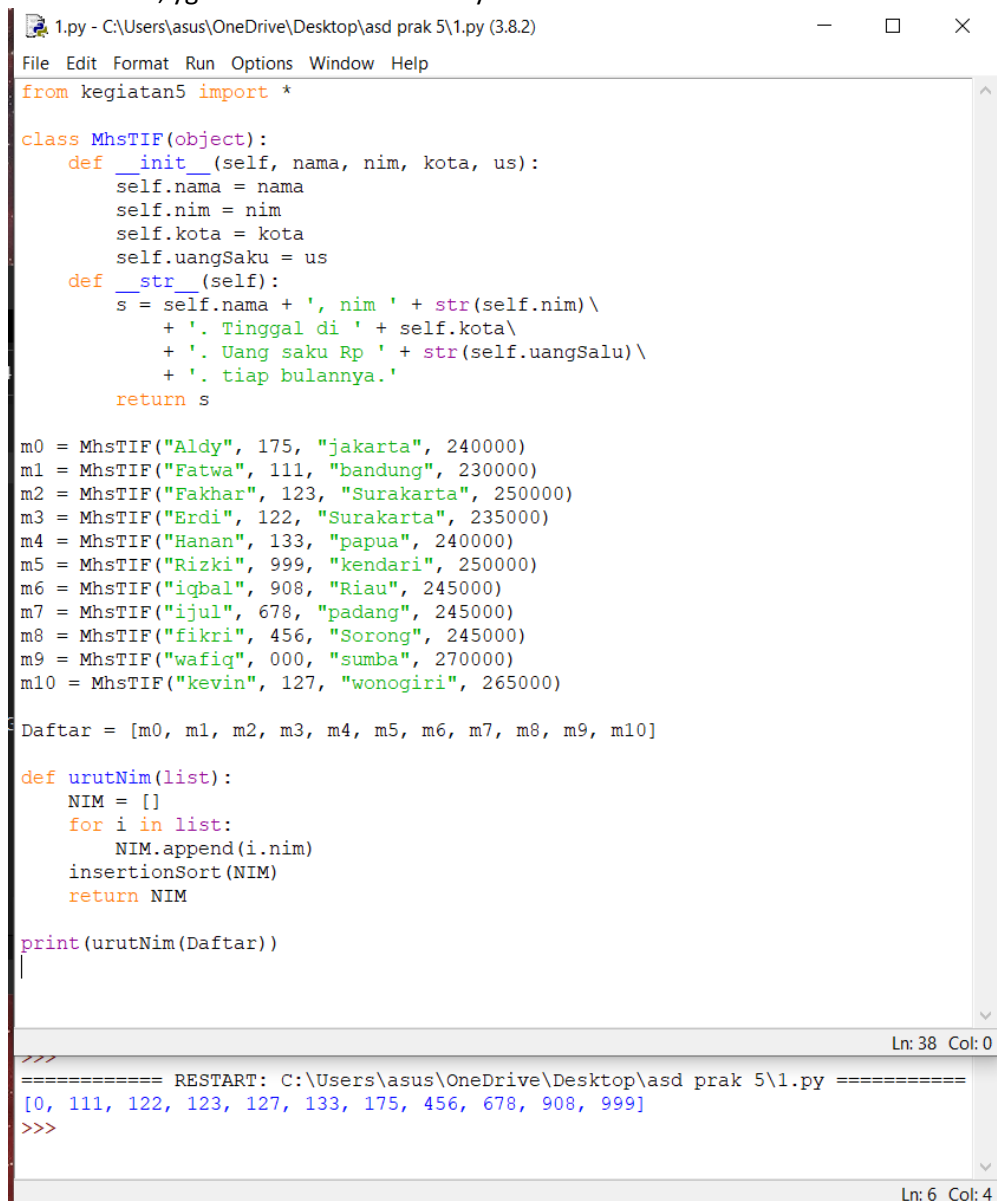
ALGORITMA DAN STRUKTUR DATA

Nama : Afrialdy Asyura Buana

Nim : L200180175

Kelas : G

1. Membuat program untuk mengurutkan array mahasiswa berdasarkan NIM, yang elemennya terbuat dari class MhsTIF, yg telah dibuat sebelumnya.



```
1.py - C:\Users\asus\OneDrive\Desktop\asd prak 5\1.py (3.8.2)
File Edit Format Run Options Window Help

from kegiatan5 import *

class MhsTIF(object):
    def __init__(self, nama, nim, kota, us):
        self.nama = nama
        self.nim = nim
        self.kota = kota
        self.uangSaku = us
    def __str__(self):
        s = self.nama + ', nim ' + str(self.nim)\
            + '. Tinggal di ' + self.kota\
            + '. Uang saku Rp ' + str(self.uangSalu)\
            + '. tiap bulannya.'
        return s

m0 = MhsTIF("Aldy", 175, "jakarta", 240000)
m1 = MhsTIF("Fatwa", 111, "bandung", 230000)
m2 = MhsTIF("Fakhar", 123, "Surakarta", 250000)
m3 = MhsTIF("Erdi", 122, "Surakarta", 235000)
m4 = MhsTIF("Hanan", 133, "papua", 240000)
m5 = MhsTIF("Rizki", 999, "kendari", 250000)
m6 = MhsTIF("iqbal", 908, "Riau", 245000)
m7 = MhsTIF("ijul", 678, "padang", 245000)
m8 = MhsTIF("fikri", 456, "Sorong", 245000)
m9 = MhsTIF("wafiq", 000, "sumba", 270000)
m10 = MhsTIF("kevin", 127, "wonogiri", 265000)

Daftar = [m0, m1, m2, m3, m4, m5, m6, m7, m8, m9, m10]

def urutNim(list):
    NIM = []
    for i in list:
        NIM.append(i.nim)
    insertionSort(NIM)
    return NIM

print(urutNim(Daftar))

===== RESTART: C:\Users\asus\OneDrive\Desktop\asd prak 5\1.py =====
[0, 111, 122, 123, 127, 133, 175, 456, 678, 908, 999]
>>>
```

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

 2.py - C:\Users\asus\OneDrive\Desktop\asd prak 5\2.py (3.8.2)

File Edit Format Run Options Window Help

```
from kegiatan5 import *
```

```
A = [1,2,3]
```

```
B = [4,5,6]
```


```
def gabungArr(list1, list2):
```

```
    C = list1 + list2
```

```
    insertionSort(C)
```

```
    return C
```

```
print(gabungArr(A,B))
```

 Python 3.8.2 Shell

File Edit Shell Debug Options Window Help

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020) on win32

Type "help", "copyright", "credits" or "license()" for more

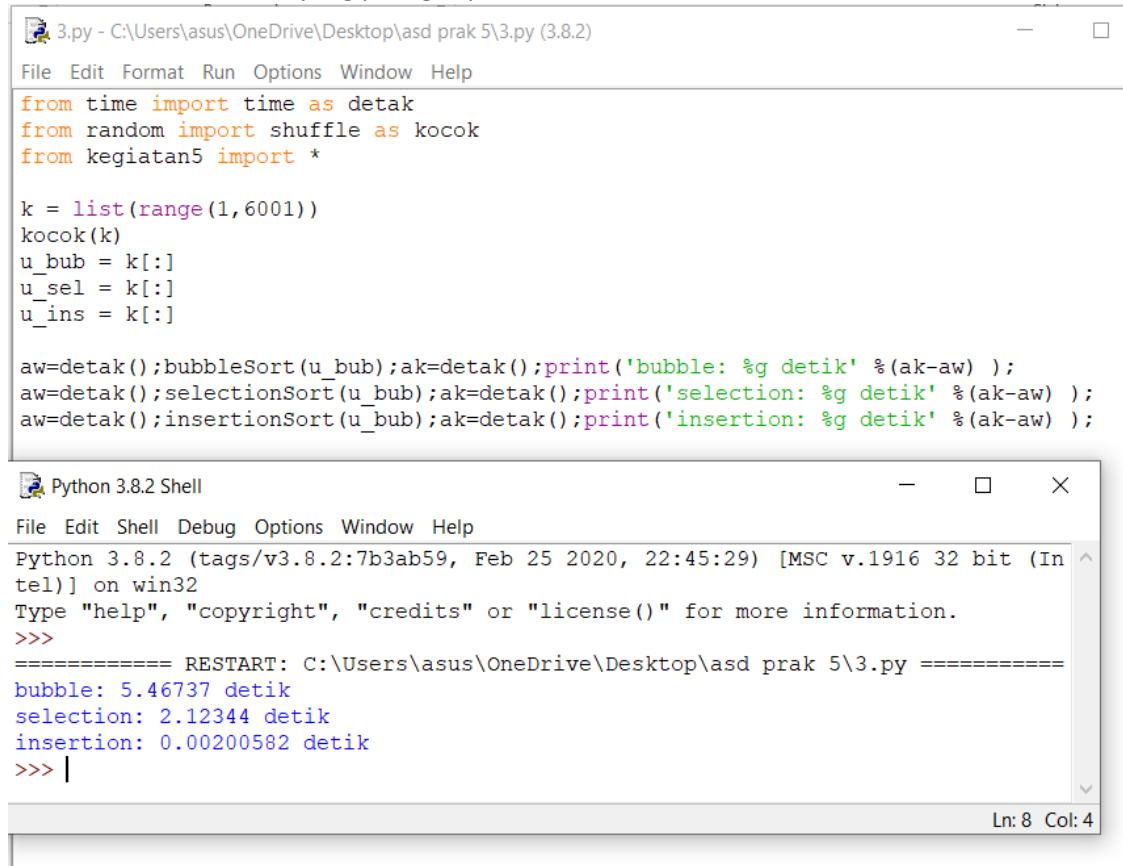
```
>>>
```

```
===== RESTART: C:\Users\asus\OneDrive\Documents\Python3\Python382\Python382Shell>
```

```
[1, 2, 3, 4, 5, 6]
```

```
>>> |
```

3. Membuktikan manakah yang paling cepat bubble sort, selection sort, atau insertion sort.



The image shows a Python script in a text editor and its execution in a Python 3.8.2 Shell. The script generates a list of 6001 random numbers and measures the execution time of three sorting algorithms: bubble sort, selection sort, and insertion sort. The results show that insertion sort is the fastest, followed by selection sort, and bubble sort is the slowest.

```
3.py - C:\Users\asus\OneDrive\Desktop\asd prak 5\3.py (3.8.2)
File Edit Format Run Options Window Help

from time import time as detik
from random import shuffle as kocok
from kegiatan5 import *

k = list(range(1,6001))
kocok(k)
u_bub = k[:]
u_sel = k[:]
u_ins = k[:]

aw=detak();bubbleSort(u_bub);ak=detak();print('bubble: %g detik' %(ak-aw) );
aw=detak();selectionSort(u_bub);ak=detak();print('selection: %g detik' %(ak-aw) );
aw=detak();insertionSort(u_bub);ak=detak();print('insertion: %g detik' %(ak-aw) );

Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\asus\OneDrive\Desktop\asd prak 5\3.py =====
bubble: 5.46737 detik
selection: 2.12344 detik
insertion: 0.00200582 detik
>>> |
```

Ln: 8 Col: 4

Jadi, metode tercepat yaitu insertion sort, kemudian selection sort, dan sudah jelas bubble sort paling lambat