PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA
(Algorithm and Data Structure)

LAPORAN TUGAS
MODUL 10

Nama : Shafa Bani Saputra
NIM : L200190151
Kelas : G

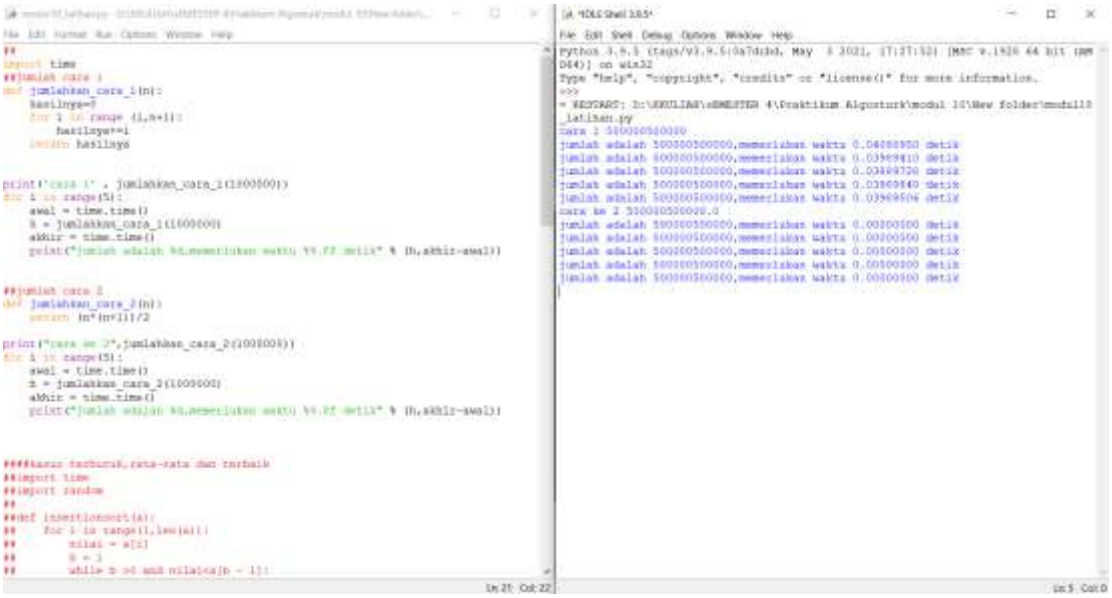
PROGRAM STUDI INFORMATIKA

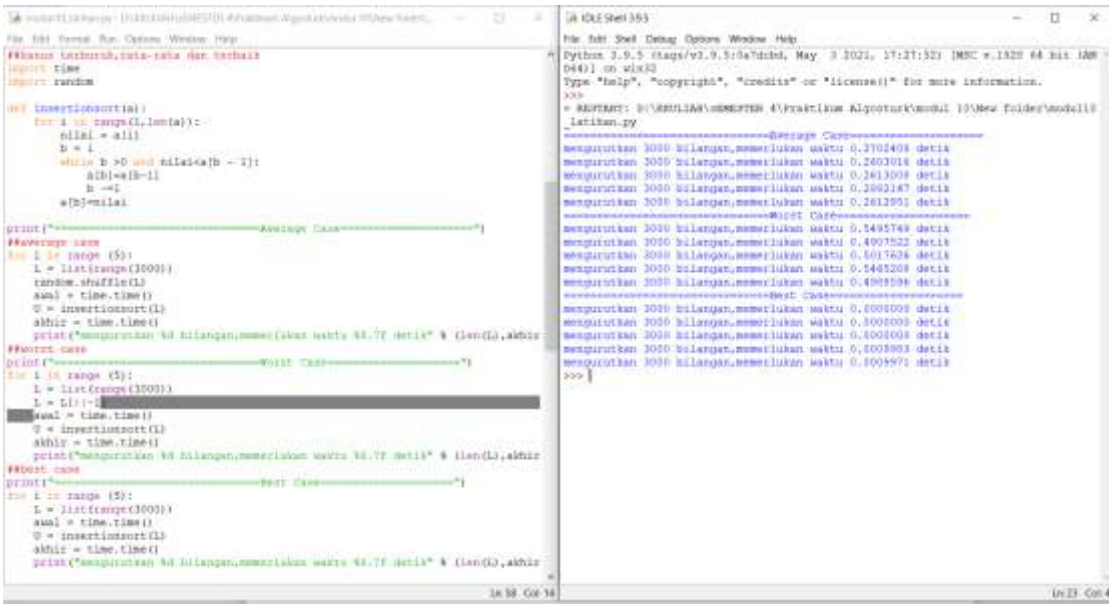
FAKULTAS KOMUNIKASI DAN INFORMATIKA

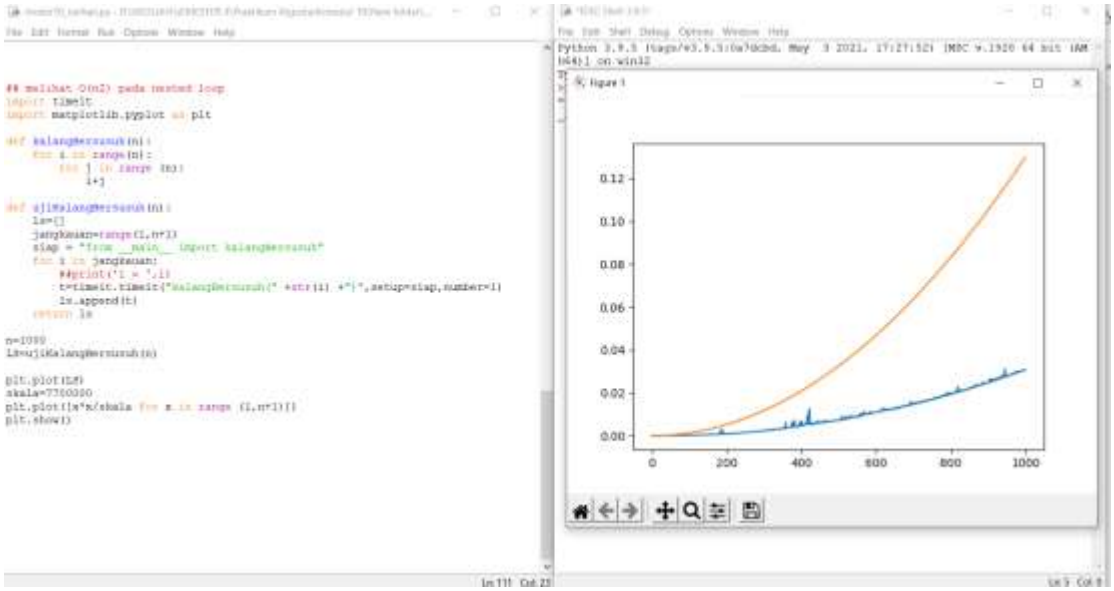
UNIVERSITAS MUHAMMADIYAH

SURAKARTA

Latihan

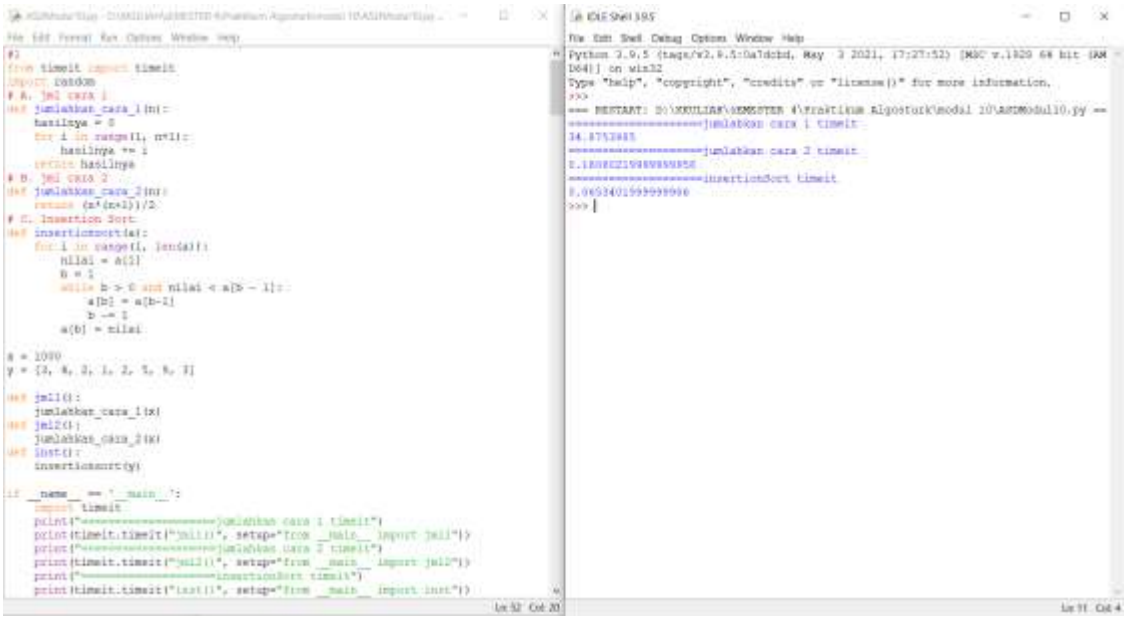




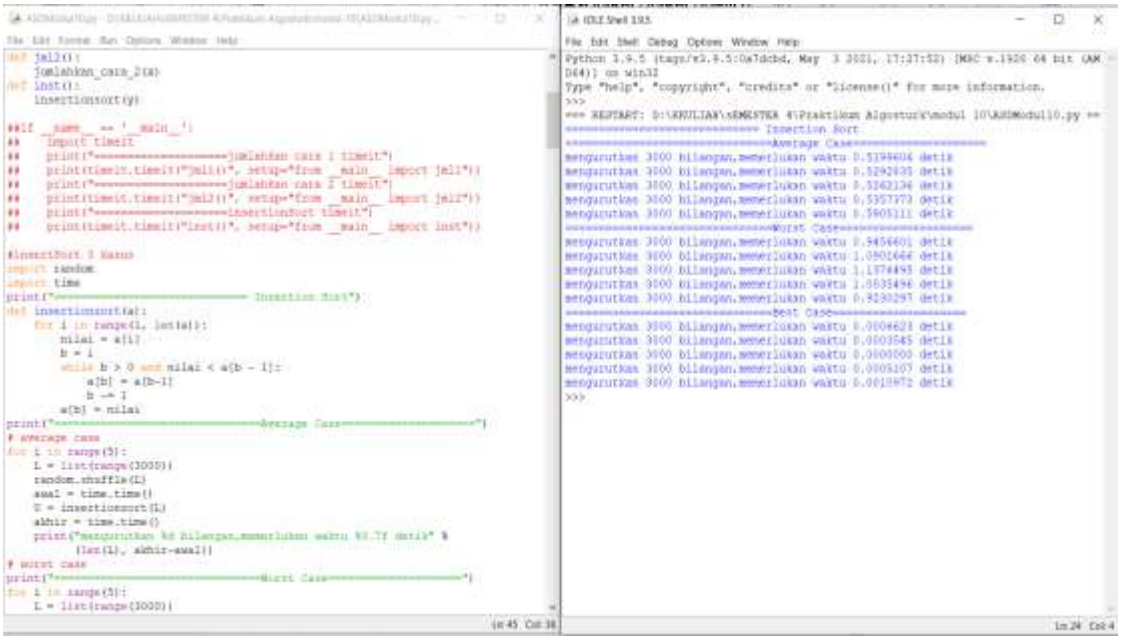


Tugas

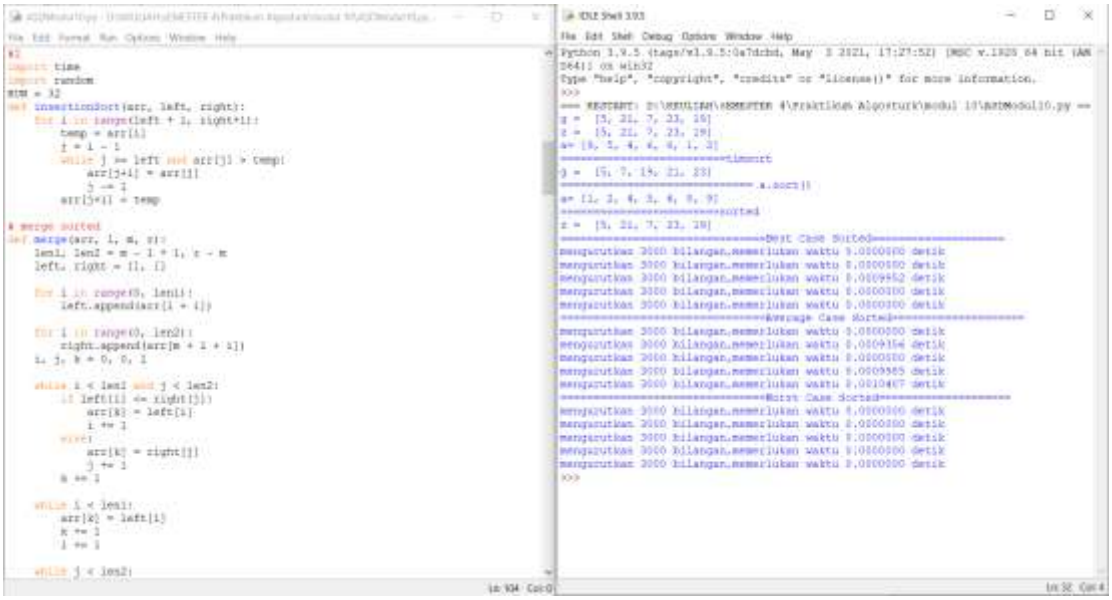
1. Time it

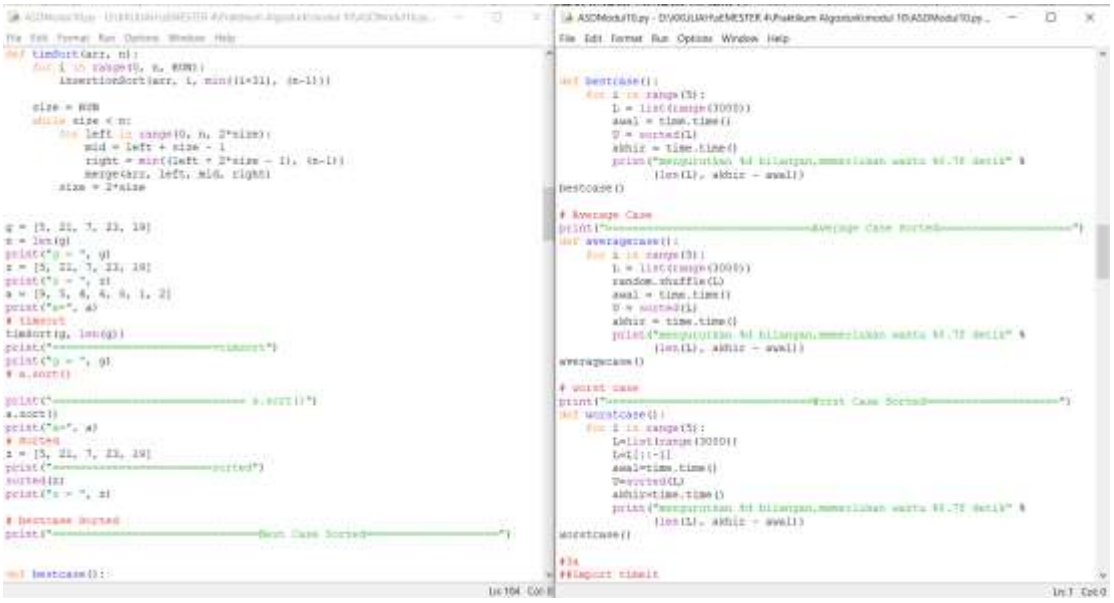


3 Kasus

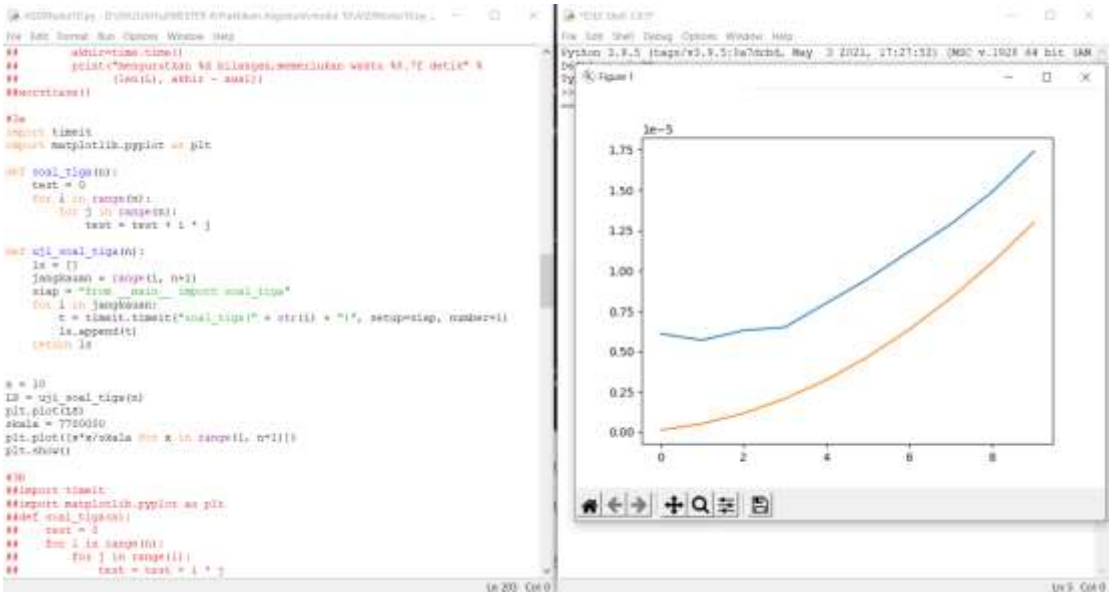


2.

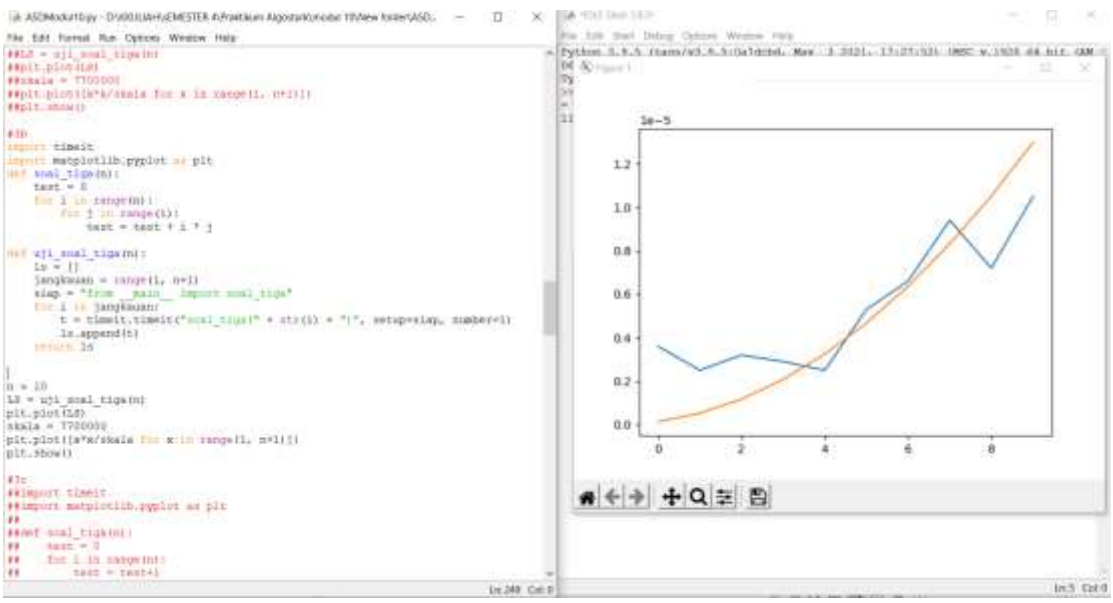




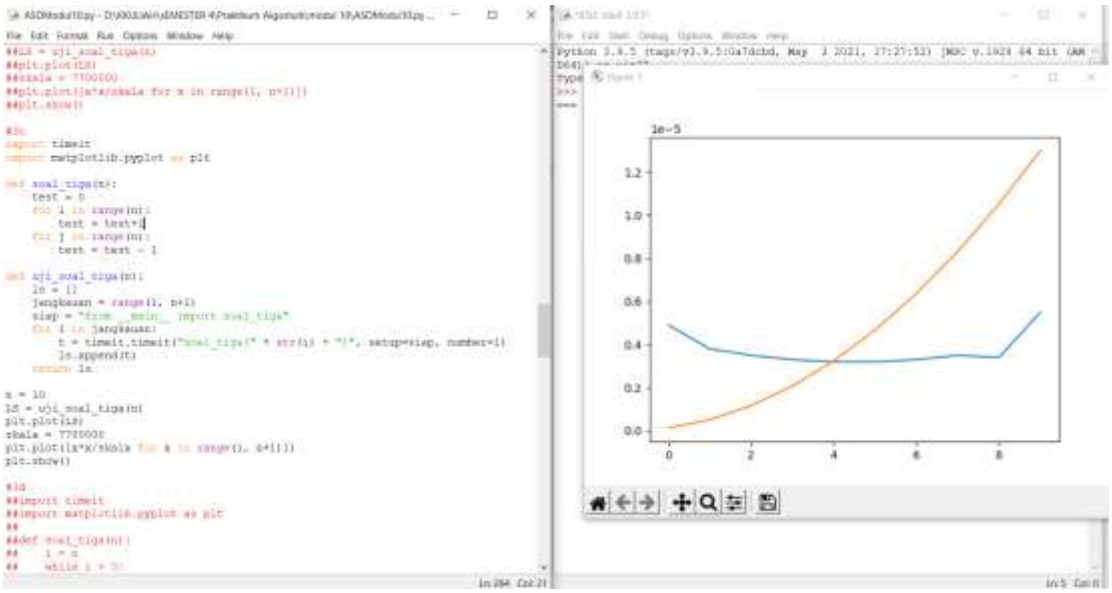
3A



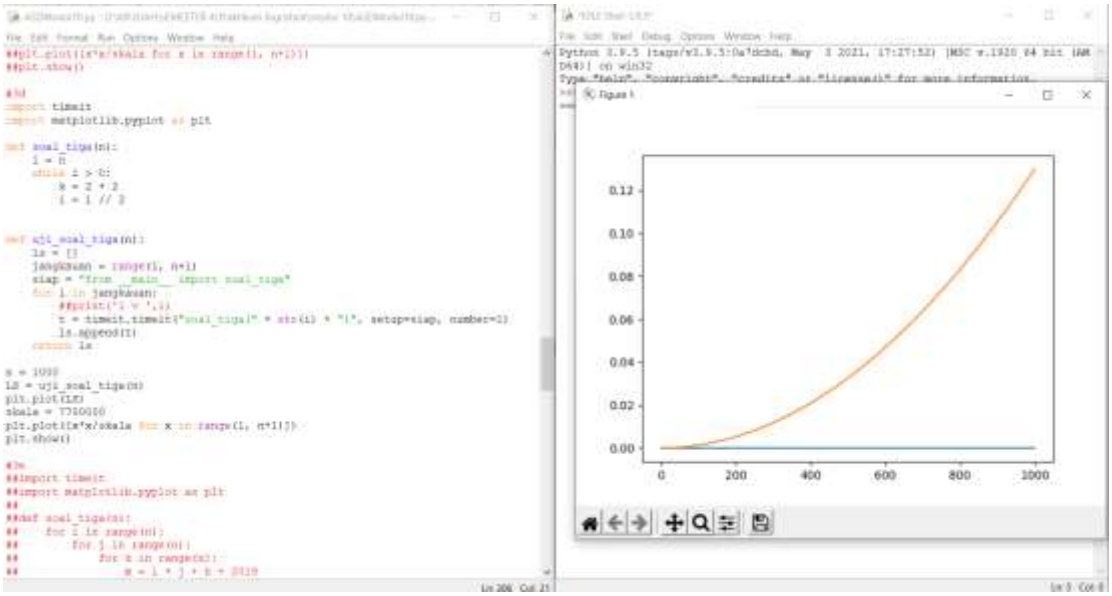
3B



3C


1e−5
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
2
4
6
8

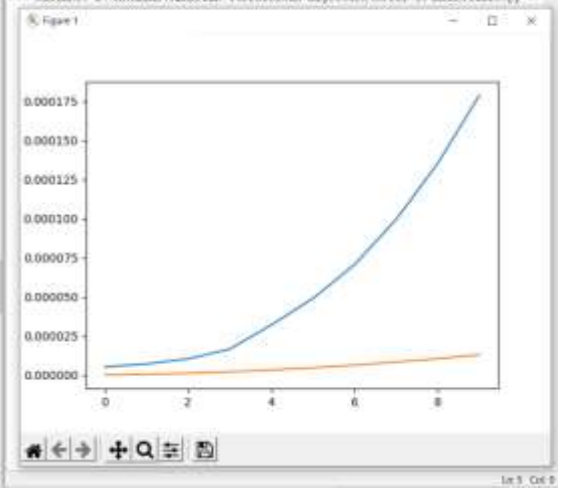
3D


0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
200
400
600
800
1000

3E

```
#3e
import timeit
import matplotlib.pyplot as plt

def soal_tiga(n):
    for i in range(n):
        for j in range(n):
            for k in range(n):
                m = i + j + k + 2019


def uji_soal_tiga(n):
    ls = []
    jangkauan = range(1, n+1)
    siap = "from __main__ import soal_tiga"
    for i in jangkauan:
        t = timeit.timeit("soal_tiga(" + str(i) + ")", setup=siap, number=1)
        ls.append(t)
    return ls

n = 10
LS = uji_soal_tiga(n)
plt.plot(LS)
skala = 7700030
plt.plot([x*x/skala for x in range(1, n+1)])
plt.show()

#3f
##import timeit
##import matplotlib.pyplot as plt
##
##def soal_tiga(n):
##    for i in range(n):
##        for j in range(i):
##            for k in range(j):
##                m = i + j + k + 2019
##
##
##def uji_soal_tiga(n):
```

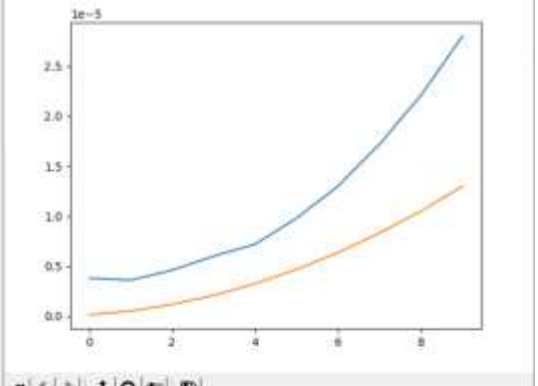
```
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May  3 2021, 17:27:52) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: D:\KEKULIAH\SEMESTER 4\Praktikum Algoritma\Modul 1\3e.py ===
>>>
```



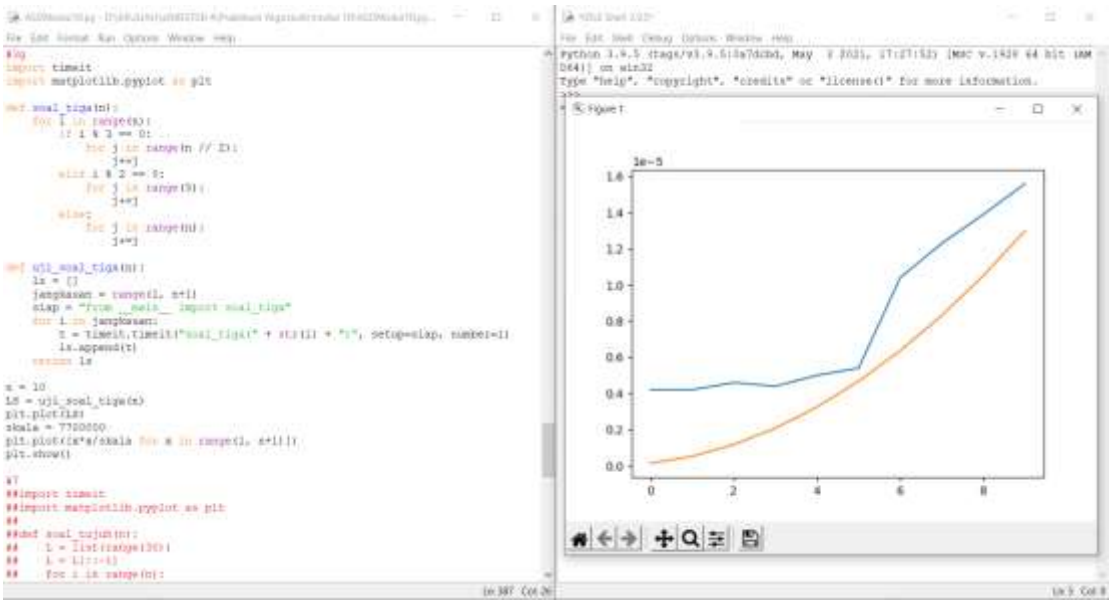
3F

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

#3f
import timeit
import matplotlib.pyplot as plt

def soal_tiga(n):
    for i in range(n):
        for j in range(i):
            for k in range(j):
                m = i + j + k + 2019


def uji_soal_tiga(n):
    ls = []
    jangkauan = range(1, n+1)
    siap = "from __main__ import soal_tiga"
    for i in jangkauan:
        t = timeit.timeit("soal_tiga(" + str(i) + ")", setup=siap, number=1)
        ls.append(t)
    return ls

n = 10
LS = uji_soal_tiga(n)
plt.plot(LS)
skala = 7700030
plt.plot([x*x/skala for x in range(1, n+1)])
plt.show()

#3g
##import timeit
##import matplotlib.pyplot as plt
##
##def soal_tiga(n):
##    for i in range(n):
##        if i % 2 == 0:
##            for j in range(n // 2):
##                j+=1
##        elif i % 2 == 0:
```

```
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May  3 2021, 17:27:52) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```



3G



4

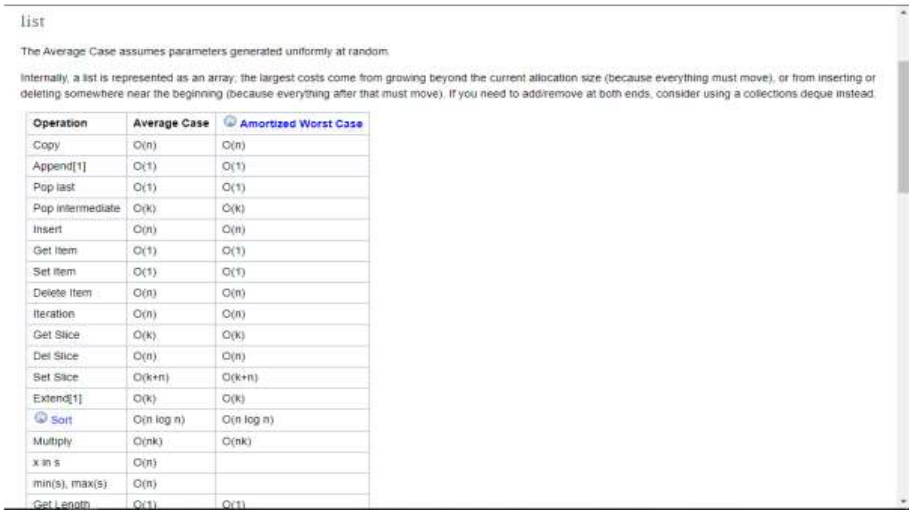
Urutkan dari yang pertumbuhan kompleksitasnya lambat ke yang cepat
n $\log_2 n$ $4n$ $10 \log_2 n$ $5n^2$ $\log_4 n$ $12n^6$ $2^{\log_2 n}$
n^3 !
$\log_4 n$ $2^{\log_2 n}$ $10 \log_2 n$ $n \log_2 n$ $5n^2$ n^3 $12n^6$ 4^n

5

Tentukan O(.) dari fungsi-fungsi
berikutini. a) $T(n) = n^2 + 32n + 8$
$O(n^2)$
b) $T(n) = 87n + 8n$
$O(n)$
c) $T(n) = 4n + 5n \log n + 102$ $O(n)$
d) $T(n) = \log n + 3n^2 + 88$ $O(n^2)$
e) $T(n) = 3(2^n) + n^2 + 647$ $O(n^2)$
f) $T(n,k) = kn + \log(k)$
$O(k^n)$
g) $T(n,k) = 8n + k \log n + 800$ $O(n)$
h) $T(n,k) = 100kn + n$
$O(n^k)$

6

Carilah di internet kompleksitas metode – metode object list di python.
- Google python list methods complexity
- Kunjungi https://wiki.python.org/moin/TimeComplexity



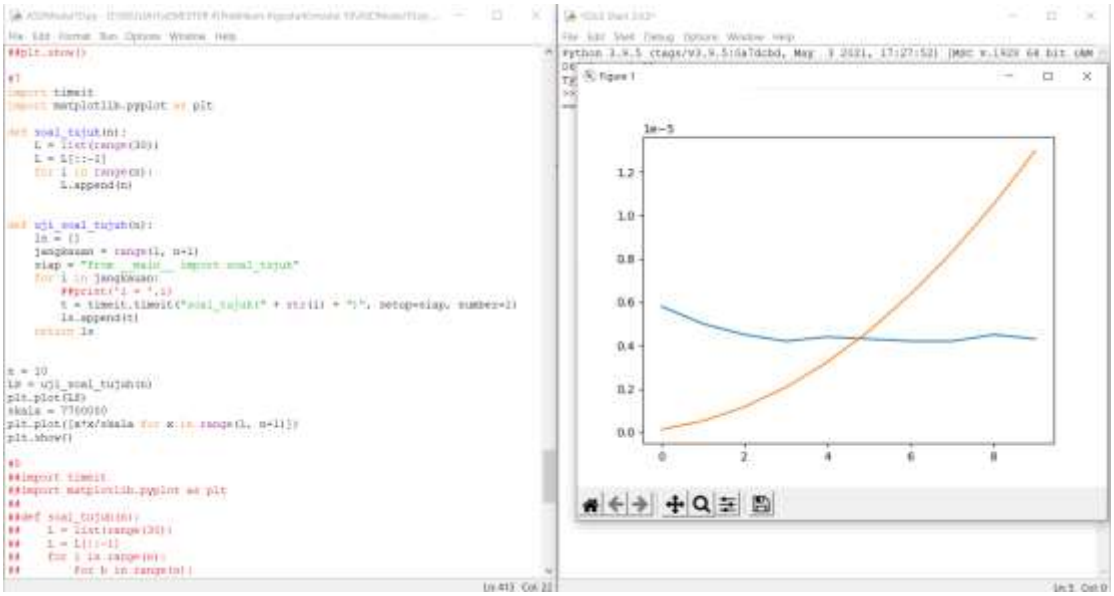
list

The Average Case assumes parameters generated uniformly at random.

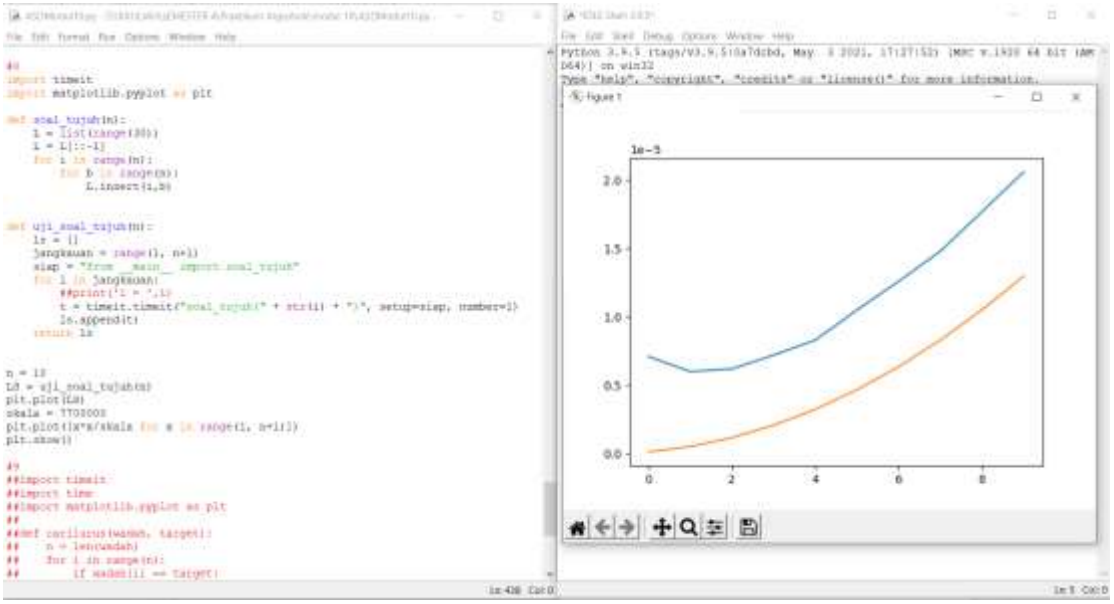
Internally, a list is represented as an array; the largest costs come from growing beyond the current allocation size (because everything must move), or from inserting or deleting somewhere near the beginning (because everything after that must move). If you need to add/remove at both ends, consider using a collections deque instead.

Operation	Average Case	Amortized Worst Case
Copy	O(n)	O(n)
Append[1]	O(1)	O(1)
Pop last	O(1)	O(1)
Pop intermediate	O(k)	O(k)
Insert	O(n)	O(n)
Get Item	O(1)	O(1)
Set Item	O(1)	O(1)
Delete Item	O(n)	O(n)
Iteration	O(n)	O(n)
Get Slice	O(k)	O(k)
Del Slice	O(n)	O(n)
Set Slice	O(k+n)	O(k+n)
Extend[1]	O(k)	O(k)
Sort	O(n log n)	O(n log n)
Multiply	O(nk)	O(nk)
x in s	O(n)	
min(s), max(s)	O(n)	
Get Length	O(1)	O(1)

7



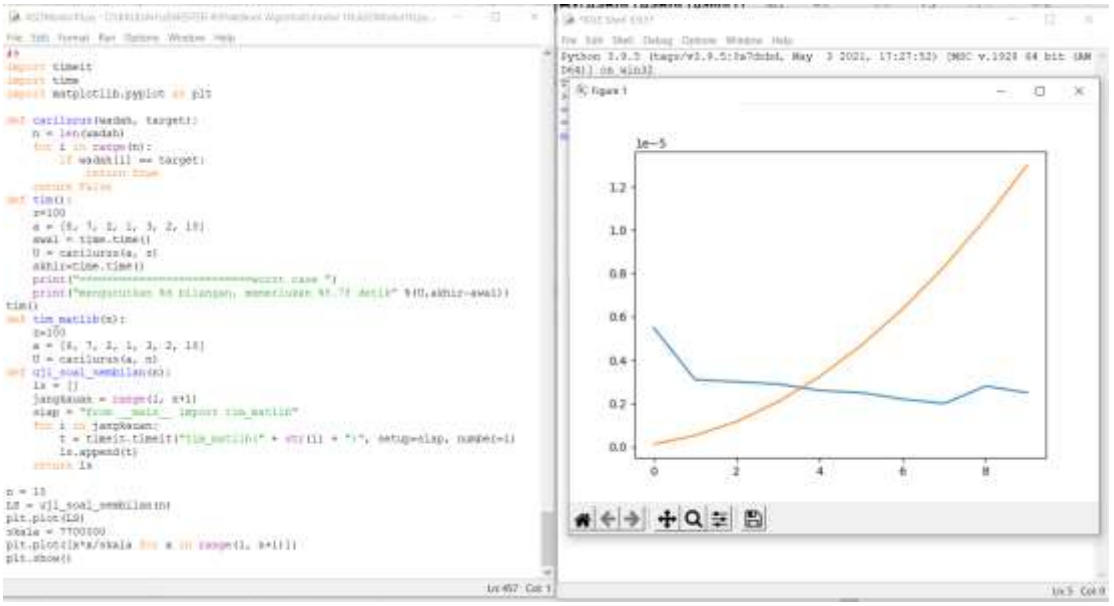
8



```
#8
import timeit
import matplotlib.pyplot as plt

def soal_tujuh(n):
    L = list(range(10))
    L = L[::-1]
    for i in range(n):
        for b in range(n):
            L.insert(1,b)


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


n = 10
LS = uji_soal_tujuh(n)
plt.plot(LS)
skala = 1700000
plt.plot([x*x/skala for x in range(1, n+1)])
plt.show()

#9
##import timeit
##import time
##import matplotlib.pyplot as plt
##
##def carilurus(wadah, target):
##    n = len(wadah)
##    for i in range(n):
##        if wadah[i] == target:
```

```
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May  3 2021, 17:27:52) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
```


9



```
#9
import timeit
import time
import matplotlib.pyplot as plt

def carilurus(wadah, target):
    n = len(wadah)
    for i in range(n):
        if wadah[i] == target:
            return True
    return False
def tim():
    n=100
    a = [8, 7, 2, 1, 3, 2, 18]
    awal = time.time()
    U = carilurus(a, n)
    akhir=time.time()
    print("------------------------------worst case ")
    print("mengurutkan %s dilangan, memerlukan %f.7f detik" %(U,akhir-awal))
tim()
def tim_matlib(n):
    n=100
    a = [8, 7, 2, 1, 3, 2, 18]
    U = carilurus(a, n)
def uji_soal_sembilan(n):
    ls = []
    jangkauan = range(1, n+1)
    siap = "from __main__ import tim_matlib"
    for i in jangkauan:
        t = timeit.timeit("tim_matlib(" + str(i) + ")", setup=siap, number=1)
        ls.append(t)
    return ls

n = 18
LS = uji_soal_sembilan(n)
plt.plot(LS)
skala = 1700000
plt.plot([x*x/skala for x in range(1, n+1)])
plt.show()
```

```
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May  3 2021, 17:27:52) [MSC v.1928 64 bit (AM
D64)] on win32
```