**PRAKTIKUM ALGORITMA STRUKTUR DATA**

**MODUL 10**

**ANALISIS ALGORITMA**

****

**Disusun oleh:**

**Adinda Aulia Hapsari**

**L200220037**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS KOMUNIKASI DAN INFORMATIKA**

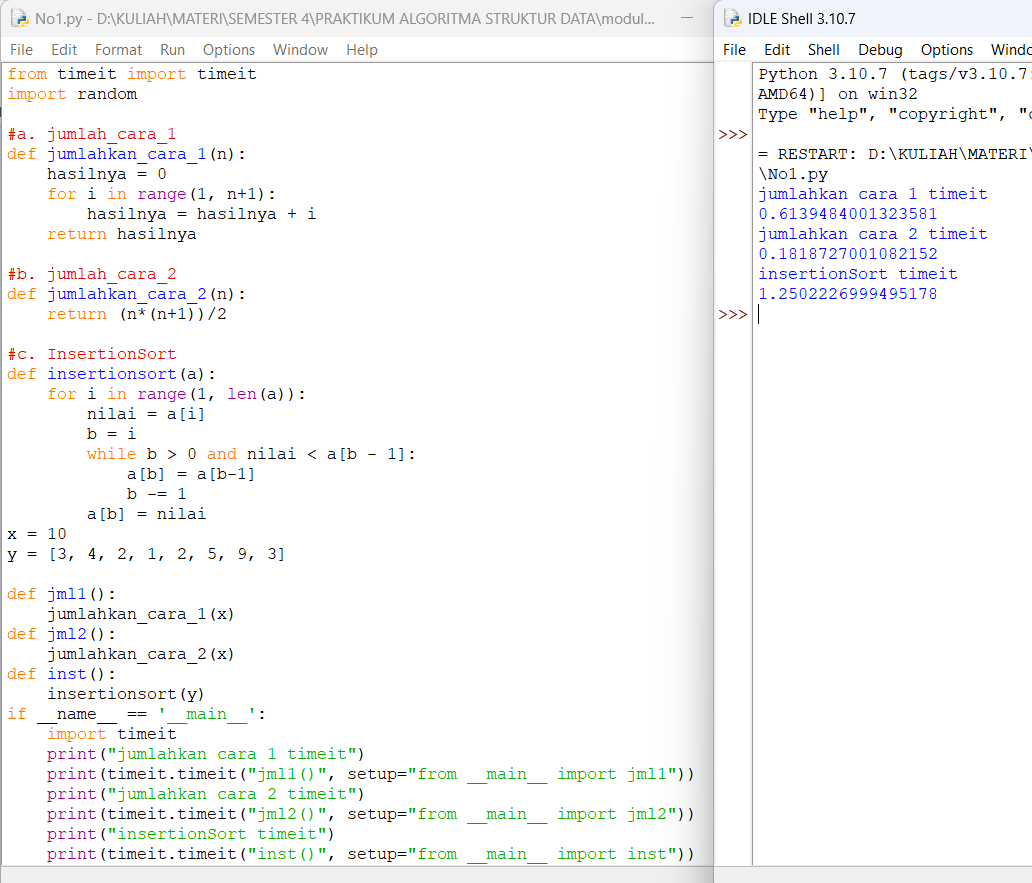
**UNIVERSITAS MUHAMMADIYAH SURAKARTA**

**TAHUN 2024**

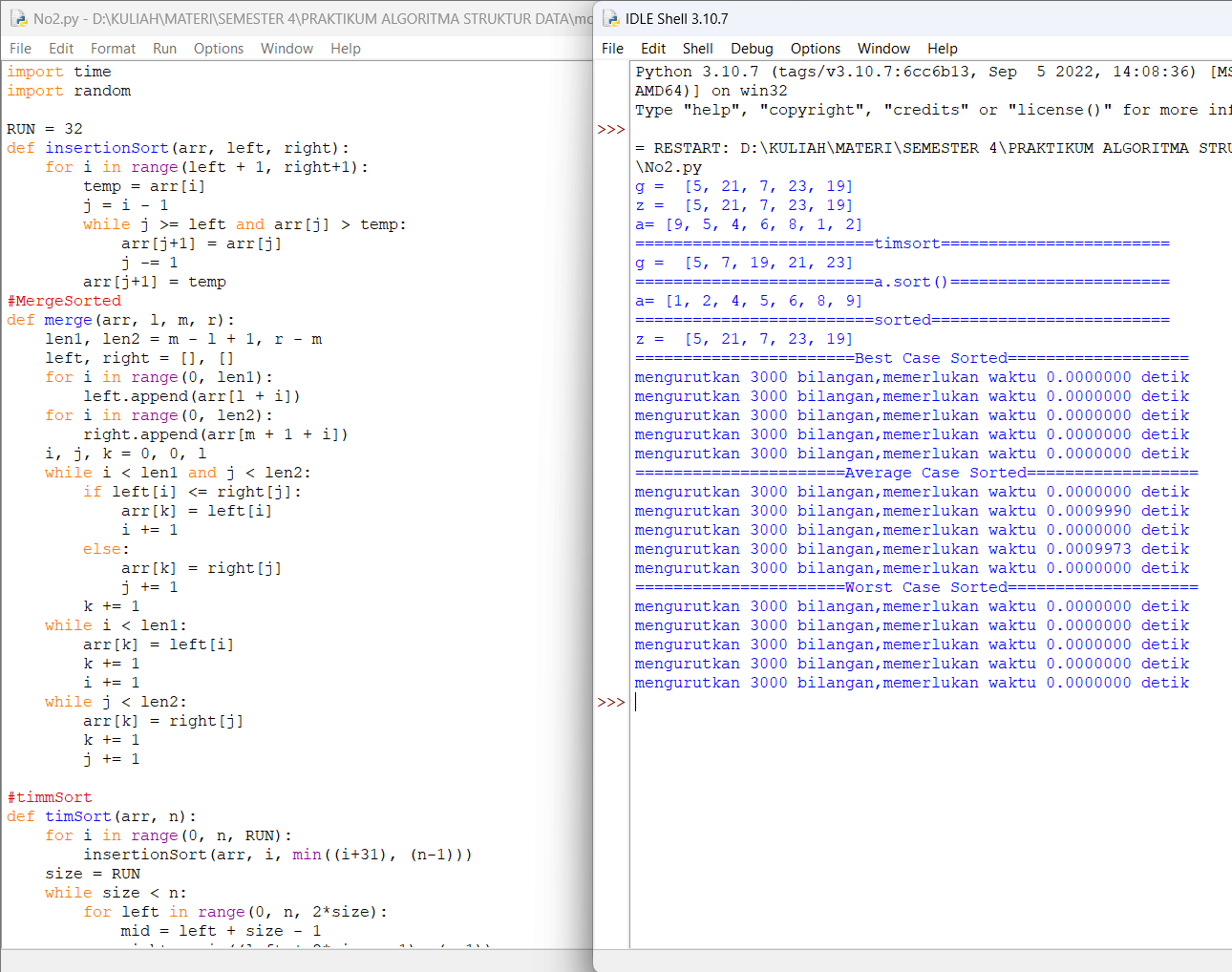
|  |  |
| --- | --- |
| Setelah kegiatan selesai, lembar kerja ini dicetak (di-print) dan dikumpulkan ke asisten.  NIM : L200220037  Nama : Adinda Aulia Hapsari  Nama Asisten :  Tanggal Praktikum : | (Diisi oleh Asisten)  Nilai Praktek :  Tanda Tangan : |

1. Kerjakan ulang contoh dan Latihan di modul ini menggunakan timeit, yaitu
2. Jumlahkan\_cara\_1
3. Jumlahkan\_cara\_2
4. insertionShort

untuk insertionShort, kerjakan untuk ketiga kasusnya.



1. Python mempunyai perintah untuk mengurutkan suatu list yang memanfaatkan algoritma Timsort. Jika g adalah suatu list berisi bilangan, maka g.sort() kan mengurutkannya. Perintah yang lain, sorted() mengurutkan list dan mengembalikan sebuah list baru yang sudah urut. Selidikilah fungsi sorted ini menggunakan timeit:
2. Apakah yang merupakan best case dan average case bagi sorted()?
3. Confirm bahwa data input urutan terbalik bukan kasus terburuk bagi sorted(). Bahkan dia lebih cepat dalam mengurutkannya dari pada data input random.



1. a. kode

## Nomer 3.a

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.a")

def soal\_tiga(n):

    test = 0

    for i in range(n):

        for j in range(n):

            test = test + i \* j

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

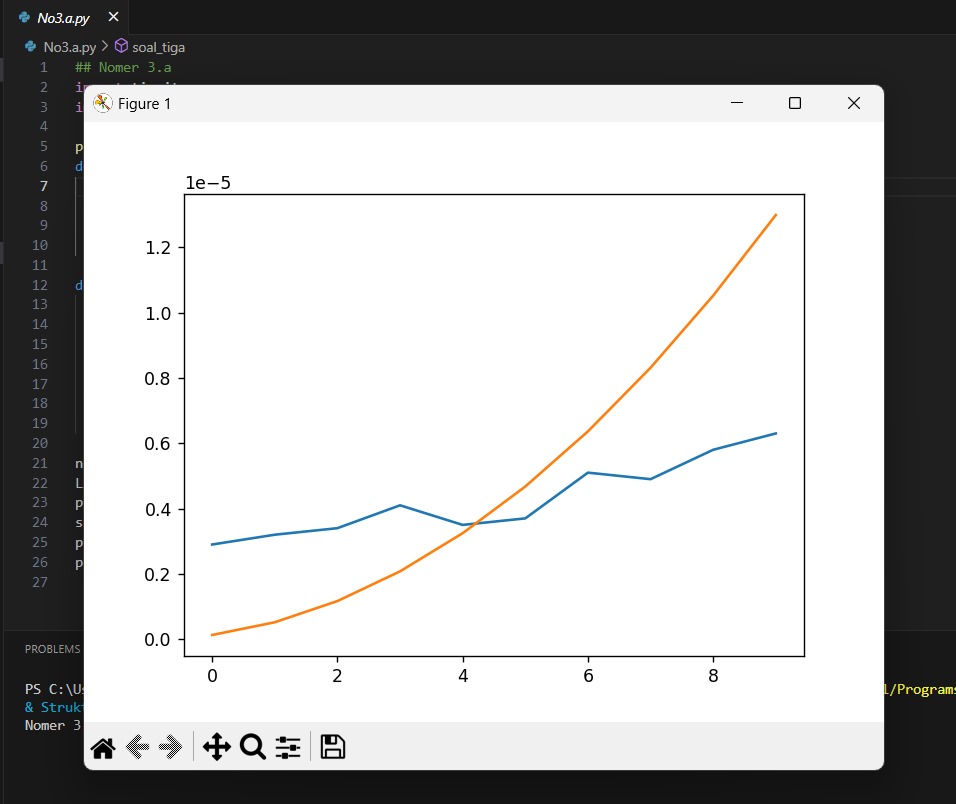
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. kode

##Nomer 3.b

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.b")

def soal\_tiga(n):

    test = 0

    for i in range(n):

        for j in range(i):

            test = test + i \* j

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

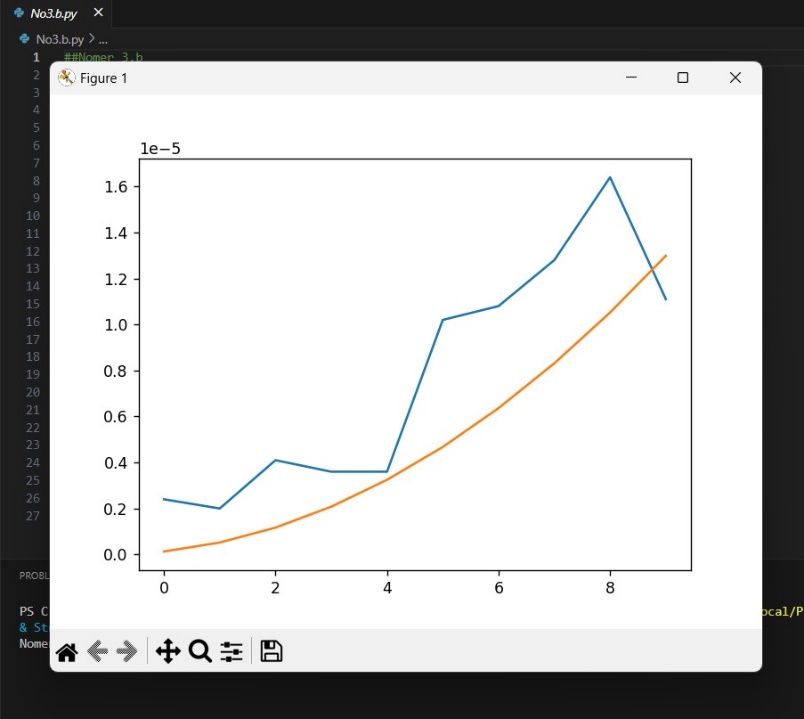
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. Kode

##Nomer 3.c

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.c")

def soal\_tiga(n):

    test = 0

    for i in range(n):

        test = test+1

    for j in range(n):

        test = test - 1

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

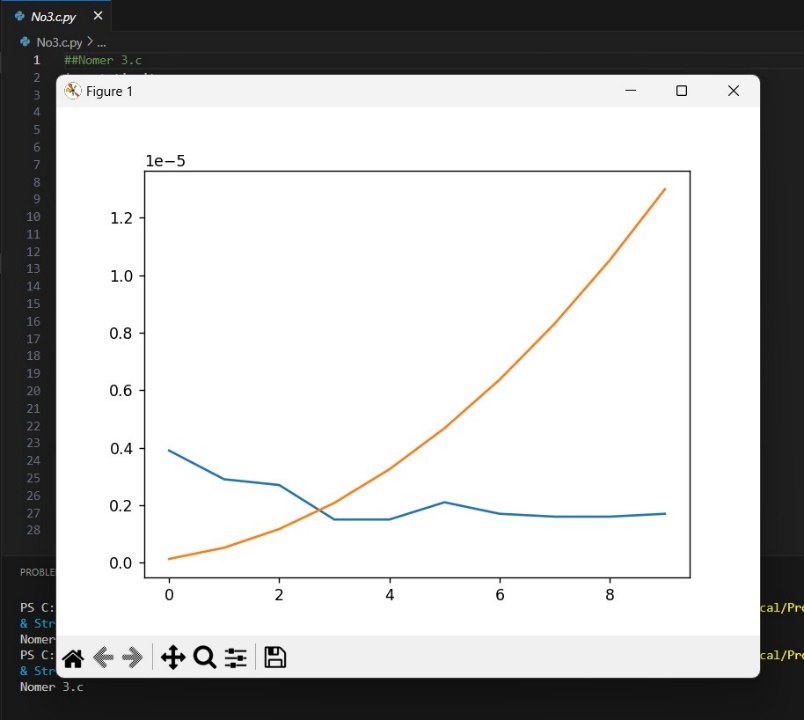
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. Kode

##Nomer 3.d

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.d")

def soal\_tiga(n):

    i = n

    while i > 0:

        k = 2 + 2

        i = i // 2

def uji\_soal\_tiga(n):

    ls = []

    jangkauan = range(1, n+1)

    siap = "from \_\_main\_\_ import soal\_tiga"

    for i in jangkauan:

        ##print('i = ',i)

        t = timeit.timeit("soal\_tiga(" + str(i) + ")", setup=siap, number=1)

        ls.append(t)

    return ls

n = 1000

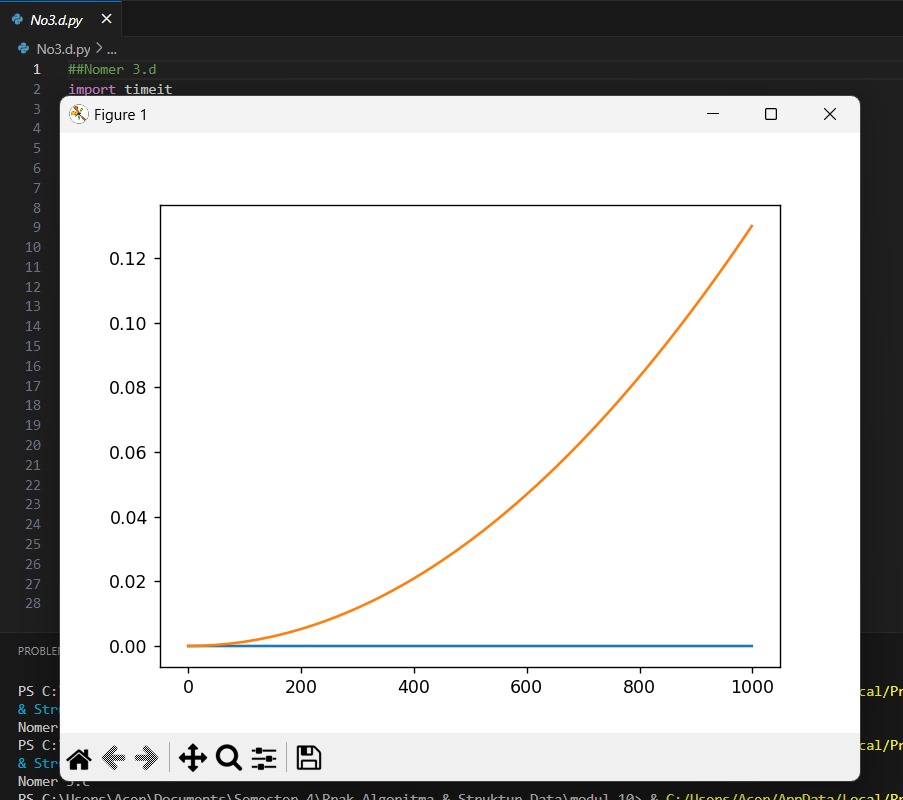
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. Kode

##Nomer 3.e

import timeit

import matplotlib.pyplot as plt

print("Nomer3.e")

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

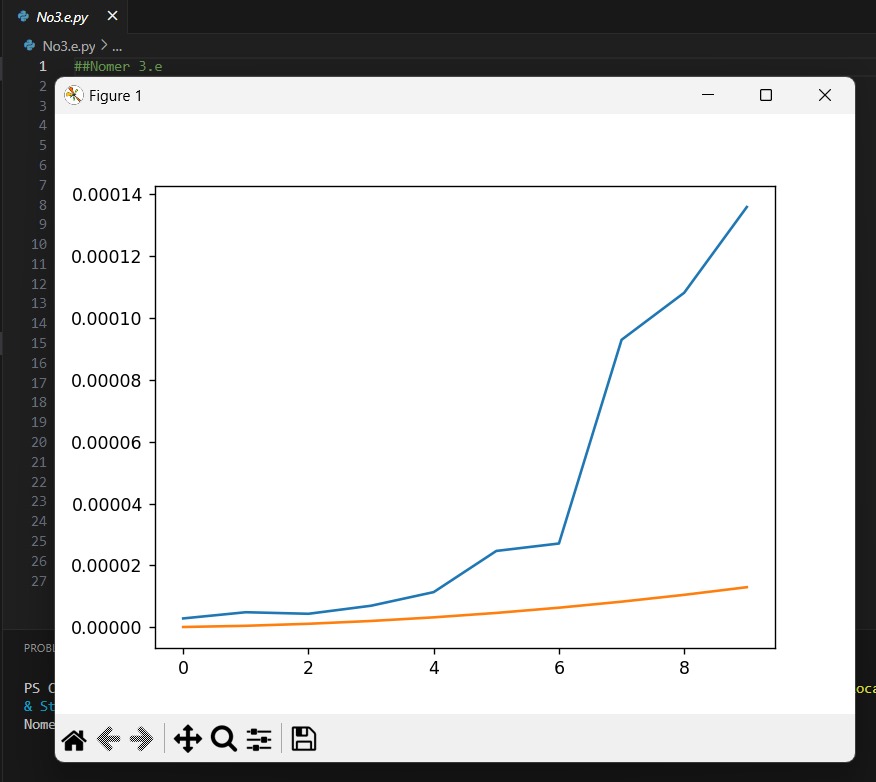
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. kode

##Nomer 3.f

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.f")

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

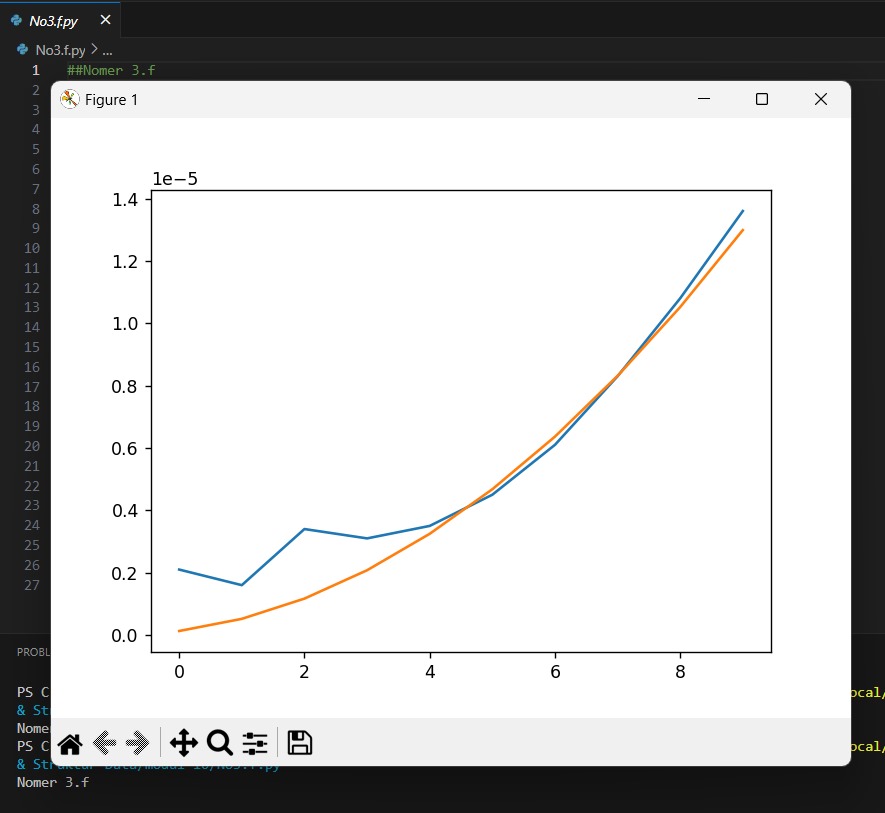
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



1. kode

##Nomer 3.g

import timeit

import matplotlib.pyplot as plt

print("Nomer 3.g")

def soal\_tiga(n):

    for i in range(n):

        if i % 3 == 0:

            for j in range(n // 2):

                j+=j

        elif i % 2 == 0:

            for j in range(5):

                j+=j

        else:

            for j in range(n):

                j+=j

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

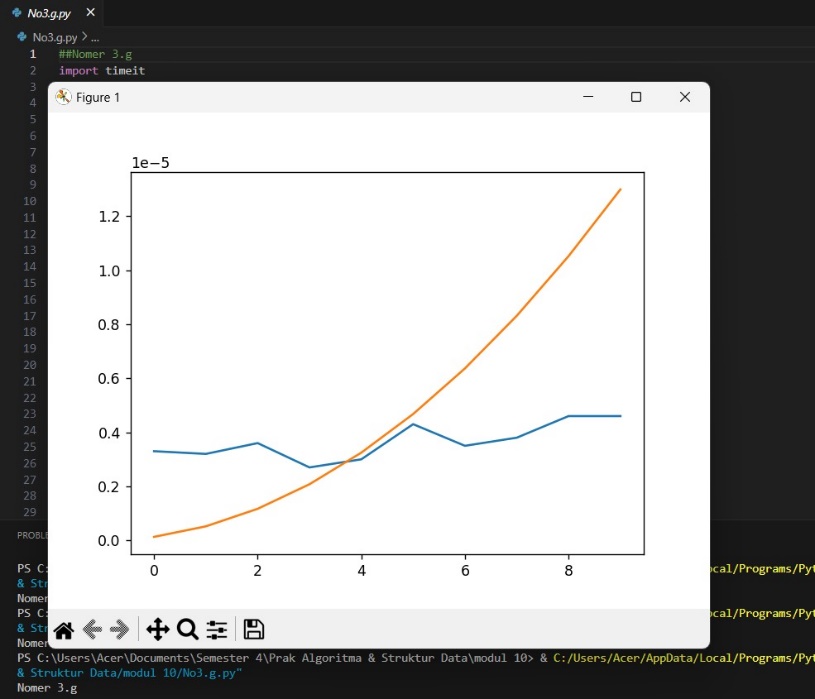
LS = uji\_soal\_tiga(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



7. Buatlah suatu uji coba untuk mengkonfirmasi bahwa metode append() adalah O(1). Gu nakan timeit dan matplotlib, seperti sebelumnya.

##Nomer 7

import timeit

import matplotlib.pyplot as plt

print("Nomer 7")

def soal\_tujuh(n):

    L = list(range(30))

    L = L[::-1]

    for i in range(n):

        L.append(n)

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

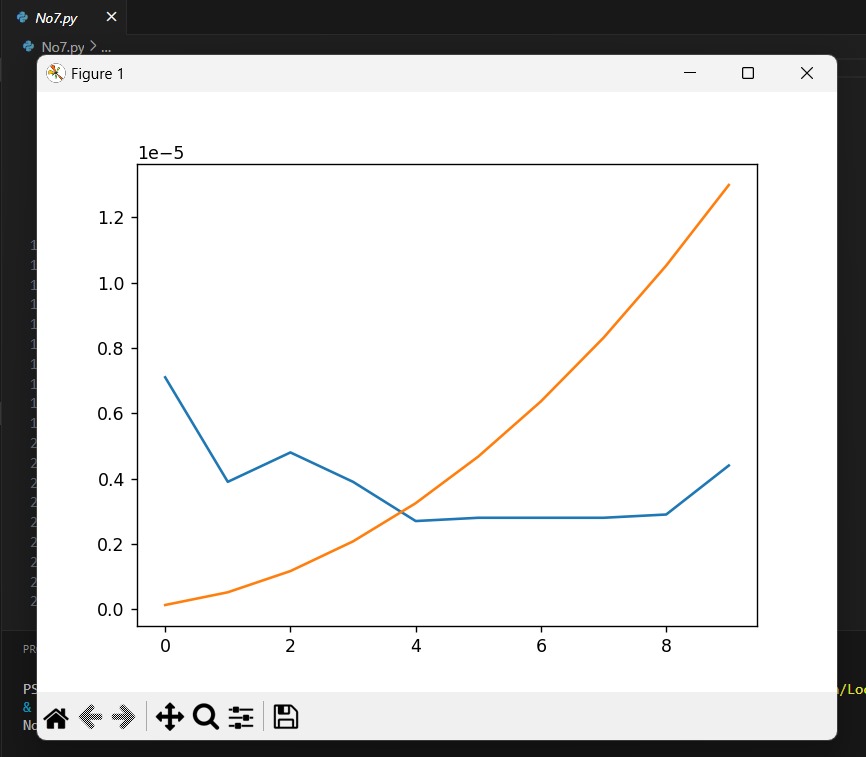
LS = uji\_soal\_tujuh(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



8. Buatlah suatu uji coba untuk mengkonfirmasi bahwa metode insert() adalah O(n). Gu nakan timeit dan matplotlib, seperti sebelumnya.

##Nomer 8

import timeit

import matplotlib.pyplot as plt

print("Nomer 8")

def soal\_tujuh(n):

    L = list(range(30))

    L = L[::-1]

    for i in range(n):

        for b in range(n):

            L.insert(i,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

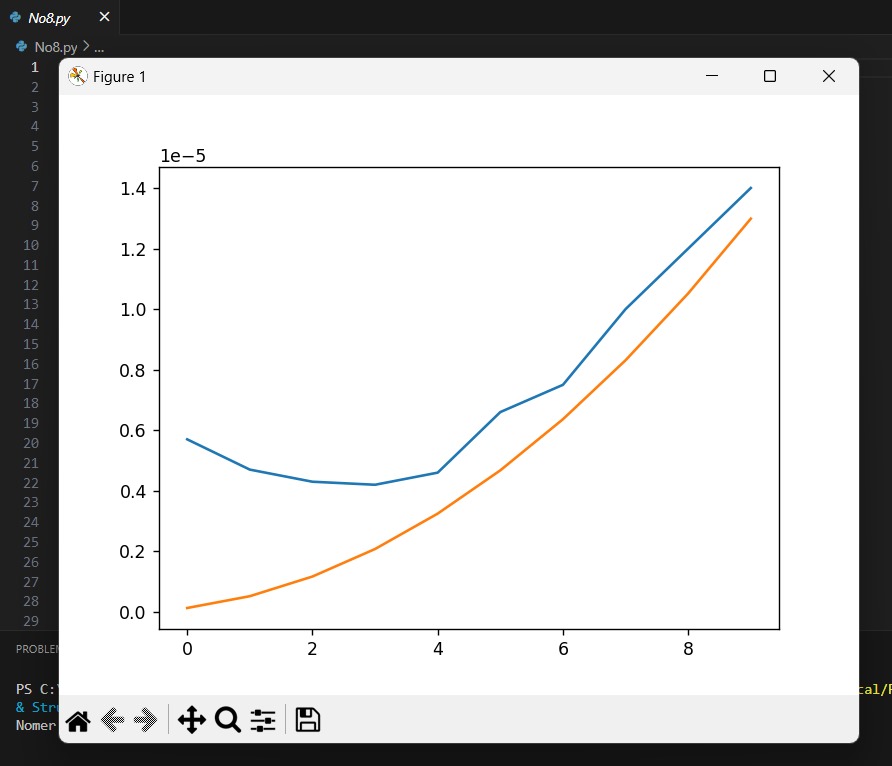
LS = uji\_soal\_tujuh(n)

plt.plot(LS)

skala = 7700000

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

plt.show()



9. Buatlah suatu uji coba untuk mengkonfirmasi bahwa untuk memeriksa apakah-suatu-nilai berada-di-suatu-list mempunyai kompleksitas O(n). Gunakan timeit dan matplotlib, seperti sebelumnya.

##Nomer 9

import timeit

import time

print("Nomer 9")

def carilurus(wadah, target):

    n = len(wadah)

    for i in range(n):

        if wadah[i] == target:

            return True

    return False

def tim():

    z=100

    a = [8, 7, 2, 1, 3, 2, 10]

    awal = time.time()

    U = carilurus(a, z)

    akhir=time.time()

    print("==========================worst case ")

    print("mengurutkan %d bilangan, memerlukan %8.7f detik" %(U,akhir-awal))

tim()

import matplotlib.pyplot as plt

def tim\_matlib(n):

    z=100

    a = [8, 7, 2, 1, 3, 2, 10]

    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 = 10

LS = uji\_soal\_sembilan(n)

plt.plot(LS)

skala = 7700000

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

plt.show()

