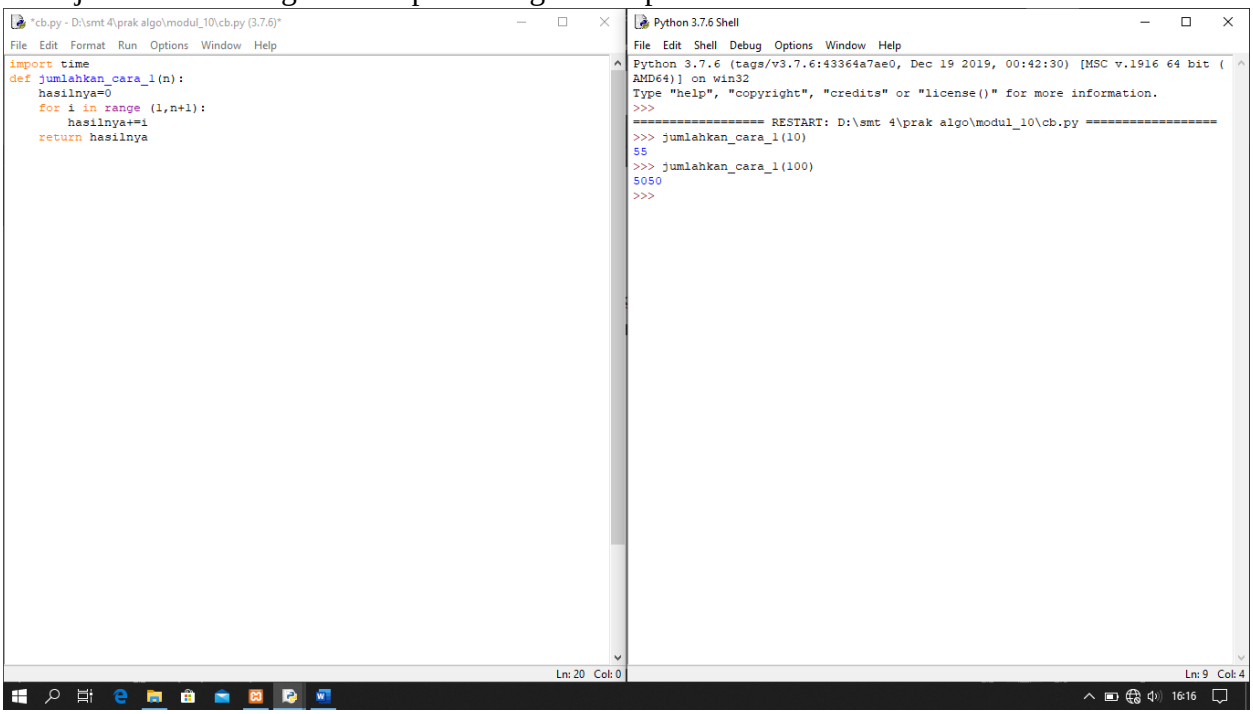
Nama : Rohmad Khoirudin
NIM : L200180101
Kelas : D

Prak-ASD

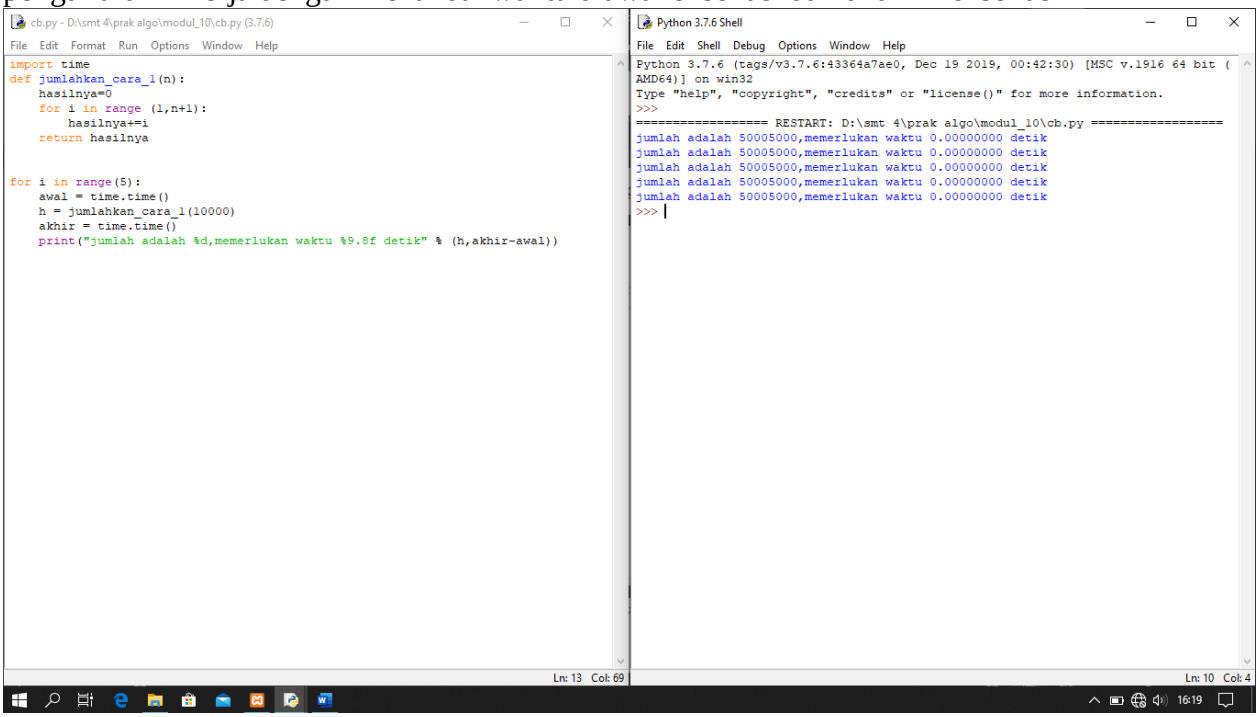
Modul 10

Latihan

1) mennjumlahkan bilangan 1 sampai n dengan cara pertama



2) pengukuran kinerja dengan menandai waktu diawal eksekusi dan di akhir eksekusi



cb.py - D:\smt 4\prak algo\modul_10\cb.py (3.7.6)

File Edit Format Run Options Window Help

```
import time
def jumlahkan_cara_1(n):
    hasilnya=0
    for i in range (1,n+1):
        hasilnya+=i
    return hasilnya


for i in range(5):
    awal = time.time()
    h = jumlahkan_cara_1(1000000)
    akhir = time.time()
    print("jumlah adalah %d,memerlukan waktu %9.8f detik" % (h,akhir-awal))
```

Ln: 13 Col: 75

Python 3.7.6 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:\smt 4\prak algo\modul_10\cb.py ==================
jumlah adalah 500000500000,memerlukan waktu 0.12490177 detik
jumlah adalah 500000500000,memerlukan waktu 0.12498975 detik
jumlah adalah 500000500000,memerlukan waktu 0.10936832 detik
jumlah adalah 500000500000,memerlukan waktu 0.10929513 detik
jumlah adalah 500000500000,memerlukan waktu 0.10931134 detik
>>>
```

Ln: 10 Col: 4

16:20

3) mennjumlahkan bilangan 1 sampai n dengan cara kedua

cb.py - D:\smt 4\prak algo\modul_10\cb.py (3.7.6)

File Edit Format Run Options Window Help

```
import time
##jml cara 1
def jumlahkan_cara_1(n):
    hasilnya=0
    for i in range (1,n+1):
        hasilnya+=i
    return hasilnya
##jml cara 2
def jumlahkan_cara_2(n):
    return (n*(n+1))/2

print("cara ke 2",jumlahkan_cara_2(1000000))
for i in range(5):
    awal = time.time()
    h = jumlahkan_cara_2(1000000)
    akhir = time.time()
    print("jumlah adalah %d,memerlukan waktu %9.8f detik" % (h,akhir-awal))
```

Ln: 15 Col: 24

Python 3.7.6 Shell

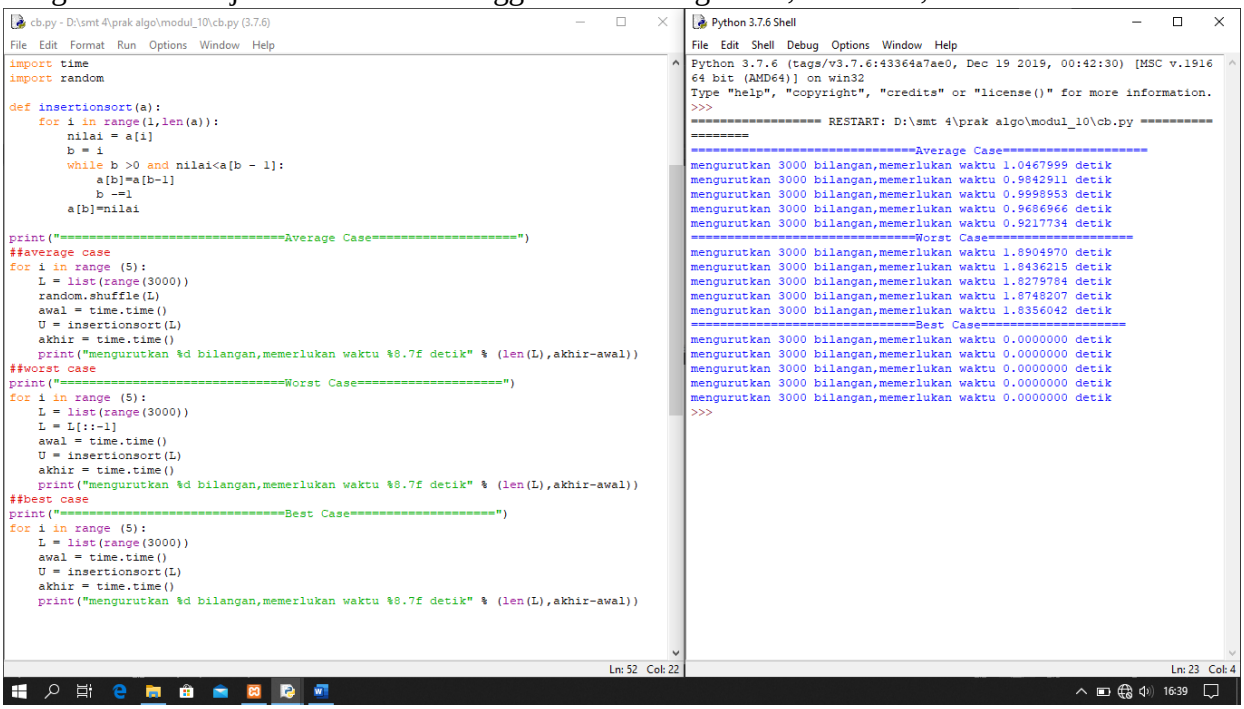
File Edit Shell Debug Options Window Help

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:\smt 4\prak algo\modul_10\cb.py ==================
cara ke 2 500000500000.0
jumlah adalah 500000500000,memerlukan waktu 0.00000000 detik
jumlah adalah 500000500000,memerlukan waktu 0.00000000 detik
jumlah adalah 500000500000,memerlukan waktu 0.00000000 detik
jumlah adalah 500000500000,memerlukan waktu 0.00000000 detik
jumlah adalah 500000500000,memerlukan waktu 0.00000000 detik
>>>
```

Ln: 11 Col: 4

16:24

4) Pengukuran kinerja insertion sort menggunakan average case,wors case, best case



```
import time
import random

def insertionsort(a):
    for i in range(1,len(a)):
        nilai = a[i]
        b = i
        while b >0 and nilai<a[b - 1]:
            a[b]=a[b-1]
            b -=1
        a[b]=nilai

print("==============================Average Case====================")
##average case
for i in range (5):
    L = list(range(3000))
    random.shuffle(L)
    awal = time.time()
    U = insertionsort(L)
    akhir = time.time()
    print("mengurutkan %d bilangan,memerlukan waktu %8.7f detik" % (len(L),akhir-awal))
##worst case
print("==============================Worst Case====================")
for i in range (5):
    L = list(range(3000))
    L = L[::-1]
    awal = time.time()
    U = insertionsort(L)
    akhir = time.time()
    print("mengurutkan %d bilangan,memerlukan waktu %8.7f detik" % (len(L),akhir-awal))
##best case
print("==============================Best Case====================")
for i in range (5):
    L = list(range(3000))
    awal = time.time()
    U = insertionsort(L)
    akhir = time.time()
    print("mengurutkan %d bilangan,memerlukan waktu %8.7f detik" % (len(L),akhir-awal))
```

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916
64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:\smt 4\prak algo\modul_10\cb.py ==========
========
==============================Average Case====================
mengurutkan 3000 bilangan,memerlukan waktu 1.0467999 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.9842911 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.9998953 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.9686966 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.9217734 detik
==============================Worst Case====================
mengurutkan 3000 bilangan,memerlukan waktu 1.8904970 detik
mengurutkan 3000 bilangan,memerlukan waktu 1.8436215 detik
mengurutkan 3000 bilangan,memerlukan waktu 1.8279784 detik
mengurutkan 3000 bilangan,memerlukan waktu 1.8748207 detik
mengurutkan 3000 bilangan,memerlukan waktu 1.8356042 detik
==============================Best Case====================
mengurutkan 3000 bilangan,memerlukan waktu 0.0000000 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.0000000 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.0000000 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.0000000 detik
mengurutkan 3000 bilangan,memerlukan waktu 0.0000000 detik
>>>
```

5) Analisis kode python

```
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: D:\smt 4\prak algo\modul_10\cb.py ==========
>>> x = 5
>>> y = x
>>> z = x + y*8
>>> d = x>0 and x<100
>>> f = [3,2,4,5]
>>> v = f[0:2]
>>> print("x = ",x)
x =  5
>>> print("y = ",y)
y =  5
>>> print("z = ",z)
z =  45
>>> print("d = ",d)
d =  True
>>> print("f = ",f)
f =  [3, 2, 4, 5]
>>> print("v = ",v)
v =  [3, 2]
>>> count  = 0
>>> i = 32
>>> while i >=1:
        count +=1
        i = i//2
        print("i = ",i," ","count = ",count)


i =  16   count =  1
i =  8   count =  2
i =  4   count =  3
i =  2   count =  4
i =  1   count =  5
i =  0   count =  6
>>> print(count)
6
>>>
```

6) Analisis pewaktuan menggunakan timeit

```
Python 3.7.6 Shell
File  Edit  Shell  Debug  Options  Window  Help
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 19 2019, 00:42:30) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:\smt 4\prak algo\modul_10\cb.py ==================
>>> from timeit import timeit
>>> timeit('sqrt(2)','from math import sqrt',number = 10000)
0.001693600000001183
>>> timeit('sqrt(2)','from math import sqrt',number = 100000)
0.03557589999999777
>>> timeit('sqrt(2)','from math import sqrt',number = 1000000)
0.21716159999999718
>>> timeit("1+2")
0.016093800000000158
>>> timeit("sin (pi/3)", setup = "from math import sin,pi")
0.269063899999999
>>> timeit("sin (1.047)", setup = "from math import sin")
0.17494119999999214
>>>
```

7) Melihat $O(n^2)$ pada nested loop

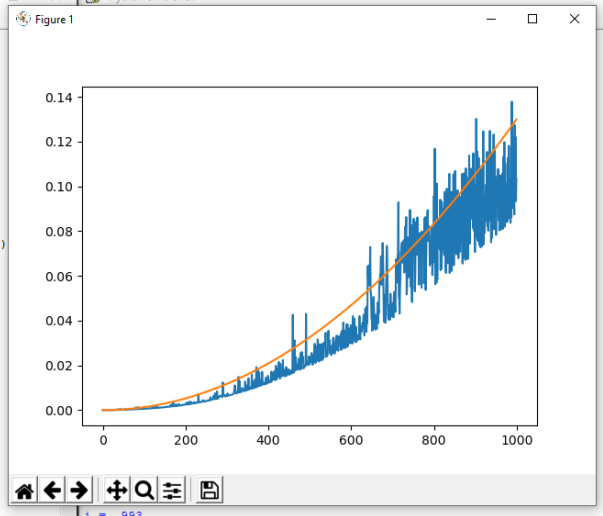
```
*cb.py - D:\smt 4\prak algo\modul_10\cb.py (3.7.6)*
File  Edit  Format  Run  Options  Window  Help
import timeit
import matplotlib.pyplot as plt

def kalangBersusuh(n):
    for i in range(n):
        for j in range (n):
            i+j

def ujiKalangBersusuh(n):
    ls=[]
    jangkauan=range(1,n+1)
    siap = "from __main__ import kalangBersusuh"
    for i in jangkauan:
        ##print('i = ',i)
        t=timeit.timeit("kalangBersusuh(" +str(i) +")",setup=siap,number=1)
        ls.append(t)
    return ls

n=1000
LS=ujiKalangBersusuh(n)

plt.plot(LS)
skala=7700000
plt.plot([x*x/skala for x in range (1,n+1)])
plt.show()
```




```
*Python 3.7.6 Shell*
i =  993
i =  994
i =  995
i =  996
i =  997
i =  998
i =  999
i =  1000
```