­



**NAMA : KEVIN AVICENNA WIDIARTO**

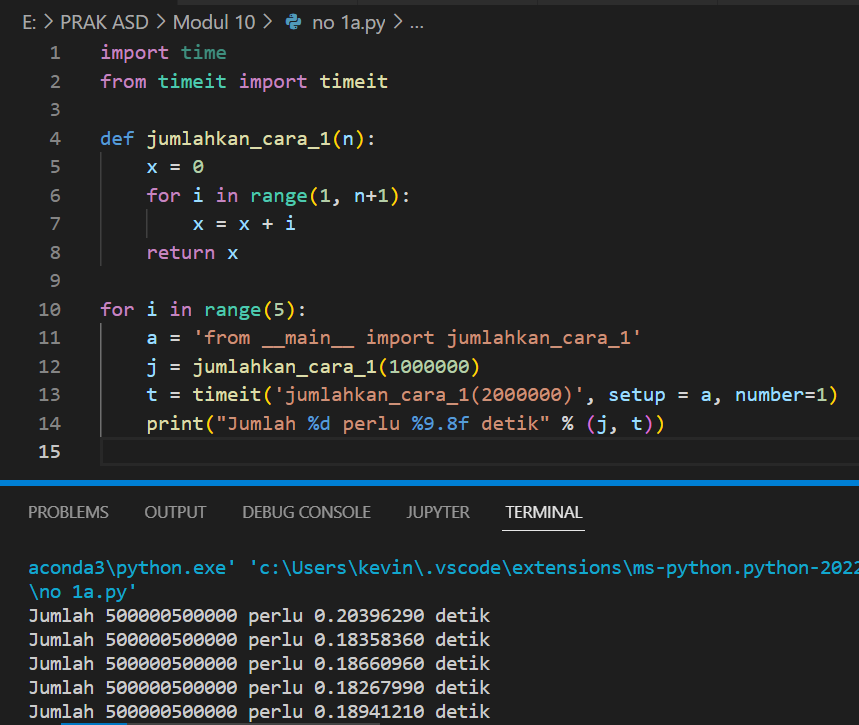
**NIM : L200200183**

**Modul : 4**

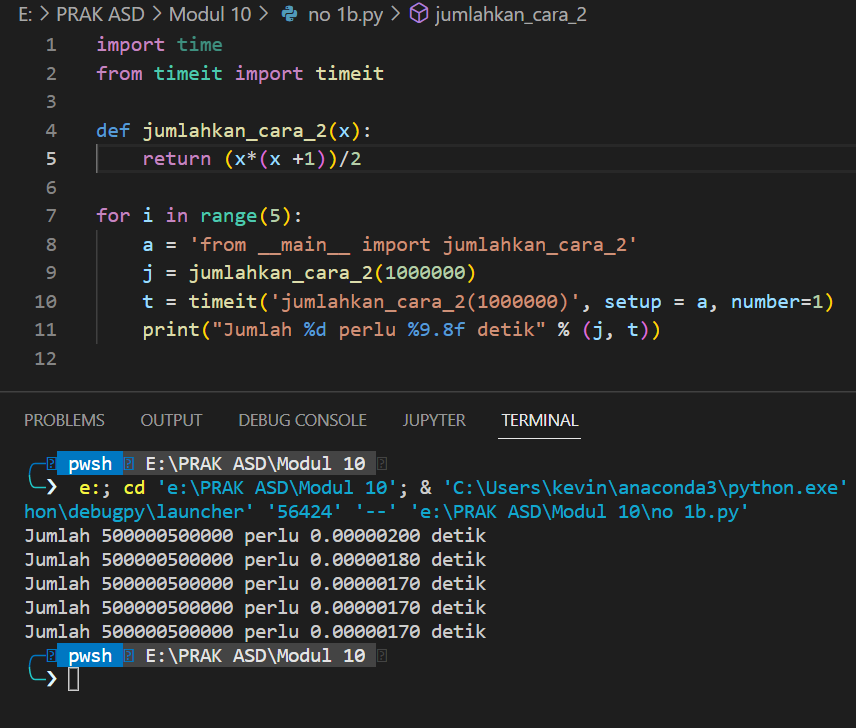
**Praktikum Algoritma Struktur Data**

**MODUL 10**

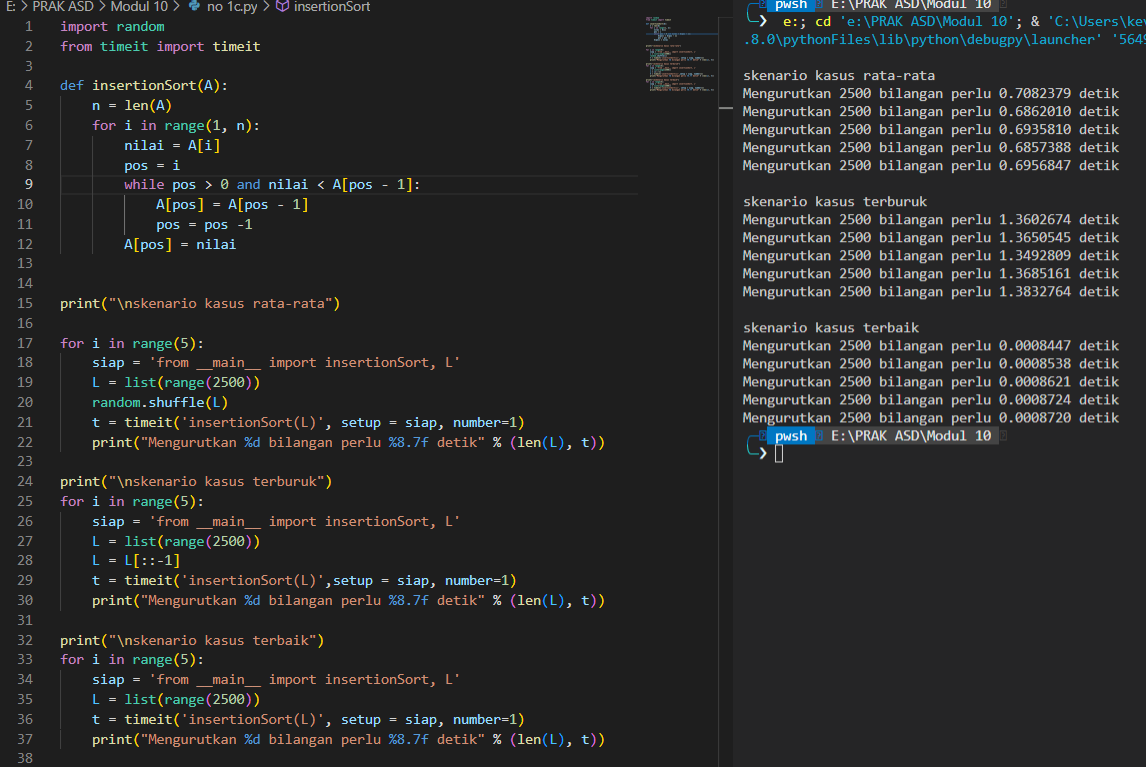
**No 1a**



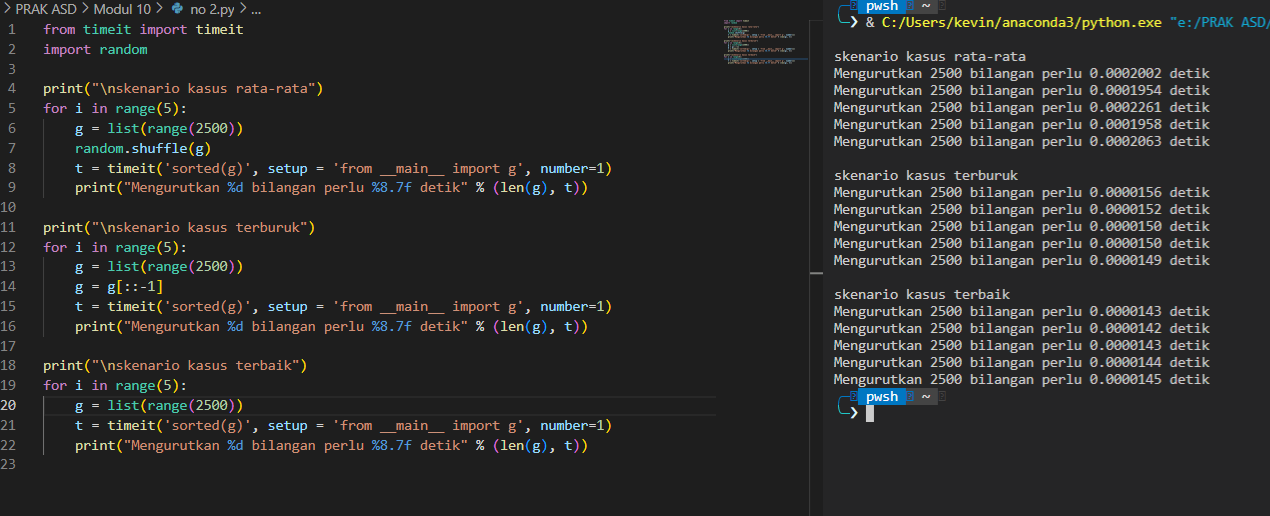
No 1b



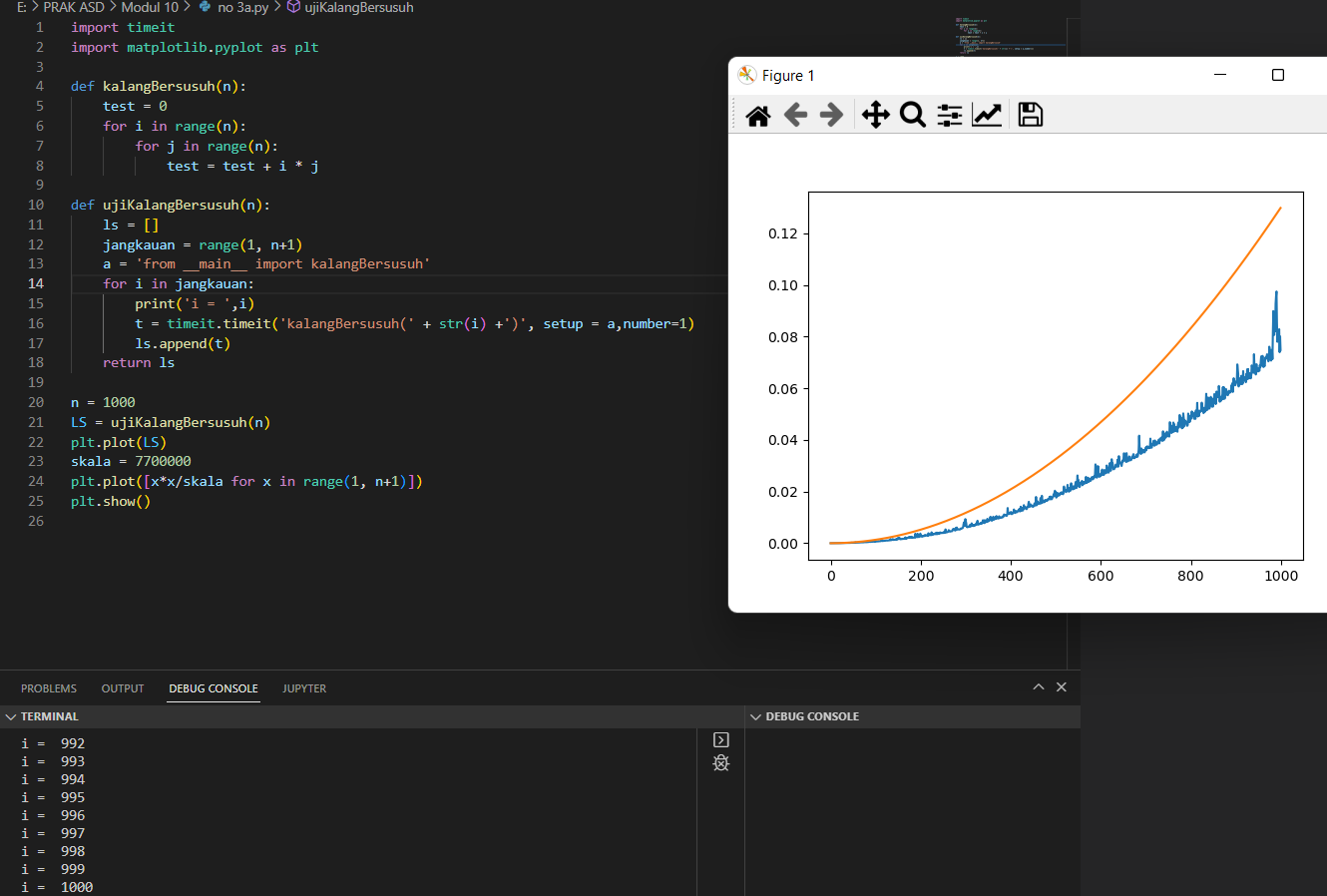
No 1c



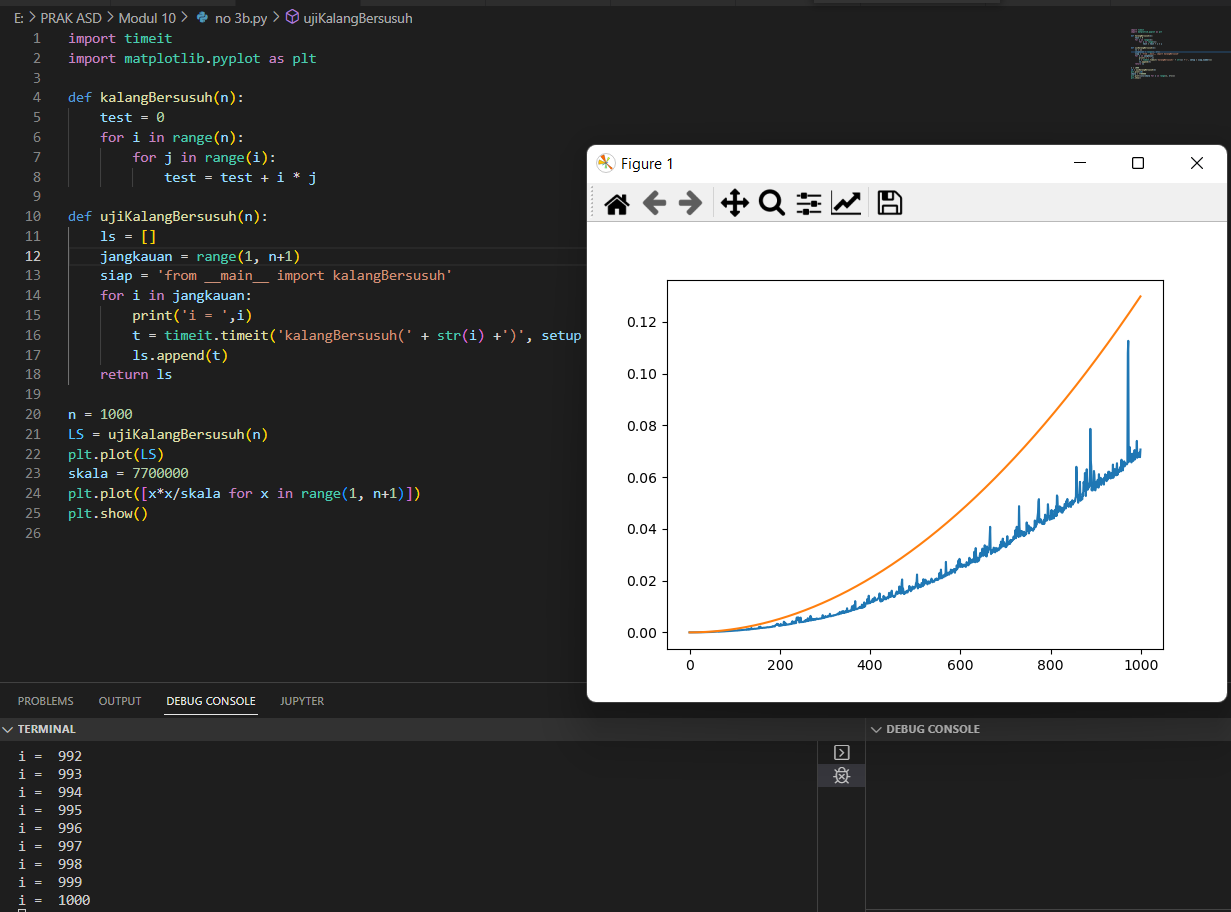
No 2



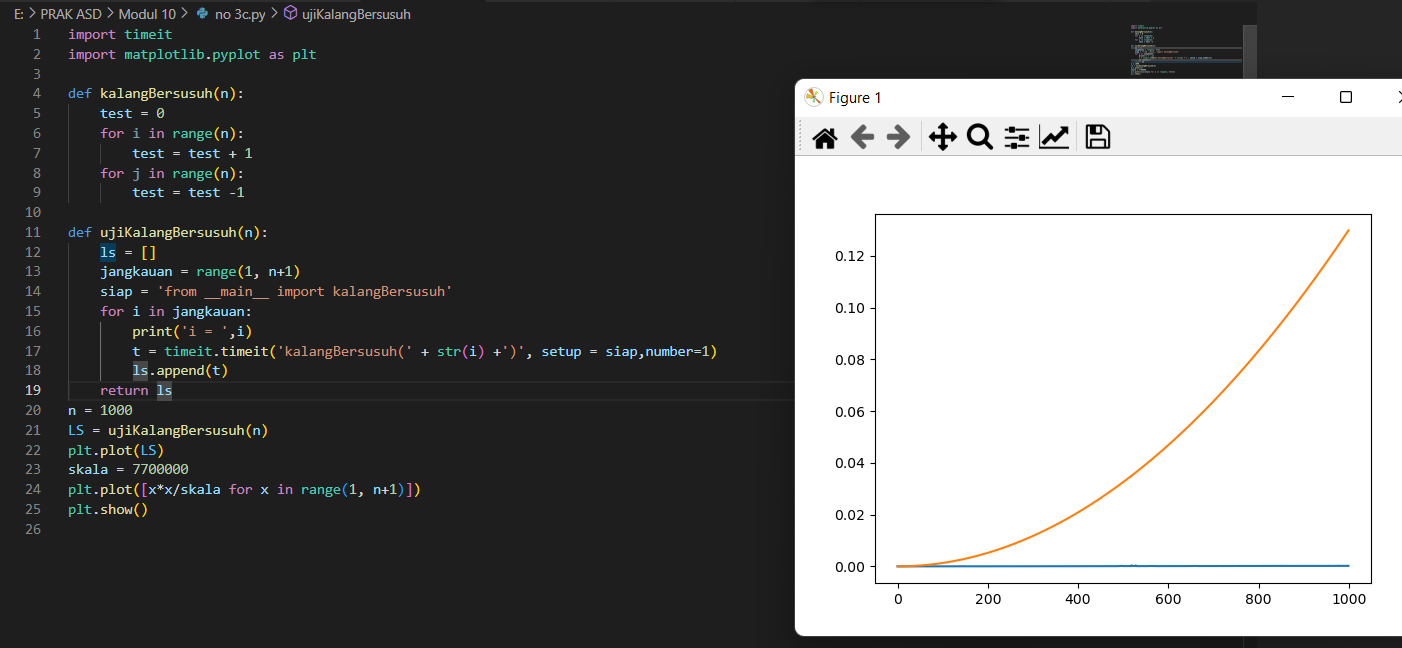
No 3a



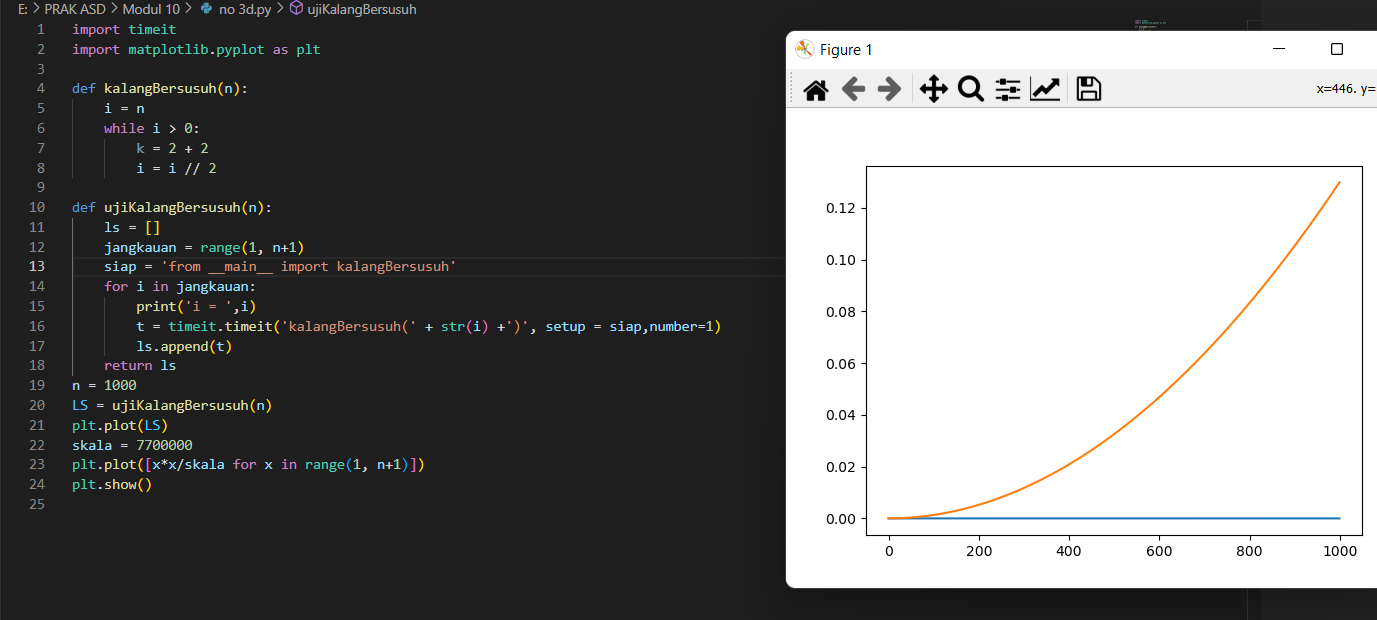
No 3b



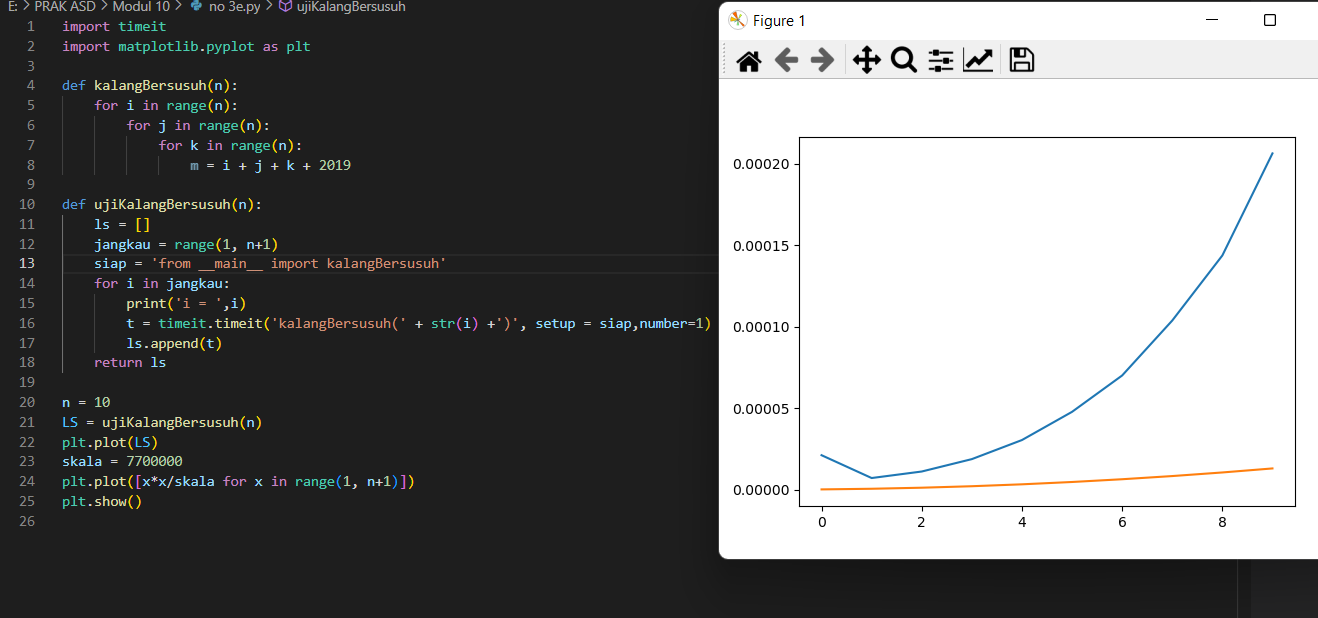
No 3c



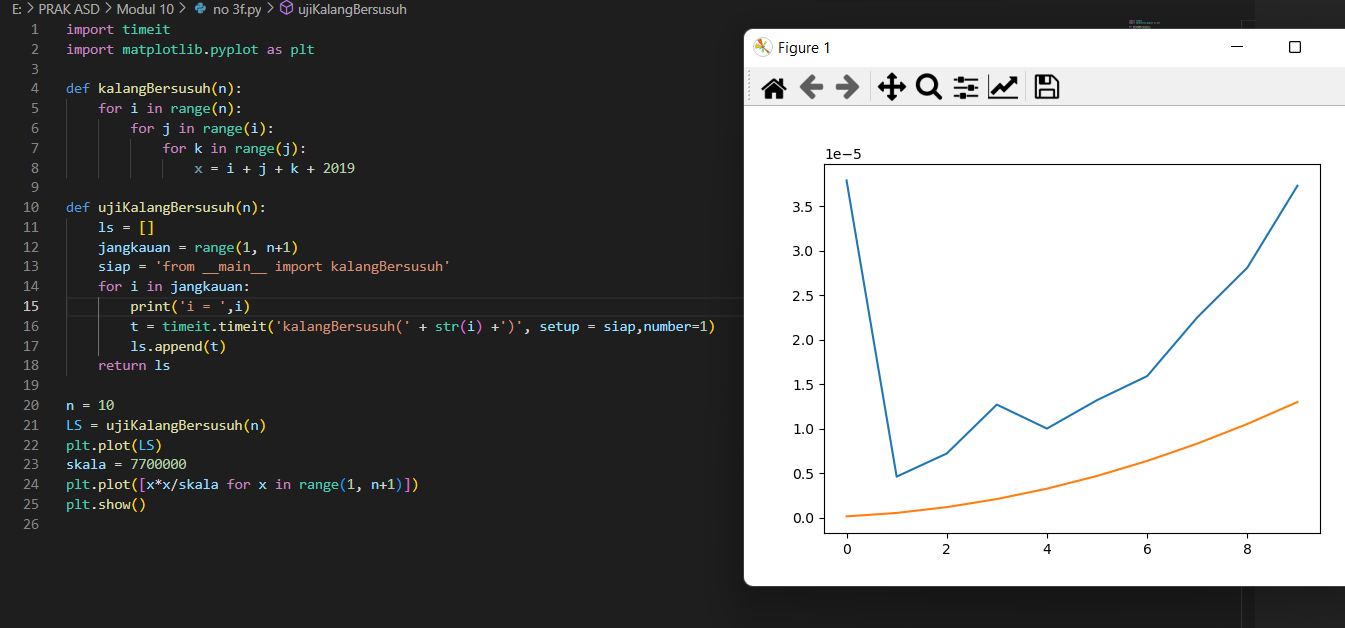
No 3d



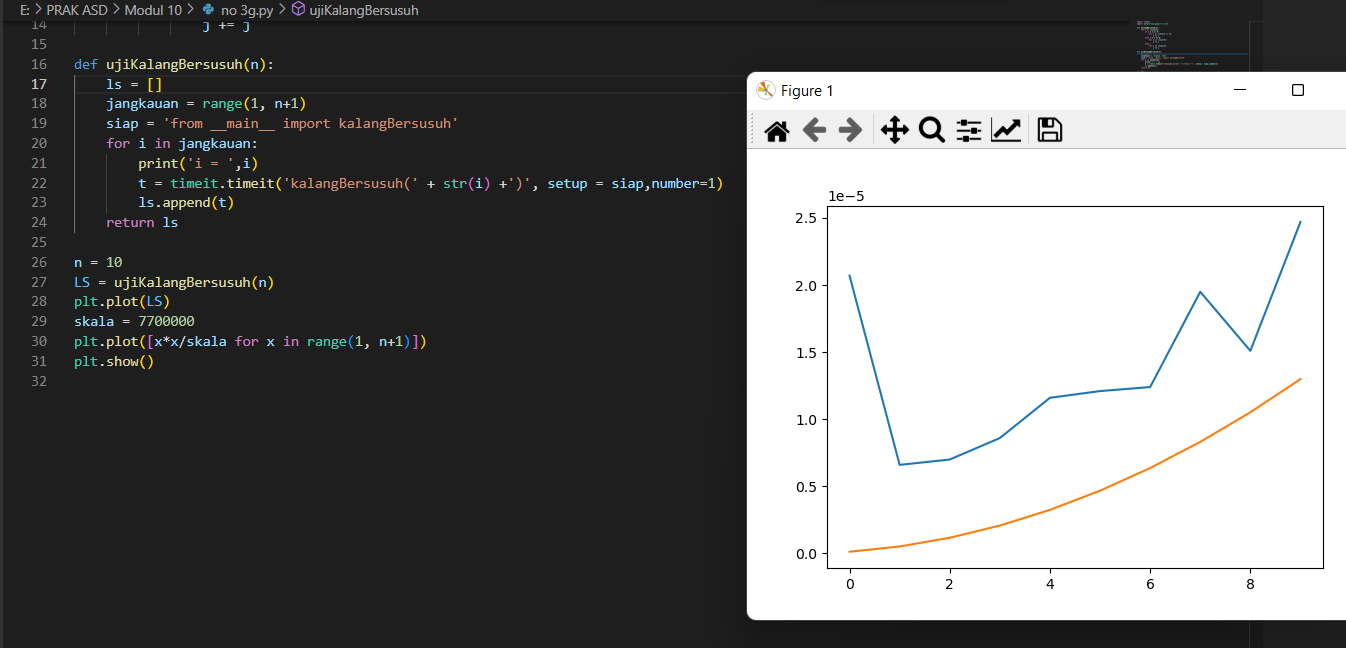
No 3e



No 3f



No 3g



Nomor 4

Mengurutkan kompleksitasnya lambat ke yang cepat.

n log2 n 4n 10 log2 n 5n2 log4 n 12n6 2 log 2n n3

= log4 n 2 log 2n 10 log2 n 5n2 n3 12n6 4n

Nomor 5

Menentukan O(·) dari fungsi-fungsi berikut:

a. T(n) = n2 + 32n + 8

= O(n2)

b. T(n) = 87n + 8n

= O(n)

c. T(n) = 4n + 5n log n + 102

= O(n)

d. T(n) = log n + 3n2 + 88

= O(n2)

e. T(n) = 3(2n ) + n2 + 647

= O(n2)

f. T(n, k) = kn + log k

= O(kn)

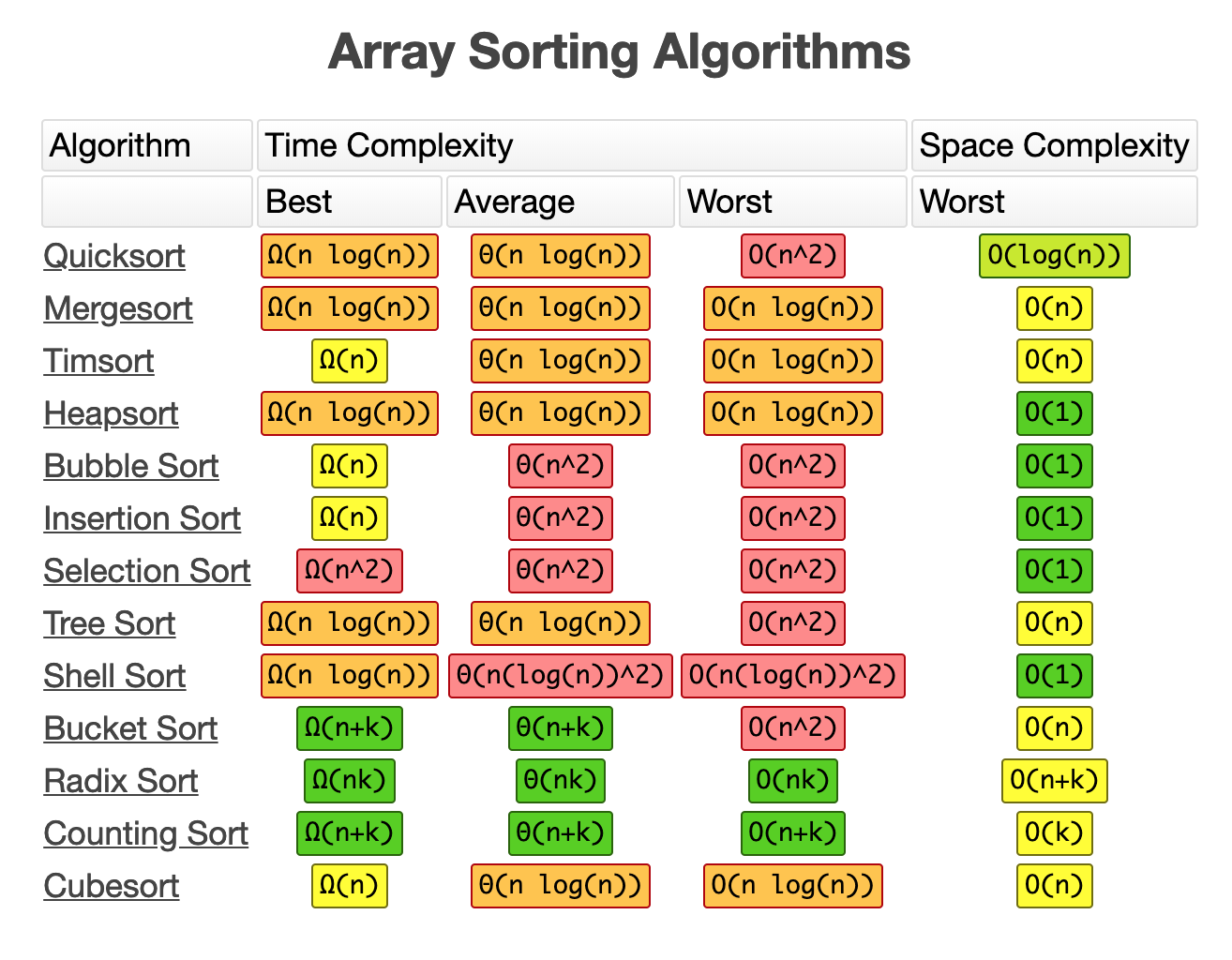
g. T(n, k) = 8n + k log n + 800

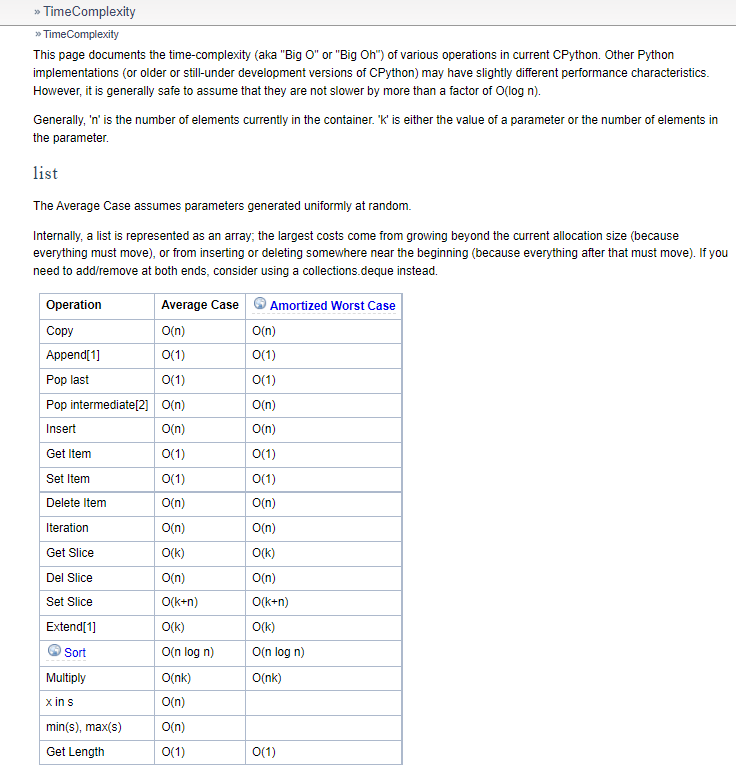
= O(n)

h. T(n, k) = 100kn + n

= O(nk)

NO 6

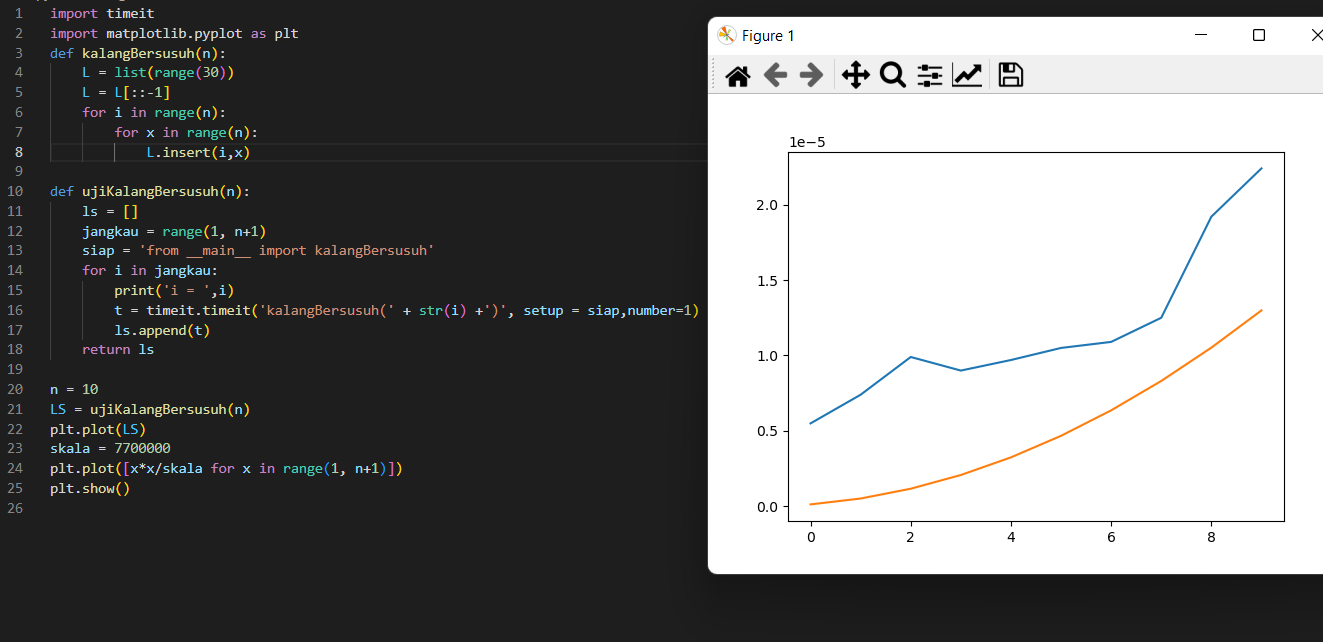




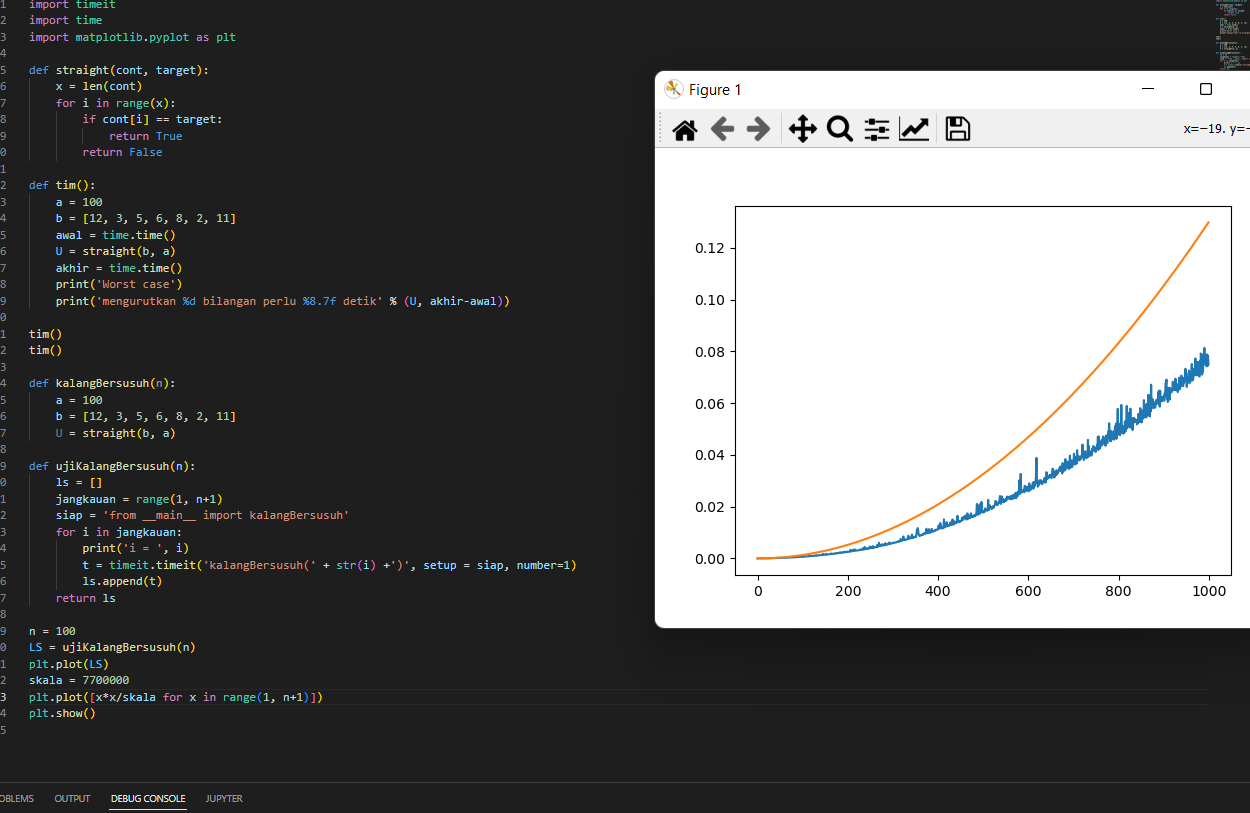
No 7



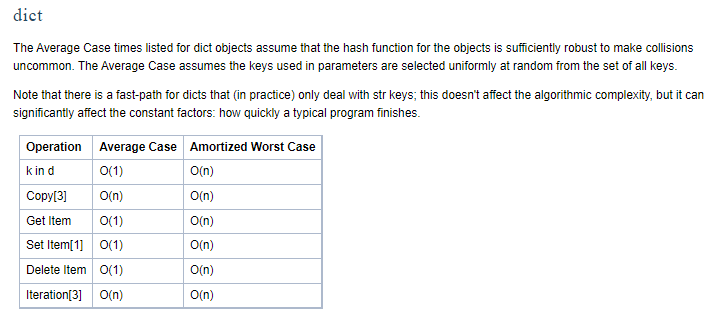
No 8



No 9



No 10



No 11

|  |  |  |
| --- | --- | --- |
| **Big-O O(·)** | **Big-Theta Θ(·)** | **Big-Omega Ω(·)** |
| Tingkat pertumbuhan  algoritma kurang dari  atau sama dengan nilai  tertentu (<=). | Tingkat pertumbuhan  sama dengan nilai  yang ditentukan (==). | Tingkat pertumbuhan  lebih bsar dari atau  sama dengan nilai  yang ditentukan ( >=). |
| Batas atas algoritma  diwakili oleh notasi BigO. Hanya fungsi di atas  yang dibatasi oleh BigO. ikatan atas asimtotik  yang diberikan oleh  notasi Big-O. | Pembatas fungsi dari  atas dan bawah  diwakili oleh notasi  theta. Perilaku  asimptotik yang tepat  dilakukan oleh notasi  theta ini. | Batas bawah  algoritma diwakili  oleh notasi Omega.  Ikatan bawah  asimptotik diberikan  oleh notasi Omega |
| Didefinisikan sebagai  batas atas (upper bound)  dan batas atas pada  algoritma adalah jumlah  waktu terbanyak yang  dibutuhkan (kinerja  kasus terburuk). | Didefinisikan sebagai  terikat paling ketat  (tight bound) dan  terikat paling ketat  adalah yang terbaik  dari semua kasus  terburuk yang dapat  diambil oleh  algoritma | Didefinisikan sebagai  batas bawah (lower  bound) dan batas  bawah pada algoritma  adalah jumlah waktu  paling sedikit yang  dibutuhkan (cara  seefesien mungkin,  dengan kata lain kasus  terbaik) |

No 12

Amortized Analysis = Metode untuk menganalisis kompleksitas waktu algoritma · Push(S,x) O(1) memasukkan objek x ke dalam stack,digunakan untuk algoritma yang mana operasi terkadang berjalan sangat lambat, tetapi sebagian besar operasi lainnya lebih cepat. Amortized analysis menganalisis uruta operasi dan menjamin waktu rata-rata kasus terburuk yang lebih rendah daripada waktu kasus terburuk dari operasi yang sangat mahal.