

## **TUGAS TRUKTUR DATA**

DosenPengampu:

AdamBachtiarS,kom,M,MT



**Disusun oleh :**

Nama:Abi manyu

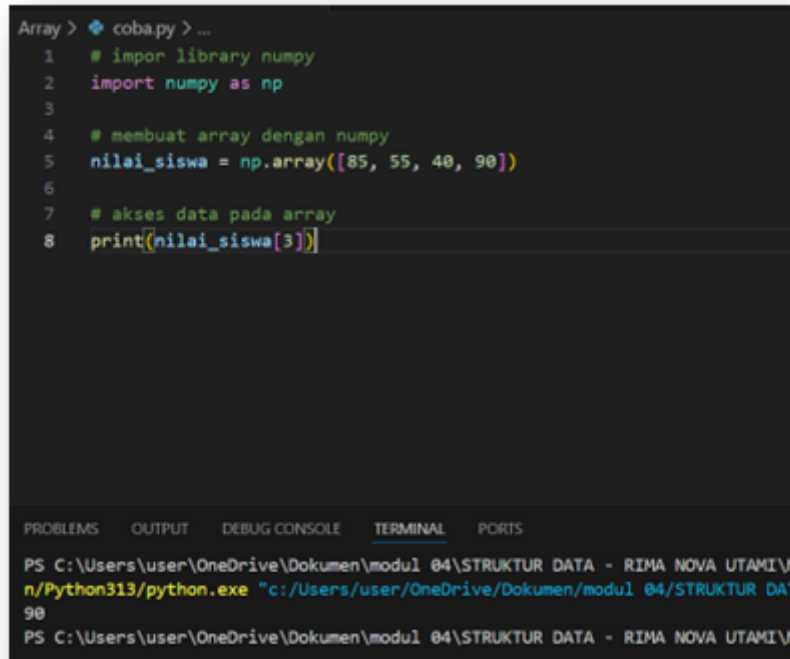
rezza saputra

Nim : 24241106

Kelas:c

**PROGRAMSTUDIPENDIDIKANTEKNOLOGIINFORMASI  
FAKULTAS SAINS, TEHNIK DAN TERAPAN  
UNIVERSITASPENDIDIKANMANDALIKAMATARAM TAHUN  
2025**

## PRAKTEKKE1



```
Array > cobapy > ...
1  # impor library numpy
2  import numpy as np
3
4  # membuat array dengan numpy
5  nilai_siswa = np.array([85, 55, 40, 90])
6
7  # akses data pada array
8  print(nilai_siswa[3])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAMI\N  
n/Python313/python.exe "c:/Users/user/OneDrive/Dokumen/modul 04/STRUKTUR DA  
90  
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAMI\N

### Baris2

```
import numpy as np
```

Baris ini **mengimpor library** bernama **numpy** dan memberikan alias **np**, sehingga Anda bisa menggunakan fungsi-fungsi NumPy dengan menulis **np.nama\_fungsi()**.

NumPy adalah library Python yang sangat kuat untuk perhitungan numerik dan manipulasi array.

### Baris5

```
nilai_siswa = np.array([85, 55, 40, 90])
```

Anda membuat sebuah **array NumPy** satu dimensi yang berisi data nilai-nilai siswa: 85, 55, 40, 90.

Ini berbeda dari list biasa Python. Array NumPy lebih efisien dan memiliki banyak fitur tambahan seperti operasi vektor/matriks.

### Baris8

```
print(nilai_siswa[3])
```

Anda mencetak nilai pada indeks ke-3 dari array **nilai\_siswa**.

Dalam Python (dan NumPy), indeks dimulai dari 0, sehingga:

- nilai\_siswa[0] → 85
- nilai\_siswa[1] → 55
- nilai\_siswa[2] → 40
- nilai\_siswa[3] → 90 (yang dicetak)

Jadi, output dari program ini adalah:

90

## PERAKTEKKE2

```
12
13 # membuat array dengan numpy
14 nilai_siswa_1 = np.array([75, 65, 45, 80])
15 nilai_siswa_2 = np.array([[85, 55, 40], [50, 40, 99]])
16
17 # cara akses elemen array
18 print(nilai_siswa_1[0])
19 print(nilai_siswa_2[1][1])
20
21 # mengubah nilai elemen array
22 nilai_siswa_1[0] = 88
23 nilai_siswa_2[1][1] = 70
24
25 # cek perubahannya dengan akses elemen array
26 print(nilai_siswa_1[0])
27 print(nilai_siswa_2[1][1])
28
29 # Cek ukuran dan dimensi array
30 print("Ukuran Array : ", nilai_siswa_1.shape)
31 print("Ukuran Array : ", nilai_siswa_2.shape)
32 print("Dimensi Array : ", nilai_siswa_2.ndim)
33
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\user\OneDrive\Dokumen\modul 04\S
n/Python313/python.exe "c:/Users/user/OneDri
75
48
88
70
Ukuran Array : (4,)
Ukuran Array : (2, 3)
Dimensi Array : 2
PS C:\Users\user\OneDrive\Dokumen\modul 04\S
```

### Baris13

```
import numpy as np
```

Mengimpor library **NumPy** dengan alias **np**.

### Baris14–15

```
nilai_siswa_1 = np.array([75, 65, 45, 80])
```

```
nilai_siswa_2 = np.array([[85, 55, 40], [50, 40, 99]])
```

- nilai\_siswa\_1: array **1 dimensi** dengan 4 elemen.
- nilai\_siswa\_2: array **2 dimensi (2 baris × 3 kolom)**.

### Baris18–19: Akses elemen array

```
print(nilai_siswa_1[0])    # Output: 75
```

```
print(nilai_siswa_2[1][1]) # Output: 40
```

- nilai\_siswa\_1[0]: elemen pertama (75)
- nilai\_siswa\_2[1][1]: baris ke-2, kolom ke-2 → 40

### Baris22–23: Ubah nilai elemen array

```
nilai_siswa_1[0] = 88
```

```
nilai_siswa_2[1][1] = 70
```

- Elemen pertama nilai\_siswa\_1 diubah dari 75 → 88
- Elemen baris ke-2 kolom ke-2 nilai\_siswa\_2 dari 40 → 70

### Baris26–27: Cek perubahan

```
print(nilai_siswa_1[0])    # Output: 88
```

```
print(nilai_siswa_2[1][1])#Output:70
```

### **Baris30–32:Cekukuran&dimensi**

```
print("UkuranArray:",nilai_siswa_1.shape)
```

```
print("UkuranArray:",nilai_siswa_2.shape)
```

```
print("DimensiArray:",nilai_siswa_2.ndim)
```

- **.shape:menunjukkanukuran/tataletakarray**
  - nilai\_siswa\_1.shape→(4,)→array1dimensidengan4elemen
  - nilai\_siswa\_2.shape→(2,3)→2baris,3kolom
- **.ndim:menunjukkanjumlahdimensi**
  - nilai\_siswa\_2.ndim→2→array2D

### **RingkasanOutput:**

75

40

88

70

Ukuran Array :(4,)

UkuranArray:(2,3)

Dimensi Array :2

### PERAKTEKKE3

```
Array > coba.py > ...
1 import numpy as np
2
3
4 # membuat array
5 a = np.array([1, 2, 3])
6 b = np.array([4, 5, 6])
7
8 # menggunakan operasi penjumlahan pada 2 array
9 print(a + b)      # array([5, 7, 9])
10
11 # Indexing dan Slicing pada Array
12 arr = np.array([10, 20, 30, 40])
13 print(arr[1:3])   # array([20, 30])
14
15
16 # iterasi pada array
17 for x in arr:
18     print(x)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - R
n/Python313/python.exe "c:/Users/user/OneDrive/Dokumen/modul
[5 7 9]
[20 30]
10
20
30
40
```

### KODEPROGRAMDENGANPENJELASAN:

Membuatduaarray1dimensi

```
a=np.array([1,2,3])
```

```
b=np.array([4,5,6])
```

adanbadalaharrayNumPysatudimensi. Isi

array:

- a= [1, 2,3]
- b= [4,5,6]

Penjumlahanduaarray

```
print(a+b)      #array([5,7,9])
```

Ini melakukan **penjumlahan elemen per elemen** (bukan menjumlahkan semua angka).

Hitungannya:

- $1 + 4 = 5$
- $2 + 5 = 7$
- $3 + 6 = 9$




Hasil: [5, 7, 9]

Indexing dan slicing pada array

```
arr = np.array([10, 20, 30, 40])
```

```
print(arr[1:3]) #array([20,30])
```

`arr[1:3]` artinya ambil elemen dari **indeks 1 sampai sebelum 3**:

- indeks 0 = 10
- indeks 1 = 20 
- indeks 2 = 30 
- indeks 3 = 40  (tidak diambil)

Hasil: [20, 30]

Iterasi (perulangan) pada array

```
for x in arr:
```

```
    print(x)
```

Ini akan mencetak **semua elemen** dalam array satu persatu: 10

20

30

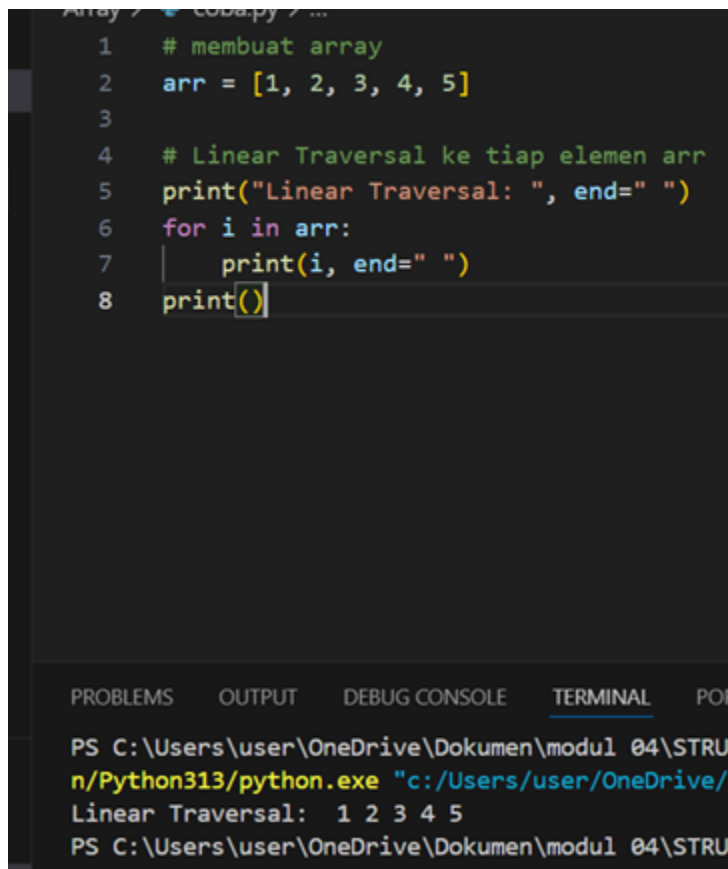
40

## RINGKASAN FUNGSI YANG DIPAKAI

Fungsi/ Konsep	Penjelasan
<code>np.array(...)</code>	Membuat array dari list

<code>a + b</code>	Menjumlahkan elemen array perposisi
<code>arr[1:3]</code>	Mengambil sebagian isi array (slicing)
<code>for x in arr:</code>	Mengulang setiap elemen di dalam array

#### 4. PERAKTEKKE4



```

1  # membuat array
2  arr = [1, 2, 3, 4, 5]
3
4  # Linear Traversal ke tiap elemen arr
5  print("Linear Traversal: ", end=" ")
6  for i in arr:
7      print(i, end=" ")
8  print()

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORT

```

PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA\Python313\python.exe "c:/Users/user/OneDrive/Dokumen/modul 04/STRUKTUR DATA/Python313/python.exe"
Linear Traversal:  1 2 3 4 5
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA\Python313\python.exe

```

1. Membuat array (dalam bentuk list biasa, bukan NumPy) `arr = [1, 2, 3, 4, 5]`  
`arr` adalah list biasa di Python (bukan array dari NumPy). List ini berisi 5 elemen: [1, 2, 3, 4, 5]
2. Linear Traversal ke tiap elemen `arr`



```
print("LinearTraversal:",end="")
```

Baris ini mencetak teks "LinearTraversal:" tanpa pindah baris, karena `end=""` mengganti karakter akhir default `\n` (newline) menjadi spasi.

for i in arr:

```
    print(i,end="")
```

Ini adalah loop for untuk mengakses setiap elemen dalam list arr.

- i akan bernilai 1, lalu 2, lalu 3, lalu 4, lalu 5.
- Setiap pangkat dicetak di baris yang sama, karena `end=""` print()

Ini mencetak baris kosong untuk mengakhiri output traversal tadi, agar kursor turun ke baris baru setelah selesai.

OUTPUT PROGRAM:

LinearTraversal:12345

APA ITU LINEAR TRAVERSAL?

Linear traversal adalah proses menelusuri atau mengunjungi setiap elemen dalam urutan satu per satu, dari awal sampai akhir.

## 5. PERAKTEKKE5

```
Array > coba.py > ...
1  # membuat array
2  arr = [1, 2, 3, 4, 5]
3
4  # Reverse Traversal dari elemen akhir
5  print("Reverse Traversal: ", end="")
6  for i in range(len(arr) - 1, -1, -1):
7      |   print(arr[i], end=" ")
8  print()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR
n/Python313/python.exe "c:/Users/user/OneDrive/Doku
Reverse Traversal: 5 4 3 2 1
```

### KODE PROGRAM DAN PENJELASAN

#### 1. Membuat array (list)

```
arr = [1, 2, 3, 4, 5]
```

Kamu membuat sebuah **list** Python yang berisi angka: [1, 2, 3, 4, 5]

#### 2. Traversal mundur (dari belakang ke depan)

```
print("Reverse Traversal: ", end="")
```

🔴 Inimencetak teks "Reverse Traversal:" tanpa pindah baris karena `end=""`.

```
for i in range(len(arr)-1, -1, -1): print(arr[i],
    end=" ")
```

Penjelasan bagian `range(len(arr)-1, -1, -1)`:

- `len(arr)-1` → posisi indeks terakhir → 4
- `-1` → batas akhir (**tidak termasuk -1**) → jadi sampai 0
- `-1` → langkah mundur

Jadi, `range(4,-1,-1)` menghasilkan:

4,3,2,1,0

Kemudian `arr[i]` mencetak elemen berdasarkan indeks itu:

- `arr[4] → 5`
- `arr[3] → 4`
- `arr[2] → 3`
- `arr[1] → 2`
- `arr[0] → 1`

`print()`

Ini untuk **pindahbaris** setelah traversal selesai.

#### **OUTPUT PROGRAM:**

Reverse Traversal: 54321

#### **CATATAN TAMBAHAN:**

##### **Penjelasan**

`range(start, stop, step)` Membuat urutan angka dari start ke stop (tidak termasuk), dengan langkah step

`len(arr)` Mengembalikan jumlah elemen dalam list

`end=""` Mencegah pindahbaris setelah print, diganti dengan spasi

[TextWrappingBreak] Kalau kamu ingin invers **terbalik otomatis** tanpa for, bisa juga pakai: `for i in`

`reversed(arr):`

`print(i, end="")`

#### **6. PERAKTEKKE6**

```
1 # membuat array
2 arr = [1, 2, 3, 4, 5]
3
4 # mendeklarasikan nilai awal
5 n = len(arr)
6 i = 0
7
8 print("Linear Traversal using while loop: ", end=" ")
9 # linear Traversal dengan while
10 while i < n:
11     print(arr[i], end=" ")
12     i += 1
13 print()
```

BLISS OUTPUT DEBUG CONSOLE TERMINAL PORTS

C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA  
python313/python.exe "C:/Users/user/OneDrive/Dokumen/modul 04/STRUKTUR DATA - RIMA NOVA  
Linear Traversal using while loop: 1 2 3 4 5  
C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA

## KODE DAN PENJELASAN

### 1. Membuat array (list biasa)

```
arr = [1, 2, 3, 4, 5]
```

Kamu membuat list berisi 5 angka: [1, 2, 3, 4, 5]

### 2. Mendeklarasikan nilai awal

```
n = len(arr) # n akan berisi 5 (panjang list) i = 0
```

# i adalah indeks awal

Variabel:

- n menyimpan panjang list (jumlah elemen)
- i adalah indeks yang akan dipakai untuk menelusuri list

```
print("Linear Traversal using while loop: ", end="")
```

Mencetak teks pembuka, tanpa pindah baris (karena end="").

### 3. Traversal menggunakan while loop

```
while i < n:
```

```
    print(arr[i], end="")
```

i+=1

Ini adalah **loop while**:

- Selama kurang dari (yaitu 5), program akan:
  - Cetak elemen arr[i]
  - Tambahkan satu persatu

Urutan yang terjadi:

i=0 → arr[0]=1

i=1 → arr[1]=2

i=2 → arr[2]=3

i=3 → arr[3]=4 i=

4 → arr[4]=5

Setelah i=5, kondisi i < n menjadi salah, maka loop berhenti.

print()

Untuk **pindah ke baris baru** setelah traversal selesai.

### **OUTPUT PROGRAM:**

Linear Traversal using while loop: 12345

### **PERBEDA DENGAN FOR LOOP**

#### **for loop**

Lebih ringkas

Cocok saat tahu jumlah pengulangan

#### **while loop**

Butuh inisialisasi dan peningkatan

Cocok saat butuh kontrol lebih fleksibel

### **7. PERAKTEKKE7**

```

1 # membuat array
2 arr = [1, 2, 3, 4, 5]
3
4 # mendeklarasikan nilai awal
5 start = 0
6 end = len(arr) - 1
7
8 print("Reverse Traversal using while loop: ", end=" ")
9 # Reverse Traversal dengan while
10 while start < end:
11     arr[start], arr[end] = arr[end], arr[start]
12     start += 1
13     end -= 1
14
15 print(arr)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAM  
n/Python313/python.exe "c:/Users/user/OneDrive/Dokumen/modul 04/STRUKTUR  
Reverse Traversal using while loop: [5, 4, 3, 2, 1]  
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAM

## KODE PROGRAM DAN PENJELASAN

### 1. Membuat array

arr = [1, 2, 3, 4, 5]

Kamu membuat list biasa Python dengan elemen [1, 2, 3, 4, 5].

### 2. Mendeklarasikan nilai awal

start = 0

end = len(arr) - 1

Kamu menyepak dua indeks:

- start = 0 → indeks pertama (elemen paling kiri)
- end = 4 (karena panjang list = 5) → indeks terakhir (elemen paling kanan)

print("Reverse Traversal using while loop: ", end="")

Mencetak teks pembuka, tanpa pindah baris (karena end="").

### 3. Reverse traversal menggunakan while loop

while start < end:

arr[start], arr[end] = arr[end], arr[start]

start += 1

end -= 1

**Penjelasanlogika:**

- Selamastart<end,kamutukarposisielimenkiridankanan
- Lalu,startmaju kekanandanendmundur ke kiri
- Inidisebutin-**placereverse**(membaliktanpamembuatlistbaru)

Langkah-langkahnya:

- Pertama:tukararr[0]danarr[4]→jadi[5,2,3,4,1]
- Kedua:tukararr[1]danarr[3]→jadi[5,4,3,2,1]
- Ketiga:start=2,end=2→kondisistart<endsalah→loopberhenti

print(arr)

Cetaklisthasilakhirsetelahdibalik:[5,4,3,2,1]

**OUTPUTPROGRAM:**

ReverseTraversalusingwhileloop:[5,4,3,2,1]

**INTILOGIKA:**

Kamutidak hanyamenelusurimundur,tapijugamembalikurutanelemenlistdengan cara:

- Menukarelemendariujungkiridanujungkanan
- Terusbergerak ketengah

## PERAKTEKKE8

```
1 # membuat array
2 arr = [12, 16, 20, 40, 50, 70]
3
4 # cetak arr sebelum penyisipan
5 print("Array Sebelum Insertion : ", arr)
6
7 # cetak panjang array sebelum penyisipan
8 print("Panjang Array : ", len(arr))
9
10 # menyisipkan array di akhir elemen menggunakan .append()
11 arr.append(26)
12
13 # cetak arr setelah penyisipan
14 print("Array Setelah Insertion : ", arr)
15
16 # cetak panjang array setelah penyisipan
17 print("Panjang Array : ", len(arr))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA I  
n\Python311\python.exe "c:/Users/user/OneDrive/Dokumen/modul 04/STRUK  
Array Sebelum Insertion : [12, 16, 20, 40, 50, 70]  
Panjang Array : 6  
Array Setelah Insertion : [12, 16, 20, 40, 50, 70, 26]  
Panjang Array : 7

### KODEDANPENJELASAN:

1. Membuatarray(list)

```
arr=[12,16,20,40,50,70]
```

Kamumembuatlistarrberisi6elemenangka.

2. Cetak array sebelum penyisipan

```
print("ArraySebelumInsertion:",arr)
```

Mencetakisilistsebelumelemenbaruditambahkan.

#### Outputsementara:

```
ArraySebelumInsertion:[12,16,20,40,50,70]
```

3. Cetakpanjangarraysebelumpenyisipan

```
print("Panjang Array : ", len(arr))
```

Menampilkantumlahelemendidalamlistsebelumditambahapapun.

#### Outputsementara:

```
PanjangArray:6
```

4. Menyisipkanelemendiakhirmenggunakan.append()



```
arr.append(26)
```

Fungsi.append() digunakan untuk **menambahkan 1 elemendibagianakhirlist**. Setelah

baris ini, arr akan menjadi:

```
[12,16,20,40,50,70,26]
```

5. Cetak array setelah penyisipan

```
print("ArraySetelahInsertion:",arr)
```

Menampilkanisilistsetelahelemenbaru(26)ditambahkankeakhir.

**Output:**

```
ArraySetelahInsertion:[12,16,20,40,50,70,26]
```

6. Cetakpanjangarraysetelahpenyisipan

```
print("Panjang Array : ", len(arr))
```

Menampilkanjumlahelemensetelahpenambahan.

**Output:**

```
PanjangArray:7
```

#### **INTISARI:**

##### **Fungsi/Perintah**

##### **Penjelasan**

arr.append(x)

Menambahkanelemenxke**akhirlist**

len(arr)

Mengembalikanjumlahtotalelemendidalamlist

Cetaksebelum/sesudah

Bergunauntukmelihatperubahanlistkarenaoperasitertentu

#### **PERAKTEKKE9**

```

1 # membuat array
2 arr = [12, 16, 20, 40, 50, 70]
3
4 # cetak arr sebelum penyisipan
5 print("Array Sebelum Insertion : ", arr)
6
7 # cetak panjang array sebelum penyisipan
8 print("Panjang Array : ", len(arr))
9
10 # menyisipkan array pada tengah elemen menggunakan .insert(pos, x)
11 arr.insert(4, 5)
12
13 # cetak arr setelah penyisipan
14 print("Array Setelah Insertion : ", arr)
15
16 # cetak panjang array setelah penyisipan
17 print("Panjang Array : ", len(arr))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\user\OneDrive\Documents\modul 04\STRUKTUR DATA - RINA NOLA UTAMI\Modu
n\Python313\python.exe "C:\Users\user\OneDrive\Documents\modul 04\STRUKTUR DATA
Array Sebelum Insertion : [12, 16, 20, 40, 50, 70]
Panjang Array : 6
Array Setelah Insertion : [12, 16, 20, 40, 5, 50, 70]
Panjang Array : 7
PS C:\Users\user\OneDrive\Documents\modul 04\STRUKTUR DATA - RINA NOLA UTAMI\Modu

```

## KODEDANPENJELASAN:

### 1. Membuatarray(list)

arr=[12,16,20,40,50,70]

Membuatlistarrdengan6elemenawal.

### 2. Cetak array sebelum penyisipan

print("ArraySebelumInsertion:",arr)

Menampilkanisilistsebelumperubahan.

#### Output:

ArraySebelumInsertion:[12,16,20,40,50,70]

### 3. Cetakpanjangarraysebelumpenyisipan

print("Panjang Array : ", len(arr))

Menampilkanpanjanglistsebelumdisisipkanelemenbaru.

#### Output:

PanjangArray:6

### 4. Menyisipkanelemen5padaindeks4menggunakan.insert()

arr.insert(4,5)

.insert(pos,x)menyisipkanelemenxpadaindekspos(posisike-4dalamlist).

- Indekske-4saatiniadalahelemen50
- Elemenbaru5akandisisipkandiposisiini

- Elemendiposisi4dancesudahnyabergeserkekanan

Setelah ini, arr jadi:

[12,16,20,40,5,50,70]

5. Cetak array setelah penyisipan

```
print("ArraySetelahInsertion:",arr)
```

Menampilkanlistsetelahelemenbarudisisipkan.

**Output:**

ArraySetelahInsertion:[12,16,20,40,5,50,70]

6. Cetakpanjangarraysetelahpenyisipan

```
print("Panjang Array : ", len(arr))
```

Menampilkanpanjanglistsetelahpenambahan.

**Output:**

PanjangArray:7

**INTISARI:**

**Fungsi/Perintah**

```
.insert(pos,x)
```

Indekslistdimulaidari0

Elemensetelahposisiposakanbergeserkekanan secara otomatis

**Penjelasan**

Menyisipkanelemenxdiindeks pos

Posisike-4artinyaelemenke-5 dalam list

Berikutadalahpenjelasan**barisperbaris**darikodePythonyangAndaberikan:

**Baris**

```
arr= [1,2, 3,4,5]
```

- Membuatsebuah**array/list**bernamaarrdenganelemen:1,2,3,4,5.

### Baris

start=0

- Menginisialisasi variabel start sebagai indeks awal dari list, yaitu indeks pertama (0).

### Baris

end=len(arr)-1

- Menginisialisasi variabel end sebagai indeks akhir dari list.
- len(arr) adalah panjang list (yaitu 5), sehingga end = 5 - 1 = 4 (indeks terakhir dari array).

### Baris

print("Reverse Traversal using while loop:", end="")

- Mencetak teks "Reverse Traversal using while loop:" tanpa pindah baris (end=" " berarti cetak spasi, bukan newline).
- Ini hanya untuk memberitahu bahwa proses berikutnya adalah traversal terbalik.

### Baris

while start < end:

arr[start], arr[end] = arr[end], arr[start]

start += 1

end -= 1

### Baris

- while start < end: adalah kondisi perulangan. Loop akan berjalan selama indeks start masih **lebih kecil** dari end.

### Baris

arr[start], arr[end] = arr[end], arr[start]

- Menukar elemen pada posisi start dengan end. Ini adalah cara membalik urutan elemen array **secara in-place** (langsung di dalam array, tanpa membuat array baru).

### Baris

start+=1

- Menaikkan nilai start agar mendekat ke tengah dari array.

### Baris

end-= 1

- Menurunkan nilai end agar juga mendekat ke tengah.

Loop ini akan terus berjalan dan menukar elemen dari luar ke dalam hingga start tidak lagi kurang dari end.

### Baris

print(arr)

- Setelah loop selesai (array sudah dibalik), baris ini mencetak isi array yang baru.

### Output

Reverse Traversal using while loop: [5,4,3,2,1]

- Elemen array telah **dibalik** dari [1,2,3,4,5] menjadi [5,4,3,2,1].

### PERAKTEKKE10

```
1 # membuat array
2 arr = [12, 16, 20, 40, 50, 70]
3
4 # cetak arr sebelum penyisipan
5 print("Array Sebelum Insertion : ", arr)
6
7 # cetak panjang array sebelum penyisipan
8 print("Panjang Array : ", len(arr))
9
10 # menyisipkan array pada tengah elemen menggunakan .insert(pos, x)
11 arr.insert(4, 5)
12
13 # cetak arr setelah penyisipan
14 print("Array Setelah Insertion : ", arr)
15
16 # cetak panjang array setelah penyisipan
17 print("Panjang Array : ", len(arr))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PLOTS

```
PS C:\Users\rislan\OneDrive\Documents> cd C:\Users\rislan\AppData\Local\Microsoft\Windows\OneDrive\Documents
PS C:\Users\rislan\AppData\Local\Microsoft\Windows\OneDrive\Documents> python3 /mnt/c:/Users/rislan/AppData/Local/Microsoft/Windows/OneDrive/
Array Sebelum Insertion : [12, 16, 20, 40, 50, 70]
Panjang Array : 6
Array Setelah Insertion : [12, 16, 20, 40, 5, 50, 70]
Panjang Array : 7
```

membuat array

Komentar ini menunjukkan bahwa baris berikut akan membuat array (dalam Python disebut list).

arr=[12,16,20,40,50,70]

Membuat sebuah list bernama `arr` yang berisi 6 elemen:

```
[12,16,20,40,50,70]
```

cetak `arr` sebelum penyisipan

Komentar bahwa baris berikut akan mencetak isi `arr` sebelum dilakukan penyisipan.

```
print("Array Sebelum Insertion:", arr)
```

Menampilkan isi `arr` sebelum ditambahkan elemen:

```
Array Sebelum Insertion: [12, 16, 20, 40, 50, 70] cetak
```

panjang array sebelum penyisipan

Komentar ini menjelaskan bahwa kita akan mencetak jumlah elemen dalam array sebelum penambahan.

```
print("Panjang Array:", len(arr))
```

Menggunakan fungsi `len()` untuk menghitung jumlah elemen dalam array. Hasilnya adalah 6:

```
Panjang Array: 6
```

menyisipkan array pada tengah elemen menggunakan `insert(pos, x)`

Komentar yang menjelaskan bahwa akan dilakukan penyisipan elemen di posisi tertentu menggunakan `.insert(posisi, nilai)`.

```
arr.insert(4, 5)
```

Baris ini menyisipkan angka 5 ke dalam array pada indeks ke-4 (ingat: indeks dimulai dari 0).

Sebelum penyisipan:

```
Index: 0 1 2 3 4 5
```

```
Value: 12 16 20 40 50 70
```

Setelah `insert(4, 5)` dijalankan, angka 5 akan masuk di posisi ke-4 (sebelum angka 50), menjadi:

[12,16,20,40,5,50,70]

cetak arr setelah penyisipan

Komentar bahwa baris berikutnya akan mencetak isi array setelah penyisipan. `print("Array Setelah Insertion : ", arr)`

Mencetak array setelah elemen 5 disisipkan:

Array Setelah Insertion: [12,16,20,40,5,50,70] cetak

panjang array setelah penyisipan

Komentar bahwa kita akan menghitung ulang jumlah elemen setelah penyisipan.

`print("Panjang Array : ", len(arr))`

Mencetak panjang array setelah penambahan elemen. Karena ada satu elemen tambahan, hasilnya sekarang:

Panjang Array: 7

Kesimpulan:

- `insert(posisi, nilai)` menyisipkan elemen pada posisi tertentu tanpa menghapus elemen lain.
- Elemen-elemen setelah posisi itu akan bergeser ke kanan

`len()` digunakan untuk melihat jumlah elemen sebelum dan sesudah perubahan.

- Kalau kamu ingin, aku juga bisa tunjukkan cara menghapus elemen dari list setelah penyisipan.

## 11. PERAKTEKKE11

```
1 # membuat array
2 a = [10, 20, 30, 40, 50]
3 print("Array Sebelum Deletion : ", a)
4
5 # menghapus elemen array pertama yang nilainya 30
6 a.remove(30)
7 print("Setelah remove(30):", a)
8
9 # menghapus elemen array pada index 1 (20)
10 popped_val = a.pop(1)
11 print("Popped element:", popped_val)
12 print("Setelah pop(1):", a)
13
14 # Menghapus elemen pertama (10)
15 del a[0]
16 print("Setelah del a[0]:", a)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\HP\OneDrive\dyni python\modul 2> & C:/Users/HP/AppData/Local/Programs/Python/Python38-64/Python.exe -py  
Array Sebelum Deletion : [10, 20, 30, 40, 50]  
Setelah remove(30): [10, 20, 40, 50]  
Popped element: 20  
Setelah pop(1): [10, 40, 50]  
Setelah del a[0]: [40, 50]  
PS C:\Users\HP\OneDrive\dyni python\modul 2>

Berikut adalah penjelasan baris per baris dari kode Python yang akan memberikan: membuat array

`a=[10,20,30,40,50]`

Artinya: Membuat sebuah list (array) bernama `a` yang berisi lima elemen: 10, 20, 30, 40, dan 50.

`print("Array Sebelum Deletion:", a)`

Artinya: Menampilkan isi list sebelum dilakukan penghapusan elemen.

- menghapus elemen array pertama yang nilainya 30

`a.remove(30)`

Artinya: Menghapus elemen pertama yang memiliki nilai 30 dari list. Jika ada lebih dari satu elemen dengan nilai 30, hanya yang pertama yang akan dihapus.

`print("Setelah remove(30):", a)`

Artinya: Menampilkan isi list setelah elemen bernilai 30 dihapus.

- menghapus elemen array pada index 1 (20)

`popped_val = a.pop(1)`



Artinya: Menghapus elemendi indekske-1 (elemen ke-2) dari list, yaitu 20, dan menyimpannya ke dalam variabel `popped_val`.

```
print("Popped element:", popped_val)
```

Artinya: Menampilkan elemen yang telah dihapus tadi (yaitu 20).

```
print("Setelah pop(1):", a)
```

Artinya: Menampilkan isi list setelah elemendi indekske-1 dihapus.

Menghapus elemen pertama (10)

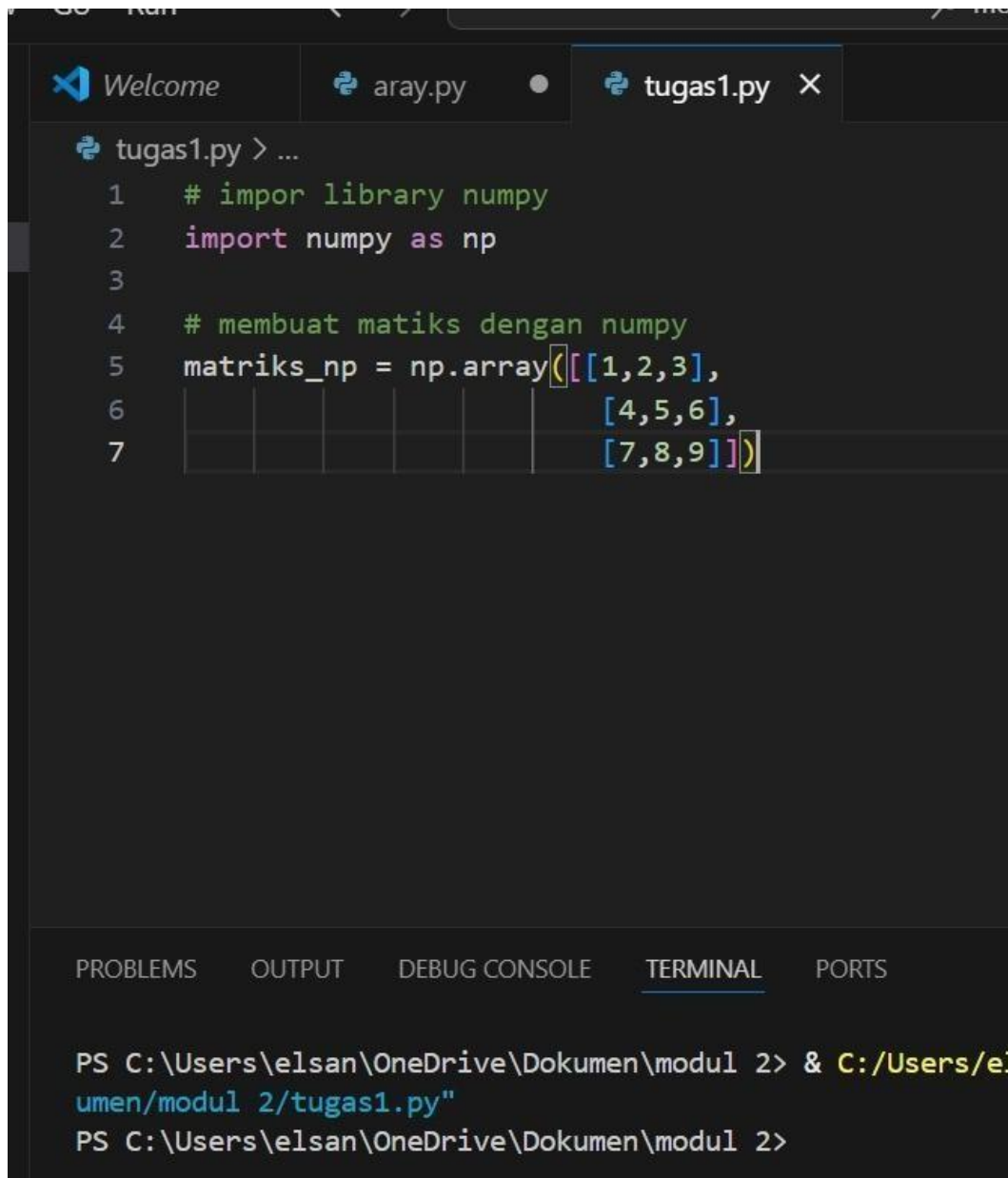
```
del a[0]
```

Artinya: Menghapus elemendi indekske-0 (elemen pertama) dari list, yaitu 10, menggunakan kata kunci `del`.

```
print("Setelah del a[0]:", a)
```

Artinya: Menampilkan isi list setelah elemen pertama dihapus.

## **PERAKTEKKE12**



The image shows a Visual Studio Code editor window with three tabs: 'Welcome', 'array.py', and 'tugas1.py'. The 'tugas1.py' tab is active, displaying a Python script. The script imports the numpy library and creates a 3x3 matrix. Below the editor, the 'TERMINAL' tab is active, showing the command to run the script and the resulting output.

```
tugas1.py > ...
1  # impor library numpy
2  import numpy as np
3
4  # membuat matiks dengan numpy
5  matriks_np = np.array([[1,2,3],
6                          [4,5,6],
7                          [7,8,9]])
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\elsan\OneDrive\Dokumen\modul 2> & C:/Users/elsan/OneDrive/Dokumen/modul 2/tugas1.py"
PS C:\Users\elsan\OneDrive\Dokumen\modul 2>
```

Berikut adalah penjelasan\*baris per baris\* dari kode Python yang kamu berikan:

## Baris 1:

python

# impor library numpy

◆ Ini adalah \*komentar\* (ditandai dengan #), artinya baris ini tidak akan dieksekusi.

◆ Tujuannya adalah memberikan penjelasan bahwa baris berikutnya akan melakukan `import library numpy`.

---

`#*Baris2:*`

python

```
import numpy as np
```

- Ini adalah baris yang `**mengimpor library numpy**` dan memberinya alias `np`.
- `numpy` adalah library Python yang digunakan untuk komputasi numerik, terutama untuk `*mengolah array atau matriks*`.
- Dengan menulis `as np`, kamu bisa menggunakan `np` sebagai singkatan dari `numpy`, sehingga lebih ringkas saat memanggil fungsinya.

`#*Baris4–7:*`

python

```
matriks_np = np.array([[1,2,3],  
                        [4,5,6],  
                        [7,8,9]])
```

Baris ini membuat sebuah `*array dua dimensi*` (atau bisa disebut matriks) menggunakan `numpy`.

`Fungsi np.array()` digunakan untuk mengubah list (daftar) biasa menjadi array `numpy`. Di dalam `np.array`, terdapat list 2 dimensi:

\* Barispertama:[1,2,3]

\* Bariskedua:[4,5,6]

\* Barisketiga:[7,8,9]

◆ Hasilnyaadalahmatriksberukuran\*3x3\*.

\*Kesimpulan:\*

Kodeinimembuatsebuah\*matriks3x3\*dengannumpy,isinya:

[[123]

[45 6]

[78 9]]

**PERAKTEKKE13**

```
4 # Membuat matriks dengan numpy
5 X = np.array([
6     [12,7,3],
7     [4,5,6],
8     [7,8,9]])
9
10 Y = np.array(
11     [[5,8,1],
12     [6,7,3],
13     [4,5,9]])
14
15 # Operasi penjumlahan dua matrik numpy
16 result = X + Y
17
18 # cetak hasil
19 print("Hasil Penjumlahan Matriks dari NumPy")
20 print(result)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAMI  
l/Programs/Python/Python313/python.exe "c:/Users/user/OneDrive/Dokumen/mod  
Modul\_2/Array/coba.py"  
Hasil Penjumlahan Matriks dari NumPy  
[[17 15 4]  
 [10 12 9]  
 [11 13 18]]  
PS C:\Users\user\OneDrive\Dokumen\modul 04\STRUKTUR DATA - RIMA NOVA UTAMI

Berikut adalah penjelasan baris per baris dari kode python tersebut

# Program penjumlahan matriks yang dibuat dari list

X= [[12,7,3],  
 [4,5,6],  
 [7,8,9]]

Y= [[5,8,1],  
 [6,7,3],  
 [4,5,9]]

result= [[0,0,0],  
 [0,0,0],  
 [0,0,0]]

# proses penjumlahan dua matriks menggunakan nested loop #  
mengulang sebanyak row (baris)

```

for i in range(len(X)):
    #mengulangsebanyakcolumn(kolom)
    for j in range(len(X[0])):
        result[i][j]=X[i][j]+Y[i][j]

print("HasilPenjumlahanMatriksdariLIST")

#cetakhasilpenjumlahansecaraiteratif for
r in result:
    print(r)

```

Berikut adalah penjelasan baris per baris dari kode Python untuk penjumlahan matriks yang dibuat dari list:

```

python
#Program penjumlahan matriks yang dibuat dari list

```

-Komentari yang menjelaskan tujuan program, yaitu menjumlahkan dua matriks yang direpresentasikan sebagai list di Python.

```

python
X= [[12,7,3],
    [4,5,6],
    [7,8,9]]

```

-Mendefinisikan matriks X sebagai list dua dimensi (list of lists) dengan 3 baris dan 3 kolom.

```

python
Y = [[5,8,1],
     [6,7,3],

```

[4,5,9]]

- Mendefinisikan matriks Y juga sebagai list dua dimensi dengan ukuran yang sama seperti X.

python

```
result=[[0,0,0],  
        [0,0,0],  
        [0,0,0]]
```

- Membuat matriks result dengan ukuran 3x3 yang diinisialisasi dengan nol sebagai tempat penyimpanan hasil penjumlahan.

python

```
# proses penjumlahan dua matriks menggunakan nested loop #  
mengulang sebanyak row (baris)  
for i in range(len(X)):
```

- Loop pertama (i) berjalan dari 0 sampai jumlah baris matriks X (3 baris). Ini mengontrol iterasi per baris.

python

```
    # mengulang sebanyak column (kolom)  
    for j in range(len(X[0])):
```

- Loop kedua (j) berjalan dari 0 sampai jumlah kolom matriks X (3 kolom). Ini mengontrol iterasi per kolom dalam setiap baris.

python

```
        result[i][j]=X[i][j]+Y[i][j]
```

- Menjumlahkan elemen pada posisi `[i][j]` dari matriks `X` dan `Y`, lalu menyimpan hasilnya di posisi yang sama pada matriks `result`.

python

```
print("Hasil Penjumlahan Matriks dari LIST")
```

- Mencetak teks sebagai judul hasil penjumlahan matriks.

python

```
# cetak hasil penjumlahan secara iteratif for
```

```
r in result:
```

```
    print(r)
```

- Loop untuk mencetak setiap baris dari matriks `result` satu persatu, sehingga hasil penjumlahan ditampilkan dalam format matriks.

# Ringkasan

Kode ini membuat dua matriks 3x3, menjumlahkan elemen-elemen yang bersesuaian dari kedua matriks tersebut menggunakan `nested loop`, menyimpan hasilnya di matriks baru, dan mencetak hasilnya baris per baris.

**PERAKTEK14**



```
tugas1.py > ...
1  # impor library numpy
2  import numpy as np
3
4  # Membuat matriks dengan numpy
5  X = np.array([
6      [12,7,3],
7      [4,5,6],
8      [7,8,9]])
9
10 Y = np.array(
11     [[5,8,1],
12      [6,7,3],
13      [4,5,9]])
14
15 # Operasi penjumlahan dua matrik numpy
16 result = X + Y
17
18 # cetak hasil
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORT

```
PS C:\Users\elsan\OneDrive\Dokumen\modul 2> & C:\Users\elsan\OneDrive\Dokumen\modul 2\tugas1.py
Hasil Penjumlahan Matriks dari NumPy
[[17 15  4]
 [10 12  9]
 [11 13 18]]
```

Berikut adalah penjelasan baris per baris dari kode Python tersebut: #

impor library numpy

import numpy as np

Penjelasan:

Mengimpor library NumPy dan memberinya alias np. NumPy adalah library Python yang digunakan untuk operasi matematika dan manipulasi array/matriks.

# Membuat matriks dengan numpy X

```
= np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])
```

Penjelasan:

Membuat array 2 dimensi (matriks) bernama X menggunakan fungsi `np.array`. Matriks X berisi:

1273

456

789

```
Y=np.array(  
    [[5,8,1],  
     [6,7,3],  
     [4,5,9]])
```

Penjelasan:

Membuat matriks kedua bernama Y, juga menggunakan `np.array`.

Matriks Y berisi:

581

673

459

```
#Operasi penjumlahan dua matriks numpy
```

```
result = X + Y
```

Penjelasan:

Melakukan operasi penjumlahan matriks antara X dan Y. NumPy secara otomatis menjumlahkan elemen yang berada di posisi yang sama.

Contohnya:

Baris 1 kolom 1:  $12+5=17$

Baris2kolom2:5+7=12

Baris3kolom3:9+9=18

Hasilnyadisimpandalamvariabelresult.

```
#cetakhasil
```

```
print("HasilPenjumlahanMatriksdariNumPy")
```

```
print(result)
```

Penjelasan:

Mencetakteksinformasi,lalumencetakisidarimatriksresult,yaituhasilpenjumlahan dari X dan Y.

Outputprogram:

HasilPenjumlahanMatriksdariNumPy

```
[[17 154]
```

```
[10129]
```

```
[111318]]
```

## PERAKTEKKE15

```
tugas1.py > ...
1  # impor library numpy
2  import numpy as np
3
4  # Membuat matriks dengan numpy
5  X = np.array([
6      [12,7,3],
7      [4,5,6],
8      [7,8,9]])
9
10 Y = np.array(
11     [[5,8,1],
12      [6,7,3],
13      [4,5,9]])
14
15 # Operasi pengurangan dua matrik numpy
16 result = X - Y
17
18 # cetak hasil
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\elsan\OneDrive\Dokumen\modul 2> & C:\Users\elsan\OneDrive\Dokumen\modul 2\tugas1.py
Hasil Pengurangan Matriks dari NumPy
[[ 7 -1  2]
 [-2 -2  3]
 [ 3  3  0]]
PS C:\Users\elsan\OneDrive\Dokumen\modul 2>
```

Berikut adalah penjelasan baris per baris dari kode Python tersebut yang menggunakan NumPy untuk melakukan pengurangan dua matriks:

#Baris1 python

#importlibrarynumpy

> Ini adalah komentar yang menjelaskan bahwa baris berikutnya akan mengimport library \*NumPy\*, sebuah library populer di Python untuk komputasi numerik, terutama operasi matriks dan array.

#Baris2

python

import numpy as np

> Mengimport library \*NumPy\* dan member alias np agar lebih ringkas saat digunakan dalam kode.

#Baris5–8

python

```
X=np.array([ [12,7,3],  
            [4,5,6],  
            [7,8,9]])
```

> Membuat \*matriks (array 2 dimensi)\* X menggunakan fungsi np.array. Matriks ini berukuran \*3x3\* dengan nilai-nilai sebagai berikut:

[12,7,3]

[ 4,5,6]

[ 7,8,9]

#Baris10–13

python

```
Y=np.array(  
    [[5,8,1],  
     [6,7,3],  
     [4,5,9]])
```

> Membuat \*matrikskedua\*Y, jugaberukuran3x3, dengannilai:

[5, 8,1]

[6, 7,3]

[4, 5,9]

#Baris15

python

```
result=X -Y
```

> Melakukan \*pengurangan elemen-elemen dari dua matriks\* (element-wise subtraction).Setiap elemen padaposisiyang samadiX danY akan dikurangkan:

- $12-5=7$
- $7-8=-1$
- $3-1=2$

- dan seterusnya...

Hasilnya adalah matriks result:

[7,-1,2]

[-2,-2,3]

[ 3,3,0]

#Baris18

python

```
print("Hasil Pengurangan Matriks dari NumPy")
```

> Menampilkan konteks judul agar hasil yang dicetak lebih mudah dipahami.

#Baris19

python

```
print(result)
```

> Menampilkan hasil pengurangan matriks yang telah disimpan dalam variabel result.

# Kesimpulan

Kode ini menunjukkan cara menggunakan NumPy untuk membuat dua matriks dan mengurangkannya secara langsung. Ini jauh lebih efisien daripada menggunakan nested loop seperti pada Python standar.

## PERAKTEKKE16

```
tugas1.py > ...
1  # impor library numpy
2  import numpy as np
3
4  # Membuat matriks dengan numpy
5  X = np.array([
6      [12,7,3],
7      [4,5,6],
8      [7,8,9]])
9
10 Y = np.array(
11     [[5,8,1],
12     [6,7,3],
13     [4,5,9]])
14
15 # Operasi perkalian dua matrik numpy
16 result = X * Y
17
18 # cetak hasil
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PC

```
PS C:\Users\elsan\OneDrive\Dokumen\modul 2> & C:\Python39\python.exe C:\Users\elsan\OneDrive\Dokumen\modul 2\tugas1.py
Hasil Perkalian Matriks dari NumPy
[[60 56  3]
 [24 35 18]
 [28 40 81]]
```

Berikut adalah penjelasan baris per baris dari kode Python yang menggunakan NumPy untuk melakukan perkalian dua matriks secara element-wise (per elemen):

#Baris1

python

#import library numpy



> Komentaryangmenjelaskanbahwabarislanjutnyaakanmengimporlibrary  
\*NumPy\*,yangdigunakanuntukoperasinumerikdiPython.

#Baris2

python

importnumpyasnp

> Mengimpor\*NumPy\*danmemberialiasnpagarlebihsingkatdipakai dalam kode.

#Baris5–8

python

```
X=np.array([ [12,7,3],  
             [4,5,6],  
             [7,8,9]])
```

> Membuat\*matriks(array2dimensi)\*Xmenggunakannp.array.Matriksini berukuran  
\*3x3\*dengan elemen:

[12,7,3]

[ 4,5,6]

[ 7,8,9]

#Baris10–13

python

```
Y=np.array(  
    [[5,8,1],  
     [6,7,3],  
     [4,5,9]])
```

> Membuat \*matriks kedua\* Y, juga berukuran \*3x3\*, dengan elemen:

```
[5, 8,1]  
[6, 7,3]  
[4, 5,9]
```

#Baris15

python

```
result=X* Y
```

> Melakukan \*perkalian elemen-per-elemen (element-wise multiplication)\* antara matriks X dan Y. Ini \*bukan perkalian matriks biasa (dot product)\*, tetapi setiap elemen dikalikan dengan elemen pada posisi yang sama:

\* 12 \* 5 = 60

\* 7 \* 8 = 56

\* 3 \* 1 = 3

\* dan seterusnya...

Hasilnya:

[60,56,3]

[24,35,18]

[28,40,81]

#Baris18

python

```
print("HasilPerkalianMatriksdariNumPy")
```

> Menampilkanteksjuduluntukmemberikankontekspadaoutput.

#Baris19

python

```
print(result)
```

> Menampilkanhasilperkalianelemen-per-elemendarimatriksXdanY.

# Kesimpulan:

Kodeinimemperlihatkan\*perkalianduatmatrikssecaraelement-wise\*menggunakan\* dalam NumPy. Jika kamu ingin melakukan \*perkalian matriks sesungguhnya (dot product)\*, kamu harus menggunakan:

python

```
result=np.dot(X,Y)
```

atau

python

```
result=X @Y
```

### PERAKTEKKE17

```
tugas1.py > ...
1  # Praktek 17 : Operasi Pembagian Matriks dengan numpy
2  # impor library numpy
3  import numpy as np
4
5  # Membuat matriks dengan numpy
6  X = np.array([
7      [12,7,3],
8      [4,5,6],
9      [7,8,9]])
10
11  Y = np.array(
12      [[5,8,1],
13       [6,7,3],
14       [4,5,9]])
15
16  # Operasi pembagian dua matrik numpy
17  result = X / Y
18
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\elsan\OneDrive\Dokumen\modul 2> & C:/Users/elsan/AppData/Local/Programs/Python/Python39-64/Python.exe C:\Users\elsan\OneDrive\Dokumen\modul 2\tugas1.py
Hasil Pembagian Matriks dari NumPy
[[2.4      0.875    3.      ]
 [0.66666667 0.71428571 2.      ]
 [1.75     1.6      1.      ]]
```

Berikut penjelasan baris per baris dari kode Python yang kam berikan:

#Praktek17:OperasiPembagianMatriksdengannumpy

Komentarinimemberikaninformasibahwainiadalahpraktikke-17danberisicontoh operasi pembagian matriks menggunakan library NumPy.

#importlibrarynumpy

KomentaryangmenjelaskanbahwakitaakanmengimportlibraryNumPy.

#importnumpyasnp

BarisinimengimportlibraryNumPydanmemberialiasnp,sehinggakitabisamenggunakan np untuk memanggil fungsi-fungsi dalam NumPy.

python

```
X=np.array([
    [12, 7, 3],
    [4,5,6],
    [7,8,9]
])
```

Barisinimembuatmatriks3x3bernamaXdarilistPythonmenggunakanfungsi np.array().  
Matriks X:

1273

456

789

```
python
Y=np.array([
    [5, 8, 1],
    [6,7,3],
    [4,5,9]
])
```

Membuat matriks 3x3 bernama Y yang juga berasal dari list Python. Matriks Y:

```
581
673
459
```

```
#Operasi pembagian dua matrik numpy
```

Komentar ini menjelaskan bahwa operasi selanjutnya adalah pembagian dua matriks.

```
result=X/Y
```

Baris ini melakukan pembagian elemen per elemen (element-wise division) antara matriks X dan Y. Artinya:

```
python
result[i][j]=X[i][j]/Y[i][j]
```

Contoh:

```
*result[0][0]=12/5=2.4  
*result[0][1]=7/8=0.875  
*danseterusnya...
```

```
#cetakhasil
```

Komentarbahwabarisberikutakanmencetakhasilkelayar.

```
python
```

```
print("HasilPembagianMatriksdariNumPy") print(result)
```

```
* print("HasilPembagianMatriksdariNumPy")mencetakjuduloutput.  
* print(result)mencetakhasildaripembagianmatriksXdanYdalambentukmatriks 3x3.
```

```
#ContohOutput:
```

Jikadijalankan,akanmunculhasilsepertiini(dibulatkanuntuktampilan):

```
HasilPembagianMatriksdariNumPy
```

```
[[2.4    0.875   3.    ]  
 [0.66666667 0.71428571 12.   ]  
 [1.75    1.6     1.    ]]
```

## PERAKTEK18

```
tugas1.py > ...
1  # impor library numpy
2  import numpy as np
3
4  # membuat matriks
5  matriks_a = np.array([
6      [1, 2, 3],
7      [4, 5, 6],
8      [7, 8, 9]
9  ])
10
11 # cetak matriks
12 print("Matriks Sebelum Transpose")
13 print(matriks_a)
14
15 # transpose matriks_a
16 balik = matriks_a.transpose()
17
18 # cetak matriks setelah dihalik
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
umen/modul 2/tugas1.py"
Matriks Sebelum Transpose
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Matriks Setelah Transpose
[[1 4 7]
 [2 5 8]
 [3 6 9]]
PS C:\Users\elsan\OneDrive\Dokumen\modul 2>
```

Berikut adalah penjelasan baris per baris dari kode Python yang menggunakan NumPy untuk melakukan transpose (permutasi baris dan kolom) pada matriks:

#Baris1

python

#import library numpy



> KomentaryangmenjelaskanbahwakodeakanmenggunakanlibraryNumPy.#Baris2

python

importnumpyasnp

> MengimporlibraryNumPydanmemberialiasnpuntukmempersingkatpenulisan  
fungsi-fungsinya.

#Baris5–9

python

```
matriks_a=np.array([ [1,  
    2, 3],  
    [4,5,6],  
    [7,8,9]  
])
```

> Membuatmatriks2dimensimatriks\_amenggunakannp.array.Matriksinimiliki  
ukuran 3x3, dengan elemen:

[1, 2,3]

[4, 5,6]

[7, 8,9]

#Baris12

```
python
```

```
print("MatriksSebelumTranspose")
```

> Menampilkanteksuntukmemberitahubahwaoutputberikutadalahmatrikssebelum dilakukan operasi transpose.

```
#Baris13
```

```
python
```

```
print(matriks_a)
```

> Menampilkansidarimatriks\_a.

```
#Baris16
```

```
python
```

```
balik=matriks_a.transpose()
```

> Melakukantranspose,yaitu\*\*menukarbarismenjadikolomdankolommenjadibaris.

> Hasiltransposedarimatriks\_aadalah:

```
[1, 4,7]
```

```
[2, 5,8]
```

```
[3, 6,9]
```

Matriks ini disimpan dalam variabel balik.

Alternatif penulisan transpose:

```
python
```

```
balik=matriks_a.T
```

```
#Baris19
```

```
python
```

```
print("Matriks Setelah Transpose")
```

> Menampilkan teks penjelasan bahwa output berikut adalah matriks hasil transpose.

```
#Baris20
```

```
python
```

```
print(balik)
```

> Menampilkan hasil dari operasi transpose yang sudah disimpan dalam variabel balik.

```
# Kesimpulan:
```

Kode ini memperlihatkan bagaimana menggunakan NumPy untuk:

- \* Membuat matriks 2 dimensi

- \* Melihat matriks sebelum dan sesudah di-transpose

Transposesangatpentingdalamaljabarlinear,sepertidalamoperasidotproduct, rotasi, atau manipulasi data tabular.