

Modul 2 Array

❖ Praktek 1



```
modul-2.py > ...
1 # impor library numpy
2 import numpy as np
3
4 # membuat array dengan numpy
5 tinggi_badan = np.array([150, 165, 172, 160])
6
7 # akses data pada array
8 print(tinggi_badan[2])
```

Penjelasan:

◆ Baris ke-5

Membuat sebuah array menggunakan fungsi `np.array()` dari library `numpy`.

Nilai yang diberikan ke array adalah `[150, 165, 172, 160]`, yang bisa diartikan sebagai tinggi badan dari empat siswa.

Hasil dari `np.array(...)` disimpan dalam variabel `tinggi_badan`.

◆ Baris ke-8

mencetak (menampilkan di layar) elemen ke-4 dari array `nilai_siswa`.

Perlu diingat bahwa indeks dalam array dimulai dari 0, jadi:

- `tinggi_badan[0]` → 150
- `tinggi_badan[1]` → 165
- `tinggi_badan[2]` → 172
- `tinggi_badan[3]` → 160

Jadi, `tinggi_badan[2]` mengambil nilai 172, dan itulah yang akan ditampilkan di output.

Output:

172

❖ Praktek 2

```
modul-2.py > ...
12 # import library numpy
13 import numpy as np
14
15 # membuat array dengan numpy
16 tinggi_siswa_1 = np.array([160, 155, 150, 165])
17 tinggi_siswa_2 = np.array([[158, 162, 149], [151, 150, 170]])
18
19 # cara akses elemen array
20 print(tinggi_siswa_1[0])
21 print(tinggi_siswa_2[1][1])
22
23 # mengubah nilai elemen array
24 tinggi_siswa_1[0] = 168
25 tinggi_siswa_2[1][1] = 155
26
27 # cek perubahannya dengan akses elemen array
28 print(tinggi_siswa_1[0])
29 print(tinggi_siswa_2[1][1])
30
31 # cek ukuran dan dimensi array
32 print("Ukuran Array : ", tinggi_siswa_1.shape)
33 print("Ukuran Array : ", tinggi_siswa_2.shape)
34 print("Dimensi Array : ", tinggi_siswa_2.ndim)
```

Penjelasan:

◆ Baris ke-16

Membuat array 1 dimensi (vektor) berisi data tinggi badan siswa, kemudian disimpan ke variabel `tinggi_siswa_1`.

Isi array: `[160, 155, 150, 165]` — artinya terdapat 4 siswa.

◆ Baris ke -17

Membuat array 2 dimensi (matriks) berisi tinggi badan siswa dari dua kelompok. Strukturnya `[[158, 162, 149],[151, 150, 170]]`, Ini seperti 2 baris (kelompok siswa), masing-masing berisi 3 nilai.

◆ Baris ke-20

Mengakses dan mencetak elemen pertama dari array `tinggi_siswa_1`.

Karena indeks dimulai dari 0, maka yang dicetak adalah 160.

◆ Baris ke-21

Mengakses elemen baris ke-2 dan kolom ke-2 dari `tinggi_siswa_2`.

Ingat bahwa indeks dimulai dari 0, jadi:

- `tinggi_siswa_2[1] = [151, 150, 170]`
- `tinggi_siswa_2[1][1] = 150`

Output: 150

◆ Baris ke-24

Mengubah elemen pertama (indeks 0) dari `tinggi_siswa_1` menjadi 168.

Sebelumnya: `tinggi_siswa_1[0] = 160`

Setelah: `tinggi_siswa_1[0] = 168`

◆ Baris ke-25

Mengubah elemen pada baris ke-2, kolom ke-2 dari tinggi_siswa_2.

Sebelumnya: tinggi_siswa_2[1][1] = 150

Setelah: tinggi_siswa_2[1][1] = 155

◆ Baris ke-28

Mencetak kembali elemen pertama dari tinggi_siswa_1 setelah diubah.

Output: 168

◆ Baris ke-29

Mencetak kembali elemen baris ke-2 kolom ke-2 dari tinggi_siswa_2 setelah diubah.

Output: 155

◆ Baris ke-32

Fungsi .shape digunakan untuk mengetahui ukuran/tata letak array.

Untuk tinggi_siswa_1 (array 1 dimensi dengan 4 elemen), outputnya:

Ukuran Array : (4,)

◆ Baris ke-33

Untuk tinggi_siswa_2 yang merupakan array 2 dimensi (2 baris, 3 kolom),

hasilnya: Ukuran Array : (2, 3)

◆ Baris ke-34

Fungsi .ndim digunakan untuk mengecek jumlah dimensi array.

Karena tinggi_siswa_2 adalah array 2 dimensi, maka hasilnya:

Dimensi Array : 2

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
NYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List/modul-2.
172
168
150
168
155
Ukuran Array : (4,)
Ukuran Array : (2, 3)
Dimensi Array : 2
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 3

```
Welcome | contoh.py | modul-2.py X
modul-2.py > ...
38 # impor library numpy
39 import numpy as np
40
41 # membuat array
42 a = np.array([150, 155, 160])
43 b = np.array([145, 150, 155])
44
45 # menggunakan operasi penjumlahan pada 2 array
46 print(a + b)          # array([295, 305, 315])
47
48 # Indexing dan Slicing pada Array
49 arr = np.array([150, 155, 160, 165])
50 print(arr[1:3])       # array([155, 160])
51
52 # iterasi pada array
53 for x in arr:
54     print(x)
```

Penjelasan:

◆ Baris ke-42

Membuat array a dengan elemen [150, 155, 160].

Array ini bisa mewakili data, misalnya tinggi badan siswa.

◆ Baris ke-43

Membuat array b dengan elemen [145, 150, 155].

Sama seperti a, ini juga array 1 dimensi dengan 3 elemen.

◆ Baris ke-46

Menjumlahkan array a dan b elemen per elemen:

- $150 + 145 = 295$
- $155 + 150 = 305$
- $160 + 155 = 315$

Hasil: array([295, 305, 315])

◆ Baris ke-49

Membuat array arr dengan 4 elemen.

Contohnya bisa mewakili tinggi 4 siswa.

◆ Baris ke-50

Melakukan slicing terhadap array arr dari indeks 1 hingga sebelum indeks 3:

- $arr[1] = 155$
- $arr[2] = 160$

Hasil: array([155, 160])

◆ Baris ke-(53-54)

- `for x in arr:` adalah perulangan untuk setiap elemen dalam array arr.
- `print(x)` akan mencetak setiap elemen satu per satu.

Output:

```
[295 305 315]
[155 160]
150
155
160
165
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array
```

❖ Praktek 4



```
Welcome | contoh.py | modul-2.py X
modul-2.py > ...
58 # membuat array
59 arr = [150, 155, 160, 165, 170]
60
61 # Linear Traversal ke tiap elemen arr
62 print("Linear Traversal: ", end=" ")
63 for i in arr:
64     print(i, end=" ")
65 print()
```

Penjelasan:

◆ Baris ke-59

Membuat list (array dalam istilah Python biasa) berisi lima angka: [150, 155, 160, 165, 170].

Contoh konteks: angka-angka ini bisa mewakili data tinggi badan siswa.

◆ Baris ke-62

Mencetak teks "Linear Traversal: " ke layar tanpa berpindah baris (karena end=" "). Tujuannya agar angka-angka hasil traversal bisa tampil di baris yang sama.

◆ Baris ke-(63-64)

- for i in arr: adalah loop (perulangan) untuk mengakses setiap elemen i dalam list arr.
- print(i, end=" ") mencetak elemen tersebut di baris yang sama, dipisahkan dengan spasi.

Output-nya akan menjadi: Linear Traversal: 150 155 160 165 170

◆ Baris ke-65

Baris kosong digunakan untuk pindah baris setelah traversal selesai.

Tanpa ini, baris selanjutnya dalam program (jika ada) akan tercetak di baris yang sama.

Output:

Linear Traversal: 150 155 160 165 170

❖ Praktek 5



```
modul-2.py > ...
69 # membuat array
70 arr = [150, 155, 160, 165, 170]
71
72 # Reverse Traversal dari elemen akhir
73 print("Reverse Traversal: ", end=" ")
74 for i in range(len(arr) - 1, -1, -1):
75     print(arr[i], end=" ")
76 print()
```

Penjelasan:

◆ Baris ke-70

Membuat list Python bernama arr yang berisi lima angka.

Contohnya bisa mewakili tinggi badan dalam cm dari 5 siswa.

◆ Baris ke-73

Mencetak teks "Reverse Traversal: " tanpa pindah baris (karena end=" "), agar angka-angka yang akan ditampilkan berada di baris yang sama.

◆ Baris ke-74

- range(start, stop, step) digunakan untuk menghasilkan urutan angka.
- len(arr) - 1 = 4, yaitu indeks terakhir dari list arr.
- -1 sebagai nilai berhenti berarti sampai sebelum indeks -1, jadi akan berhenti di 0.
- -1 sebagai langkah (step) menunjukkan arah mundur (reverse).
- Artinya: loop berjalan dari indeks 4 → 3 → 2 → 1 → 0.

◆ Baris ke-75

- Mencetak elemen array arr berdasarkan indeks i, dimulai dari belakang.
- end=" " digunakan agar angka-angka tercetak dalam satu baris dengan spasi antar elemen.

Output:

Reverse Traversal: 170 165 160 155 150

❖ Praktek 7



```
modul-2.py > ...
80 # membuat array
81 arr = [150, 155, 160, 165, 170]
82
83 # mendeklarasikan nilai awal
84 n = len(arr)
85 i = 0
86
87 print("Linear Traversal using while loop: ", end=" ")
88 # Linear Traversal dengan while
89 while i < n:
90     print(arr[i], end=" ")
91     i += 1
92 print()
```

Penjelasan:

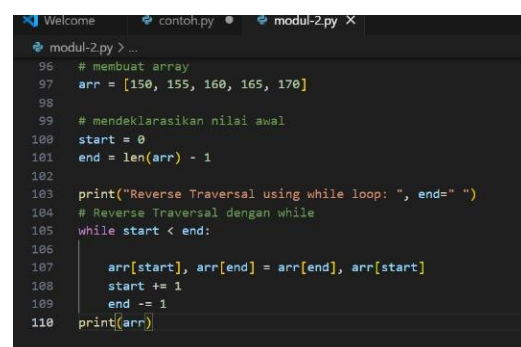
- ◆ Baris ke-81
 - Membuat list arr berisi lima angka.
 - Data ini bisa dianggap sebagai contoh data tinggi badan atau nilai siswa.
- ◆ Baris ke-84
 - n menyimpan jumlah elemen dalam array, yaitu 5.
 - Ini digunakan sebagai batas atas dalam perulangan.
- ◆ Baris ke-85

i adalah indeks awal yang digunakan untuk mulai menelusuri array dari posisi pertama (arr[0]).
- ◆ Baris ke-87
 - Mencetak teks deskriptif di awal output.
 - end=" " mencegah pindah baris, sehingga elemen array akan tercetak di baris yang sama.
- ◆ Baris ke-(89-91)
 - while i < n: adalah loop yang akan terus berjalan selama nilai i lebih kecil dari jumlah elemen (n).
 - print(arr[i], end=" ") mencetak elemen pada indeks i di array tanpa pindah baris.
 - i += 1 menaikkan indeks, agar traversal berjalan ke elemen berikutnya.

Output:

Linear Traversal using while loop: 150 155 160 165 170

❖ Praktek 8



```
96 # membuat array
97 arr = [150, 155, 160, 165, 170]
98
99 # mendeklarasikan nilai awal
100 start = 0
101 end = len(arr) - 1
102
103 print("Reverse Traversal using while loop: ", end=" ")
104 # Reverse Traversal dengan while
105 while start < end:
106
107     arr[start], arr[end] = arr[end], arr[start]
108     start += 1
109     end -= 1
110 print(arr)
```

Penjelasan:

- ◆ Baris ke-97

List arr dibuat dengan 5 elemen. Misalnya, ini bisa mewakili data tinggi badan siswa.

◆ Baris ke-(100-101)

- start adalah indeks awal dari array (0).
- end adalah indeks terakhir array ($\text{len}(\text{arr}) - 1 = 4$).
- Kedua variabel ini digunakan untuk menunjuk dua ujung array yang akan ditukar.

◆ Baris ke-103

Mencetak label deskriptif di awal output tanpa pindah baris (`end=" "`).

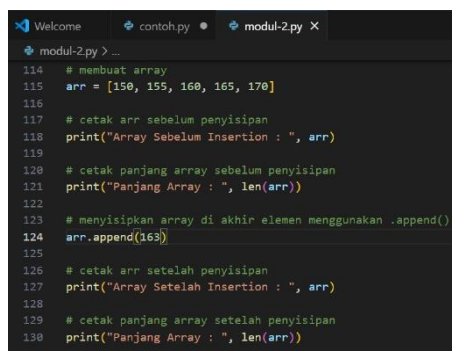
◆ Baris ke-(105-109)

- Kondisi loop: `while start < end` artinya perulangan terus dilakukan selama start belum melewati end.
- Pertukaran elemen:
`arr[start], arr[end] = arr[end], arr[start]` menukar posisi elemen paling kiri dengan paling kanan.
- Increment dan decrement:
 - ✓ `start += 1` memajukan pointer dari kiri ke kanan.
 - ✓ `end -= 1` memundurkan pointer dari kanan ke kiri.
- Proses ini membalik urutan elemen dalam array secara in-place (langsung di dalam array yang sama, tanpa membuat array baru).

Output:

Reverse Traversal using while loop: [170, 165, 160, 155, 150]

❖ Praktek 9



```
modul-2.py > ...
114 # membuat array
115 arr = [150, 155, 160, 165, 170]
116
117 # cetak arr sebelum penyisipan
118 print("Array Sebelum Insertion : ", arr)
119
120 # cetak panjang array sebelum penyisipan
121 print("Panjang Array : ", len(arr))
122
123 # menyisipkan array di akhir elemen menggunakan .append()
124 arr.append(163)
125
126 # cetak arr setelah penyisipan
127 print("Array Setelah Insertion : ", arr)
128
129 # cetak panjang array setelah penyisipan
130 print("Panjang Array : ", len(arr))
```

Penjelasan:

◆ Baris ke-115

Membuat list `arr` berisi lima elemen. Misalnya, ini bisa mewakili tinggi badan siswa.

◆ Baris ke-118

Menampilkan isi array sebelum ditambahkan elemen baru.

Output: Array Sebelum Insertion : [150, 155, 160, 165, 170]

◆ Baris ke-121

Menampilkan panjang array, yaitu jumlah elemen saat ini.

Output: Panjang Array : 5

◆ Baris ke-124

Menambahkan elemen 163 ke akhir list arr.

Setelah baris ini, isi list menjadi: [150, 155, 160, 165, 170, 163]

◆ Baris ke-127

Menampilkan isi array setelah penyisipan.

Output: Array Setelah Insertion : [150, 155, 160, 165, 170, 163]

◆ Baris ke-130

Menampilkan panjang array setelah penambahan elemen baru.

Output: Panjang Array : 6

Output:

```
Array Sebelum Insertion : [150, 155, 160, 165, 170]
Panjang Array : 5
Array Setelah Insertion : [150, 155, 160, 165, 170, 163]
Panjang Array : 6
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 10

```
Welcome  contoh.py  modul-2.py X
modul-2.py > ...
134 # membuat array
135 arr = [150, 155, 160, 165, 170]
136
137 # cetak arr sebelum penyisipan
138 print("Array Sebelum Insertion : ", arr)
139
140 # cetak panjang array sebelum penyisipan
141 print("Panjang Array : ", len(arr))
142
143 # menyisipkan array pada tengah elemen menggunakan .insert(pos, x)
144 arr.insert(3, 100)
145
146 # cetak arr setelah penyisipan
147 print("Array Setelah Insertion : ", arr)
148
149 # cetak panjang array setelah penyisipan
150 print("Panjang Array : ", len(arr))
```

Penjelasan:

◆ Baris ke-135

Membuat list arr berisi lima angka. Misalnya, ini bisa mewakili tinggi badan siswa dalam cm.

◆ Baris ke-138

Menampilkan isi array sebelum penambahan elemen.

Output: Array Sebelum Insertion : [150, 155, 160, 165, 170]

◆ Baris ke-141

Menampilkan jumlah elemen array sebelum penyisipan.

Output: Panjang Array : 5

◆ Baris ke-144

Menambahkan nilai 100 ke posisi indeks ke-3, yaitu di antara 160 dan 165.

Sebelum: [150, 155, 160, 165, 170]

Sesudah: [150, 155, 160, 100, 165, 170]

◆ Baris ke-147

Menampilkan array setelah elemen baru disisipkan.

Output: Array Setelah Insertion : [150, 155, 160, 100, 165, 170]

◆ Baris ke-150

Menampilkan jumlah elemen setelah penambahan (menjadi 6).

Output: Panjang Array : 6

Output:

```
Array Sebelum Insertion : [150, 155, 160, 165, 170]
Panjang Array : 5
Array Setelah Insertion : [150, 155, 160, 100, 165, 170]
Panjang Array : 6
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

Prakterk 11

```
modul-2.py > ...
154 # membuat array
155 a = [150, 155, 160, 165, 170]
156 print("Array Sebelum Deletion : ", a)
157
158 # menghapus elemen array pertama yang nilainya 30
159 a.remove(160)
160 print("Setelah remove(160):", a)
161
162 # menghapus elemen array pada index 1 (155)
163 popped_val = a.pop(1)
164 print("Popped element:", popped_val)
165 print("Setelah pop(1):", a)
166
167 # Menghapus elemen pertama (150)
168 del a[0]
169 print("Setelah del a[0]:", a)
```

Penjelasan:

◆ Baris ke-155

List a dibuat berisi lima elemen. Ini bisa merepresentasikan data seperti tinggi badan siswa.

◆ Baris ke-156

Menampilkan isi array sebelum ada elemen yang dihapus.

Output: Array Sebelum Deletion : [150, 155, 160, 165, 170]

◆ Baris ke-159

Metode `.remove()` akan menghapus elemen pertama yang bernilai 160 dari array.
Setelah ini, array menjadi: [150, 155, 165, 170]

◆ Baris ke-160

Menampilkan array setelah 160 dihapus.

Output: Setelah `remove(160)`: [150, 155, 165, 170]

◆ Baris ke-163

`pop(1)` akan menghapus dan mengembalikan elemen di indeks ke-1, yaitu 155.

Nilai yang dihapus disimpan dalam variabel `popped_val`.

◆ Baris ke-164

Menampilkan elemen yang dihapus (155).

Output: Popped element: 155

◆ Baris ke-165

Menampilkan array setelah `pop(1)` dilakukan.

Output: Setelah `pop(1)`: [150, 165, 170]

◆ Baris ke-168

Menghapus elemen pada indeks ke-0 menggunakan keyword `del`.

Array menjadi: [165, 170].

◆ Baris ke-169

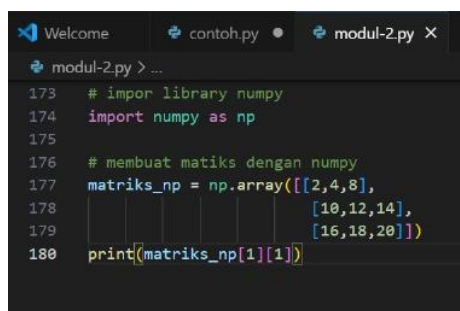
Menampilkan array setelah penghapusan dengan `del`.

Output: Setelah `del a[0]`: [165, 170]

Output:

```
Array Sebelum Deletion : [150, 155, 160, 165, 170]
Setelah remove(160): [150, 155, 165, 170]
Popped element: 155
Setelah pop(1): [150, 165, 170]
Setelah del a[0]: [165, 170]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 12



```
Welcome  contoh.py  modul-2.py x
modul-2.py > ...
173 # impor library numpy
174 import numpy as np
175
176 # membuat matiks dengan numpy
177 matriks_np = np.array([[2,4,8],
178                        [10,12,14],
179                        [16,18,20]])
180 print(matriks_np[1][1])
```

Penjelasan:

◆ Baris ke-(177-179)

Membuat array 2 dimensi (3x3) dengan NumPy.

Isi dari matriks_np adalah: $\begin{bmatrix} 2 & 4 & 8 \\ 10 & 12 & 14 \\ 16 & 18 & 20 \end{bmatrix}$

◆ Baris ke-180

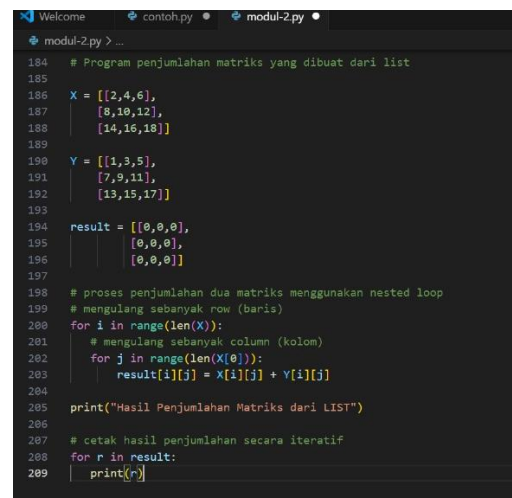
Mengakses elemen pada baris ke-1 dan kolom ke-1 (dalam indeks dimulai dari nol).

- `matriks_np[1]` mengakses baris kedua: $[10, 12, 14]$
- `matriks_np[1][1]` mengakses kolom kedua dari baris tersebut: 12

Output:

12

❖ Praktek 13



```
184 # Program penjumlahan matriks yang dibuat dari list
185
186 X = [[2,4,6],
187      [8,10,12],
188      [14,16,18]]
189
190 Y = [[1,3,5],
191      [7,9,11],
192      [13,15,17]]
193
194 result = [[0,0,0],
195           [0,0,0],
196           [0,0,0]]
197
198 # proses penjumlahan dua matriks menggunakan nested loop
199 # mengulang sebanyak row (baris)
200 for i in range(len(X)):
201     # mengulang sebanyak column (kolom)
202     for j in range(len(X[0])):
203         result[i][j] = X[i][j] + Y[i][j]
204
205 print("Hasil Penjumlahan Matriks dari LIST")
206
207 # cetak hasil penjumlahan secara iteratif
208 for r in result:
209     print(r)
```

Penjelasan:

◆ Baris ke-(186-192)

Matriks X dan Y dideklarasikan sebagai list 2 dimensi (3x3).

- X adalah matriks pertama
- Y adalah matriks kedua

Masing-masing elemen dari X dan Y akan dijumlahkan berdasarkan posisi indeks $[i][j]$.

◆ Baris ke-(194-196)

Matriks kosong result (3x3) dibuat sebagai tempat menyimpan hasil penjumlahan elemen-elemen dari X dan Y.

◆ Baris ke-200

Loop pertama untuk mengakses baris-baris dari matriks (i = 0 sampai 2).

◆ Baris ke-202

Loop kedua di dalam loop pertama untuk mengakses kolom-kolom dari setiap baris (j = 0 sampai 2).

◆ Baris ke-203

Menjumlahkan elemen dari X dan Y pada posisi [i][j] lalu menyimpan hasilnya di result[i][j].

Contoh:

- $\text{result}[0][0] = 2 + 1 = 3$
- $\text{result}[1][2] = 12 + 11 = 23$, dst.

◆ Baris ke-205

Menampilkan teks sebagai penanda bahwa hasil penjumlahan akan dicetak.

◆ Baris ke-(208-209)

Loop untuk mencetak setiap baris dari matriks result.
r merepresentasikan satu baris dalam result.

Output:

```
Hasil Penjumlahan Matriks dari LIST
[3, 7, 11]
[15, 19, 23]
[27, 31, 35]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 14

```
Welcome  contoh.py  modul-2.py
modul-2.py > ...
213 # impor library numpy
214 import numpy as np
215
216 # Membuat matriks dengan numpy
217 X = np.array([
218     [2,4,6],
219     [8,10,12],
220     [14,16,18]])
221
222 Y = np.array([
223     [1,3,5],
224     [7,9,11],
225     [13,15,17]])
226
227 # Operasi penjumlahan dua matrik numpy
228 result = X + Y
229
230 # cetak hasil
231 print("Hasil Penjumlahan Matriks dari NumPy")
232 print(result)
```

Penjelasan:

◆ Baris ke-(217-220)

Matriks X dibuat sebagai array 2 dimensi (3x3) menggunakan np.array.

Isi matriks X: $\begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix}$

◆ Baris ke-(222-225)

Matriks Y juga dibuat sebagai array 2 dimensi (3x3).

Isi matriks Y: $\begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \\ 13 & 15 & 17 \end{bmatrix}$

◆ Baris ke-228

Menjumlahkan matriks X dan Y secara elemen per elemen (element-wise).

Karena NumPy mendukung operasi vektor/matriks, proses ini langsung dijalankan tanpa perlu loop.

Contoh hasil penjumlahan:

- $X[0][0] + Y[0][0] = 2 + 1 = 3$
- $X[1][2] + Y[1][2] = 12 + 11 = 23$

◆ Baris ke-231

- Fungsi print() digunakan untuk mencetak output ke konsol.
- String "Hasil Penjumlahan Matriks dari NumPy" memberi informasi kepada pengguna bahwa output berikut adalah hasil dari penjumlahan dua matriks NumPy.

◆ Baris ke-232

Mencetak matriks hasil penjumlahan.

Output:

```
Hasil Penjumlahan Matriks dari NumPy
[[ 3  7 11]
 [15 19 23]
 [27 31 35]]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 15

```
Welcome | contoh.py | modul-2.py
modul-2.py > ...
236 # import library numpy
237 import numpy as np
238
239 # Membuat matriks dengan numpy
240 X = np.array([
241     [22,4,6],
242     [18,10,12],
243     [14,36,18]])
244
245 Y = np.array([
246     [1,30,15],
247     [7,19,11],
248     [23,15,27]])
249
250 # Operasi pengurangan dua matriks numpy
251 result = X - Y
252
253 # cetak hasil
254 print("Hasil Pengurangan Matriks dari NumPy")
255 print(result)
```

Penjelasan:

◆ Baris ke-(240-243)

Membuat matriks X berukuran 3x3 menggunakan np.array.

Nilai-nilainya: $\begin{bmatrix} 22 & 4 & 6 \\ 18 & 10 & 12 \\ 14 & 36 & 18 \end{bmatrix}$

◆ Baris ke-(245-248)

Membuat matriks Y juga berukuran 3x3, dengan nilai: $\begin{bmatrix} 1 & 30 & 15 \\ 7 & 19 & 11 \\ 23 & 15 & 27 \end{bmatrix}$

◆ Baris ke-251

Melakukan pengurangan elemen-per-elemen antar dua matriks (element-wise subtraction) menggunakan operator -.

Contoh hasil:

- $X[0][0] - Y[0][0] = 22 - 1 = 21$
- $X[1][1] - Y[1][1] = 10 - 19 = -9$
- $X[2][2] - Y[2][2] = 18 - 27 = -9$

◆ Baris ke-(254-225)

- Mencetak teks penjelasan.
- Menampilkan hasil pengurangan dalam bentuk array NumPy.

Output:

```
[27 31 35]
Hasil Pengurangan Matriks dari NumPy
[[ 21 -26  -9]
 [ 11  -9   1]
 [-9  21  -9]]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 16

```
Welcome  contoh.py  modul-2.py x
modul-2.py >...
259 # impor library numpy
260 import numpy as np
261
262 # Membuat matriks dengan numpy
263 X = np.array([
264     [2,4,6],
265     [8,10,2],
266     [14,6,8]])
267
268 Y = np.array(
269     [[1,3,5],
270     [7,9,11],
271     [3,15,7]])
272
273 # Operasi perkalian dua matrik numpy
274 result = X * Y
275
276 # cetak hasil
277 print("Hasil Perkalian Matriks dari NumPy")
278 print(result)
```

Penjelasan:

◆ Baris ke-(263-266)

Membuat matriks X berukuran 3x3 menggunakan np.array.

Isi matriks X: $\begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 2 \\ 14 & 6 & 8 \end{bmatrix}$

◆ Baris ke-(268-271)

Membuat matriks Y juga berukuran 3x3, dengan isi: $\begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \\ 3 & 15 & 7 \end{bmatrix}$

◆ Baris ke-274

- Setiap elemen $X[i][j]$ dikalikan dengan $Y[i][j]$.
- Ini bukan perkalian matriks secara matematis, tapi perkalian elemen-wise (Hadamard product).

Contoh:

- $result[0][0] = 2 * 1 = 2$
- $result[1][1] = 10 * 9 = 90$
- $result[2][2] = 8 * 7 = 56$

◆ Baris ke-(277-278)

- Menampilkan teks penjelas.
- Mencetak hasil perkalian elemen-wise.

Output:

```
Hasil Perkalian Matriks dari NumPy
[[ 2 12 30]
 [56 90 22]
 [42 90 56]]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```


❖ Praktek 17

```
Welcome  contoh.py  modul-2.py X
modul-2.py > ...
282 # import library numpy
283 import numpy as np
284
285 # Membuat matriks dengan numpy
286 X = np.array([
287     [2,4,6],
288     [8,10,2],
289     [14,6,8]])
290
291 Y = np.array([
292     [1,3,5],
293     [7,9,11],
294     [3,15,7]])
295
296 # Operasi pembagian dua matrik numpy
297 result = X / Y
298
299 # cetak hasil
300 print("Hasil Pembagian Matriks dari NumPy")
301 print(result)
```

Pembahasan:

◆ Baris ke-(286-289)

Membuat matriks X berukuran 3x3 dengan isi: $\begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 2 \\ 14 & 6 & 8 \end{bmatrix}$

◆ Baris ke-(291-294)

Membuat matriks Y berukuran 3x3 dengan isi: $\begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \\ 3 & 15 & 7 \end{bmatrix}$

◆ Baris ke-297

- Setiap elemen $X[i][j]$ dibagi dengan $Y[i][j]$.
- Ini bukan pembagian matriks secara matematis, tapi pembagian elemen-per-elemen (element-wise division).

Contoh hasil:

- $X[0][0] / Y[0][0] = 2 / 1 = 2.0$
- $X[1][1] / Y[1][1] = 10 / 9 \approx 1.111...$
- $X[2][2] / Y[2][2] = 8 / 7 \approx 1.142...$

◆ Baris ke-(300-301)

- Mencetak judul penjelasan.
- Menampilkan hasil pembagian dalam bentuk array NumPy bertipe float.

Output:

```
Hasil Pembagian Matriks dari NumPy
[[2.         1.33333333 1.2       ]
 [1.14285714 1.11111111 0.18181818]
 [4.66666667 0.4       1.14285714]]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```

❖ Praktek 18

```
modul-2.py > ...
305 # import library numpy
306 import numpy as np
307
308 # membuat matriks
309 matriks_a = np.array([
310     [2, 4, 6],
311     [8, 10, 12],
312     [14, 16, 18]
313 ])
314
315 # cetak matriks
316 print("Matriks Sebelum Transpose")
317 print(matriks_a)
318
319 # transpose matriks_a
320 balik = matriks_a.transpose()
321
322 # cetak matriks setelah dibalik
323 print("Matriks Setelah Transpose")
324 print(balik)
```

Pembahasan:

◆ Baris ke-(309-313)

Membuat matriks 3x3 dengan np.array: $\begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix}$

◆ Baris ke-(316-317)

Menampilkan isi matriks sebelum dilakukan transpose.

◆ Baris ke-320

- Fungsi `.transpose()` membalik baris menjadi kolom dan kolom menjadi baris.
- Transpose dari matriks_a akan menjadi: $\begin{bmatrix} 2 & 8 & 14 \\ 4 & 10 & 16 \\ 6 & 12 & 18 \end{bmatrix}$

◆ Baris ke-(323-324)

Mencetak hasil matriks setelah dilakukan operasi transpose.

Output:

```
Matriks Sebelum Transpose
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
Matriks Setelah Transpose
[[ 2  8 14]
 [ 4 10 16]
 [ 6 12 18]]
PS C:\PENYIMPANAN DATA\Struktur Data\Modul 2 Array Vs Linked-List>
```