

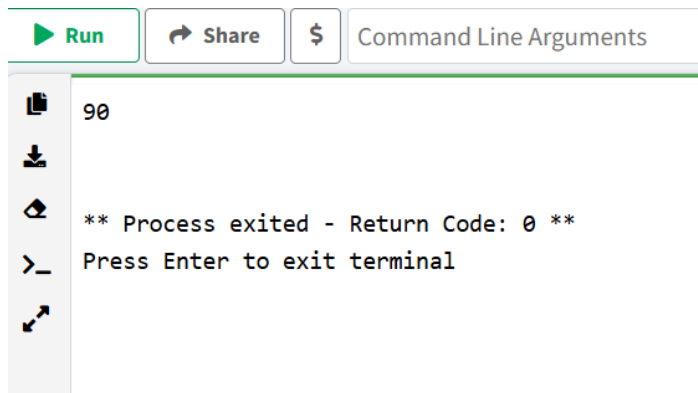
## Praktik 1 :

```
# impor library numpy
import numpy as np

# membuat array dengan numpy
nilai_siswa = np.array([85, 55, 40, 90])

# akses data pada array
print(nilai_siswa[3])
```

## Hasil Run :



## Penjelasannya :

Baris -1 Mengimpor library NumPy dan memberi alias np.

Baris -2 Membuat array 1 dimensi berisi angka 1, 2, dan 3.

Baris -3 Membuat array 2 dimensi (matriks) berisi dua baris dan dua kolom.

## Praktik 2 :

```
# impor library numpy
import numpy as np

# membuat array dengan numpy
nilai_siswa_1 = np.array([75, 65, 45, 80])
nilai_siswa_2 = np.array([[85, 55, 40], [50, 40, 90]])

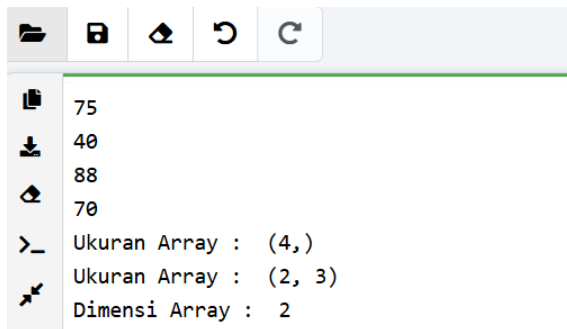
# cara akses elemen array
print(nilai_siswa_1[0])
print(nilai_siswa_2[1][1])

# mengubah nilai elemen array
nilai_siswa_1[0] = 88
nilai_siswa_2[1][1] = 70

# cek perubahannya dengan akses elemen array
print(nilai_siswa_1[0])
print(nilai_siswa_2[1][1])

# Cek ukuran dan dimensi array
print("Ukuran Array : ", nilai_siswa_1.shape)
print("Ukuran Array : ", nilai_siswa_2.shape)
print("Dimensi Array : ", nilai_siswa_2.ndim)
```

## Hasil run :



## Penjelasannya :

Baris -1 Mengimpor NumPy untuk digunakan dalam manipulasi array.

Baris -2 & -3 Membuat array 1 dimensi (nilai\_siswa\_1) dan array 2 dimensi (nilai\_siswa\_2).

Baris -4 & -5 Mengakses elemen array: elemen pertama dari nilai\_siswa\_1 dan baris ke-2 kolom ke-2 dari nilai\_siswa\_2.

Baris -6 & -7 Mengubah nilai elemen pada array.

Baris -8 & -9 Mengecek nilai yang telah diubah.

Baris -10 , - 11 & -12 Menampilkan ukuran dan jumlah dimensi array:

- shape: menunjukkan ukuran array (jumlah elemen tiap dimensi).
- ndim: menunjukkan jumlah dimensi array.

### Praktik 3 :

```
# impor library numpy
import numpy as np

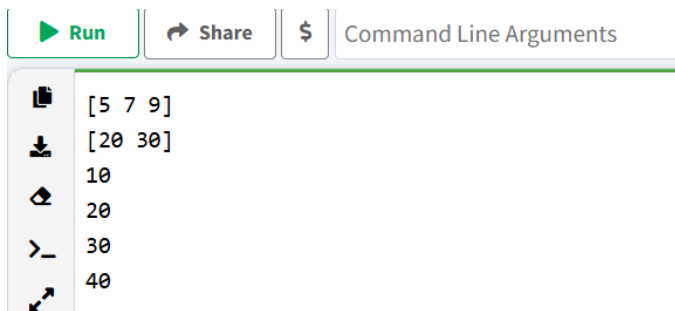
# membuat array
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# menggunakan operasi penjumlahan pada 2 array
print(a + b)      # array([5, 7, 9])

# Indexing dan Slicing pada Array
arr = np.array([10, 20, 30, 40])
print(arr[1:3])    # array([20, 30])

# iterasi pada array
for x in arr:
    print(x)
```

### Hasil Run :



### Penjelasannya :

Baris -1 Mengimpor library NumPy untuk digunakan dalam manipulasi array.

Baris -2 & -3 Membuat dua array 1 dimensi (a dan b).

Baris -4 Menjumlahkan elemen array a dan b satu per satu:

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

Baris -5 & -6 Mengambil elemen dari indeks 1 sampai sebelum indeks 3 (yaitu elemen ke-2 dan ke-3).

Baris -7 & -8 Melakukan perulangan untuk mencetak setiap elemen dalam array arr:

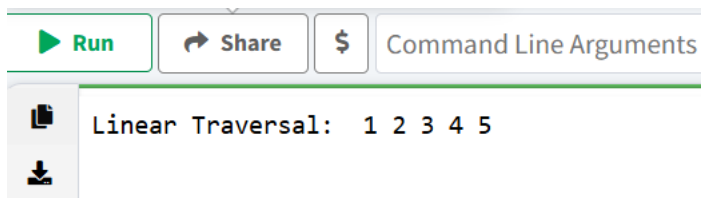
- Output: 10, 20, 30, 40 (dicetak satu per satu).

#### Praktik 4 :

```
# membuat array
arr = [1, 2, 3, 4, 5]

# Linear Traversal ke tiap elemen arr
print("Linear Traversal: ", end=" ")
for i in arr:
    print(i, end=" ")
print()
```

Hasil run :



Penjelasannya :

Baris -1 Membuat list (array) berisi elemen 1 sampai 5.

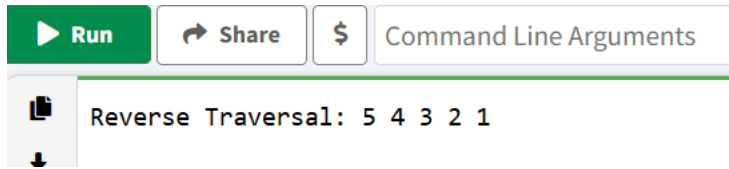
Baris -2 , -3 & -4 Melakukan **linear traversal**, yaitu menelusuri setiap elemen dalam array satu per satu, dan mencetaknya dalam satu baris.

#### Praktik 5 :

```
# membuat array
arr = [1, 2, 3, 4, 5]

# Reverse Traversal dari elemen akhir
print("Reverse Traversal: ", end="")
for i in range(len(arr) - 1, -1, -1):
    print(arr[i], end=" ")
print()
```

Hasil run :



The screenshot shows a code editor interface with three buttons at the top: 'Run' (green), 'Share' (blue), and '\$' (grey). Below the buttons, the output of the code is displayed in a grey box: 'Reverse Traversal: 5 4 3 2 1'.

Penjelasannya :

Baris -1 Membuat list (array) dengan elemen 1 sampai 5.

Baris -2 , -3 & -4 Melakukan **reverse traversal**, yaitu menelusuri array dari elemen terakhir ke pertama.

`range(len(arr) - 1, -1, -1)` artinya mulai dari indeks terakhir (4) hingga indeks pertama (0), mundur satu per satu.

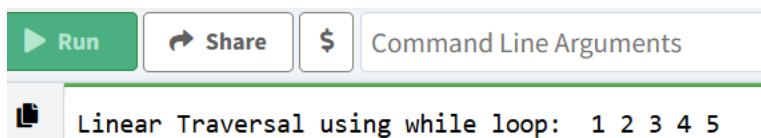
Praktik 7 :

```
# membuat array
arr = [1, 2, 3, 4, 5]

# mendeklarasikan nilai awal
n = len(arr)
i = 0

print("Linear Traversal using while loop: ", end=" ")
# Linear Traversal dengan while
while i < n:
    print(arr[i], end=" ")
    i += 1
print()
```

Hasil run :



The screenshot shows a code editor interface with three buttons at the top: 'Run' (green), 'Share' (blue), and '\$' (grey). Below the buttons, the output of the code is displayed in a grey box: 'Linear Traversal using while loop: 1 2 3 4 5'.

Penjelasannya :

Baris -1 Membuat array berisi angka 1 hingga 5.

Baris -2 & -3 Mendefinisikan n sebagai panjang array dan i sebagai indeks awal.

Baris -4 , -5 , -6 & -7 Melakukan **linear traversal** menggunakan perulangan while. Elemen array dicetak satu per satu dari indeks 0 hingga akhir.

Praktik 8 :

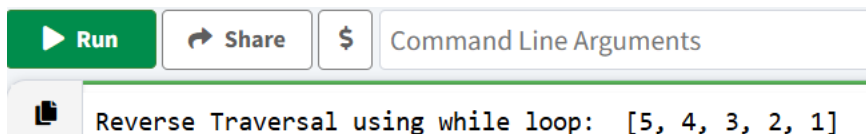
```
# membuat array
arr = [1, 2, 3, 4, 5]

# mendeklarasikan nilai awal
start = 0
end = len(arr) - 1

print("Reverse Traversal using while loop: ", end=" ")
# Reverse Traversal dengan while
while start < end:

    arr[start], arr[end] = arr[end], arr[start]
    start += 1
    end -= 1
print(arr)
```

Hasil run :



Penjelasannya :

Baris -1 Membuat array berisi angka 1 hingga 5.

Baris -2 & -3 start adalah indeks awal, dan end adalah indeks akhir array.

Baris -4 , -5 , -6 , -7 & -8 Melakukan **reverse traversal dan membalik isi array** menggunakan while loop. Elemen paling depan dan paling belakang ditukar, lalu indeks digeser ke tengah hingga selesai.

Baris -9 Menampilkan array yang telah dibalik

Praktik 9 :

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

# cetak arr sebelum penyisipan
print("Array Sebelum Insertion : ", arr)

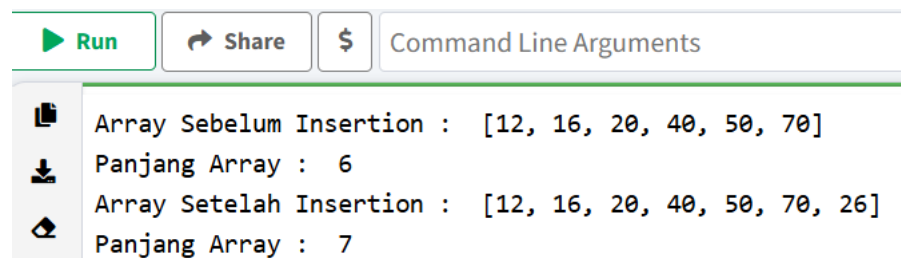
# cetak panjang array sebelum penyisipan
print("Panjang Array : ", len(arr))

# menyisipkan array di akhir elemen menggunakan .append()
arr.append(26)

# cetak arr setelah penyisipan
print("Array Setelah Insertion : ", arr)

# cetak panjang array setelah penyisipan
print("Panjang Array : ", len(arr))
```

Hasil run :



```
Run Share $ Command Line Arguments

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
```

Penjelasannya :

Baris -1 Membuat array (list) awal berisi beberapa angka.

Baris -2 & -3 Menampilkan isi array dan panjangnya sebelum penyisipan.

Baris -4 Menyisipkan (menambahkan) elemen 26 di **akhir array** menggunakan metode .append().

Baris -5 & -6 Menampilkan isi array dan panjangnya setelah elemen ditambahkan.

## Praktik 10 :

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

# cetak arr sebelum penyisipan
print("Array Sebelum Insertion : ", arr)

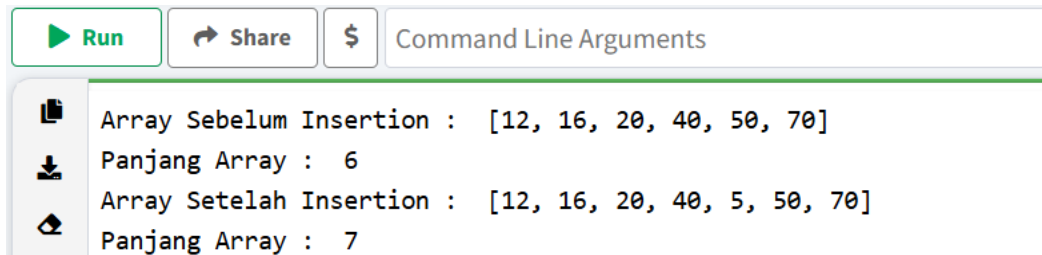
# cetak panjang array sebelum penyisipan
print("Panjang Array : ", len(arr))

# menyisipkan array pada tengah elemen menggunakan .insert(pos, x)
arr.insert(4, 5)

# cetak arr setelah penyisipan
print("Array Setelah Insertion : ", arr)

# cetak panjang array setelah penyisipan
print("Panjang Array : ", len(arr))
```

Hasil run :



```
Run Share $ Command Line Arguments
```

```
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
```

Penjelasannya :

Baris – 1 Membuat array awal dengan beberapa angka.

Baris -2 & -3 Menampilkan isi dan panjang array sebelum penyisipan.

Baris -4 Menyisipkan elemen 5 pada indeks ke-4 (di antara elemen 50 dan 40).

Metode .insert(pos, x) menambahkan elemen x di posisi pos.

Baris -5 & -6 Menyisipkan elemen 5 pada indeks ke-4 (di antara elemen 50 dan 40).

Metode .insert(pos, x) menambahkan elemen x di posisi pos.



## Praktik 11 :

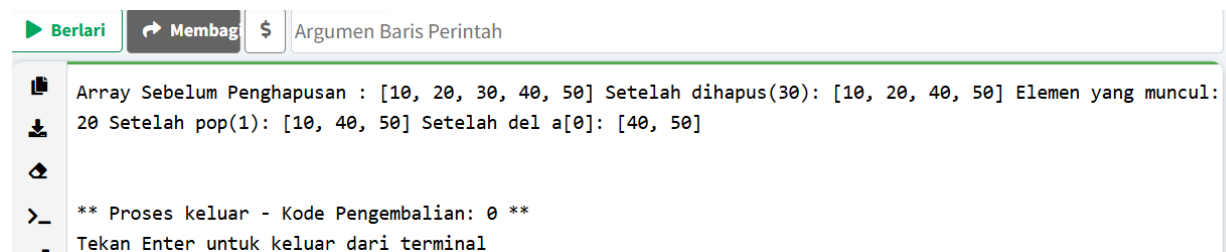
```
# membuat array
a = [10, 20, 30, 40, 50]
print("Array Sebelum Deletion : ", a)

# menghapus elemen array pertama yang nilainya 30
a.remove(30)
print("Setelah remove(30):", a)

# menghapus elemen array pada index 1 (20)
popped_val = a.pop(1)
print("Popped element:", popped_val)
print("Setelah pop(1):", a)

# Menghapus elemen pertama (10)
del a[0]
print("Setelah del a[0]:", a)
```

## Hasil run :



```
▶ Berlari  ➡ Membagi  $ Argumen Baris Perintah

Array Sebelum Penghapusan : [10, 20, 30, 40, 50] Setelah dihapus(30): [10, 20, 40, 50] Elemen yang muncul:
20 Setelah pop(1): [10, 40, 50] Setelah del a[0]: [40, 50]

** Proses keluar - Kode Pengembalian: 0 **
Tekan Enter untuk keluar dari terminal
```

## Penjelasannya :

Baris -1 Membuat **list a** berisi lima elemen: [10, 20, 30, 40, 50].

Baris -2 Menampilkan isi list sebelum dilakukan penghapusan elemen.

Baris -3 Menghapus elemen **bernilai 30** dari list a.

Metode `.remove()` menghapus **elemen berdasarkan nilai**, bukan indeks.

Baris -4 Menampilkan list setelah elemen 30 dihapus.

Baris -5 Menghapus elemen **pada indeks ke-1**, yaitu 20 (karena setelah 30 dihapus, elemen di indeks 1 adalah 20).

Nilai yang dihapus disimpan ke variabel `popped_val`.

Baris -6 Menampilkan elemen yang telah di-*pop* (yaitu 20).

Baris -7 Menampilkan isi list setelah `pop(1)`.

Baris -8 Menghapus elemen **pada indeks ke-0** (yaitu 10) menggunakan perintah del.

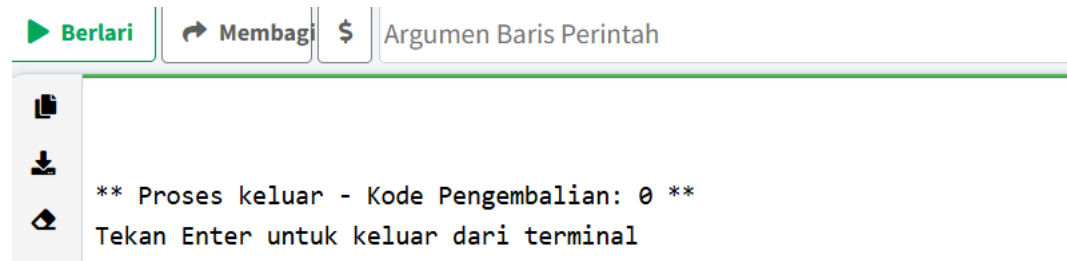
Baris -9 Menampilkan list setelah elemen pertama (10) dihapus.

Praktik 12 :

```
# impor library numpy
import numpy as np

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

Hasil run :



Penjelasannya :

Baris -1 Mengimpor library numpy sebagai np.

Library ini digunakan untuk manipulasi array dan operasi matematika numerik.

Baris -2 Membuat matriks 3x3 menggunakan np.array() dan menyimpannya ke dalam variabel matriks\_np.

Matriks ini memiliki 3 baris dan 3 kolom:

[1, 2, 3]
[4, 5, 6]
[7, 8, 9]

### Prakti 13 :

```
# 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)):
    # mengulang sebanyak column (kolom)
    for j in range(len(X[0])):
        result[i][j] = X[i][j] + Y[i][j]


print("Hasil Penjumlahan Matriks dari LIST")


# cetak hasil penjumlahan secara iteratif
for r in result:
    print(r)
```


Hasil run :


 Berlari


 Membagi

 Argumen Baris Perintah

 Hasil Penjumlahan Matriks dari LIST [17, 15, 4] [10, 12, 9] [11, 13, 18]



 \*\* Proses keluar - Kode Pengembalian: 0 \*\*

 >\_ Tekan Enter untuk keluar dari terminal

Penjelasannya :

Baris -1 Komentar penjelas bahwa program ini menjumlahkan dua matriks yang dibuat dari list (bukan NumPy).

Baris -2 Membuat matriks X berupa list dua dimensi (3x3) yang berisi bilangan-bilangan.

Baris -3 Membuat matriks Y dengan ukuran dan struktur yang sama seperti X.

Baris -4 Membuat matriks result berukuran 3x3 yang diisi dengan nol, sebagai tempat menyimpan hasil penjumlahan X + Y.

Baris -5 Memulai loop luar yang berjalan sebanyak jumlah baris pada matriks (3 kali). i mewakili indeks baris.

Baris -6 Memulai loop dalam yang berjalan sebanyak jumlah kolom dalam satu baris (juga 3 kali). j mewakili indeks kolom.

Baris -7 Menjumlahkan elemen dari X dan Y pada posisi [i][j], lalu disimpan ke dalam result[i][j].

Baris -8 Menampilkan teks judul sebelum mencetak hasil penjumlahan.

Baris -9 Melakukan loop untuk mencetak setiap baris dari hasil penjumlahan matriks.

Praktik 14 :

```
# impor library numpy
import numpy as np

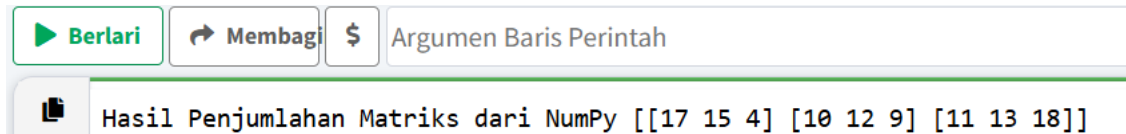
# Membuat matriks dengan numpy
X = np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])

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

# Operasi penjumlahan dua matrik numpy
result = X + Y

# cetak hasil
print("Hasil Penjumlahan Matriks dari NumPy")
print(result)
```

Hasil run :



Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np.

Library ini wajib digunakan untuk operasi numerik seperti array dan matriks.

Baris -2 Membuat matriks X berukuran 3×3 menggunakan np.array().

Baris -3 Membuat matriks Y juga berukuran 3×3 menggunakan np.array().

Baris -4 Melakukan penjumlahan elemen per elemen antara matriks X dan Y menggunakan operator +.

Contoh:  $X[0][0] + Y[0][0] = 12 + 5 = 17$ .

Hasil penjumlahan disimpan dalam variabel result.

Baris -5 Menampilkan teks informasi "Hasil Penjumlahan Matriks dari NumPy" ke layar.

Baris -6 Mencetak hasil penjumlahan matriks X + Y yang tersimpan di result.

### Praktik 15 :

```
# impor library numpy
import numpy as np

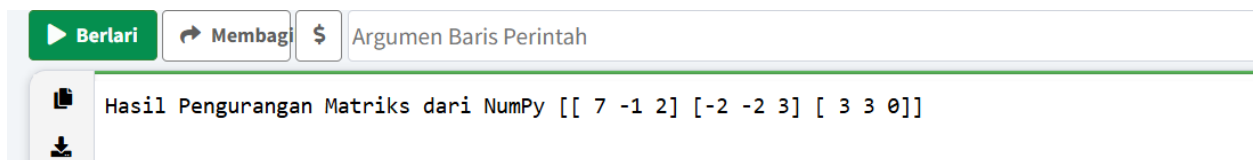
# Membuat matriks dengan numpy
X = np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])

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

# Operasi pengurangan dua matrik numpy
result = X - Y

# cetak hasil
print("Hasil Pengurangan Matriks dari NumPy")
print(result)
```

### Hasil run :



### Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np.

NumPy digunakan untuk operasi array dan matriks secara efisien.

Baris -2 Membuat matriks X berukuran 3×3 menggunakan np.array().

Baris -3 Membuat matriks Y juga berukuran 3×3 menggunakan np.array().

Baris -4 Melakukan penjumlahan elemen per elemen antara matriks X dan Y menggunakan operator +.

Contoh:  $X[0][0] + Y[0][0] = 12 + 5 = 17$ .

Hasil penjumlahan disimpan dalam variabel result.

Baris -5 Menampilkan teks informasi "Hasil Penjumlahan Matriks dari NumPy" ke layar.

Baris -6 Mencetak hasil penjumlahan matriks X + Y yang tersimpan di result.

#### Praktik 16 :

```
# impor library numpy
import numpy as np

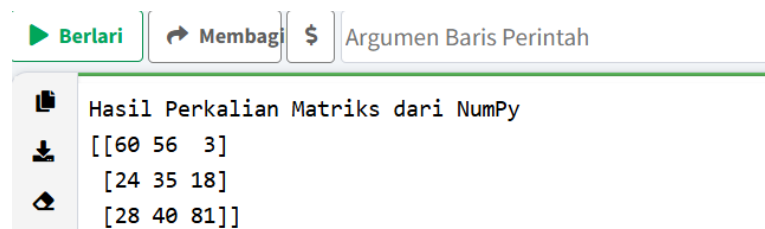
# Membuat matriks dengan numpy
X = np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])

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

# Operasi perkalian dua matrik numpy
result = X * Y

# cetak hasil
print("Hasil Perkalian Matriks dari NumPy")
print(result)
```

#### Hasil Run :



#### Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np.  
NumPy digunakan untuk operasi array dan matriks secara efisien.

Baris -2 Membuat matriks X berukuran 3×3 menggunakan np.array().

Baris -3 Membuat matriks Y juga berukuran 3×3 menggunakan np.array().

Baris -4 Melakukan penjumlahan elemen per elemen antara matriks X dan Y menggunakan operator +.

Contoh:  $X[0][0] + Y[0][0] = 12 + 5 = 17$ .

Hasil penjumlahan disimpan dalam variabel result.

Baris -5 Menampilkan teks informasi "Hasil Penjumlahan Matriks dari NumPy" ke layar.

Baris -6 Mencetak hasil penjumlahan matriks X + Y yang tersimpan di variabel result.

## Praktik 9 :

```
# Praktek 17 : Operasi Pembagian Matriks dengan numpy
# impor library numpy
import numpy as np

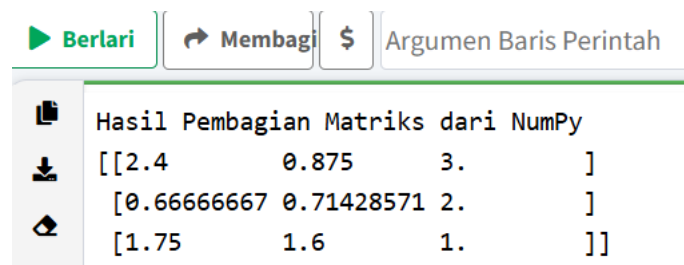
# Membuat matriks dengan numpy
X = np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])

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

# Operasi pembagian dua matrik numpy
result = X / Y

# cetak hasil
print("Hasil Pembagian Matriks dari NumPy")
print(result)
```

Hasil run :



The screenshot shows a code execution interface with three buttons at the top: 'Berlari' (green), 'Membagi' (blue), and '\$' (grey). Below the buttons, the output of the code is displayed in a text area. The output is a 3x3 matrix of floating-point numbers, which is the result of dividing matrix X by matrix Y. The text area has a dark background and white text. The output is as follows:

```
Hasil Pembagian Matriks dari NumPy
[[2.4      0.875    3.        ]
 [0.66666667 0.71428571 2.        ]
 [1.75     1.6      1.        ]]
```

Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np.  
NumPy digunakan untuk operasi array dan matriks secara efisien.

Baris -2 Membuat matriks X berukuran 3×3 menggunakan np.array().

Baris -3 Membuat matriks Y juga berukuran 3×3 menggunakan np.array().



Baris -4 Melakukan pembagian elemen per elemen antara matriks X dan Y menggunakan operator /.

Contoh:  $X[0][0] / Y[0][0] = 12 / 5 = 2.4$ .

Hasil pembagian disimpan dalam variabel result.

Baris -5 Menampilkan teks informasi "Hasil Pembagian Matriks dari NumPy" ke layar.

Baris -6 Mencetak hasil pembagian matriks X / Y yang tersimpan di variabel result.

## Praktik 18 :

```
# impor library numpy
import numpy as np

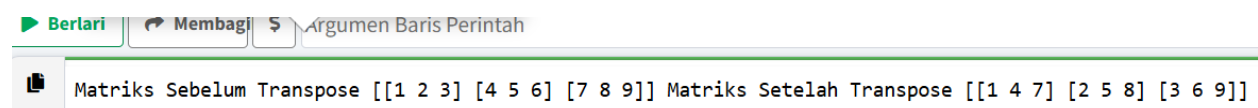
# membuat matriks
matriks_a = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
])

# cetak matriks
print("Matriks Sebelum Transpose")
print(matriks_a)

# transpose matriks_a
balik = matriks_a.transpose()

# cetak matriks setelah dibalik
print("Matriks Setelah Transpose")
print(balik)
```

## Hasil run :



## Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np untuk operasi array dan matriks.

Baris -2 Membuat matriks matriks\_a berukuran 3×3 menggunakan np.array().

Baris -3 Menampilkan teks "Matriks Sebelum Transpose" sebagai informasi.

Baris -4 Mencetak isi matriks matriks\_a sebelum operasi transpose.

Baris -5 Melakukan transpose matriks matriks\_a dengan metode .transpose() dan menyimpan hasilnya di variabel balik.

Baris -6 Menampilkan teks "Matriks Setelah Transpose" sebagai informasi.

Baris -7 Mencetak matriks hasil transpose yang tersimpan di variabel balik.

#### Praktik 19 :

```
# impor library numpy
import numpy as np

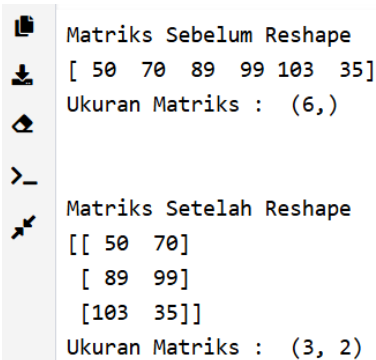
# membuat array 1 dimensi
arr_1d = np.array([50, 70, 89, 99, 103, 35])

# cetak matriks sebelum reshape
print("Matriks Sebelum Reshape")
print(arr_1d)
print("Ukuran Matriks : ", arr_1d.shape)
print("\n")

# mengubah matriks menjadi ordo 3 x 2
ubah = arr_1d.reshape(3, 2)

# cetak matriks setelah reshape ke ordo 3 x 2
print("Matriks Setelah Reshape")
print(ubah)
print("Ukuran Matriks : ", ubah.shape)
```

#### Hasil run :



```
Matriks Sebelum Reshape
[ 50 70 89 99 103 35]
Ukuran Matriks : (6,)

Matriks Setelah Reshape
[[ 50 70]
 [ 89 99]
 [103 35]]
Ukuran Matriks : (3, 2)
```

#### Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np untuk operasi array dan matriks.

Baris -2 Membuat array 1 dimensi arr\_1d berisi 6 elemen menggunakan np.array().

Baris -3 Menampilkan teks "Matriks Sebelum Reshape" sebagai informasi.

Baris -4 Mencetak isi array arr\_1d sebelum diubah bentuknya.

Baris -5 Menampilkan ukuran (dimensi) array arr\_1d menggunakan atribut .shape (hasil: (6,)).

Baris -6 Mencetak baris kosong untuk memberi jarak.

Baris -7 Mengubah bentuk array arr\_1d menjadi matriks 3 baris dan 2 kolom dengan metode .reshape(3, 2), hasil disimpan di variabel ubah.

Baris -8 Menampilkan teks "Matriks Setelah Reshape" sebagai informasi.

Baris -9 Mencetak isi matriks ubah setelah direshape.

Baris -10 Menampilkan ukuran (dimensi) matriks ubah dengan .shape (hasil: (3, 2)).

Praktik 20 :

```
import numpy as np

# vektor baris (1 dimensi)
vek_1 = np.array([1, 2, 3])

# vektor kolom (2 dimensi, bentuk matriks 3x1)
vek_2 = np.array([
    [1],
    [2],
    [3]
])

# membuat vektor kolom dengan transpose
vek_3 = np.array([[1, 2, 3]]).T

print("Vektor Baris")
print(vek_1)

print("Vektor Kolom")
print(vek_2)

print("Vektor Kolom dengan transpose()")
print(vek_3)
```

Hasil run :

```
Vektor Baris
[1 2 3]
Vektor Kolom
[[1]
 [2]
 [3]]
Vektor Kolom dengan transpose()
[[1]
 [2]
 [3]]
```

Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np.

Baris -2 Membuat vektor baris 1 dimensi dengan elemen [1, 2, 3] menggunakan np.array().

Baris -3 Membuat vektor kolom 2 dimensi berukuran 3x1 dengan elemen masing-masing di dalam list terpisah.

Baris -4 Membuat vektor kolom dengan cara mentranspose (transpose) dari vektor baris 1x3 yang dibungkus dalam list 2 dimensi.

Baris -5 Menampilkan teks "Vektor Baris" sebagai informasi.

Baris -6 Mencetak isi vektor baris vek\_1.

Baris -7 Menampilkan teks "Vektor Kolom" sebagai informasi.

Baris -8 Mencetak isi vektor kolom vek\_2.

Baris -9 Menampilkan teks "Vektor Kolom dengan transpose()" sebagai informasi.

Baris -10 Mencetak isi vektor kolom hasil transpose vek\_3.

## Praktik 21 :

```
# impor library numpy
import numpy as np

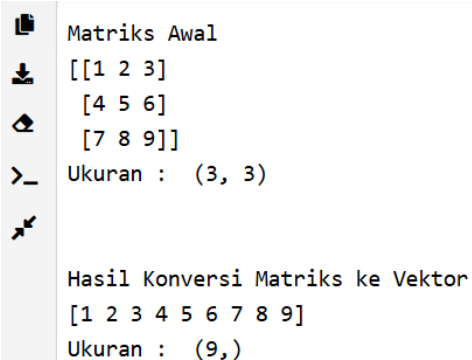
# membuat matriks
matriks_a = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
])

# cetak matriks awal
print("Matriks Awal")
print(matriks_a)
print("Ukuran : ", matriks_a.shape)
print("\n")

# ubah matriks menjadi vektor
jd_vektor = matriks_a.flatten()

# cetak vektor
print("Hasil Konversi Matriks ke Vektor")
print(jd_vektor)
print("Ukuran : ", jd_vektor.shape)
```

## Hasil run :



```
Matriks Awal
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Ukuran :  (3, 3)

Hasil Konversi Matriks ke Vektor
[1 2 3 4 5 6 7 8 9]
Ukuran :  (9,)
```

## Penjelasannya :

Baris -1 Mengimpor library NumPy sebagai np untuk operasi array dan matriks.

Baris -2 Membuat matriks matriks\_a berukuran 3×3 menggunakan np.array().

Baris -3 Menampilkan teks "Matriks Awal" sebagai informasi.

Baris -4 Mencetak isi matriks matriks\_a sebelum diubah.

Baris -5 Menampilkan ukuran (dimensi) matriks matriks\_a dengan atribut .shape (hasil: (3, 3)).

Baris -6 Mencetak baris kosong untuk memberi jarak.

Baris -7 Mengubah matriks matriks\_a menjadi vektor 1 dimensi menggunakan metode .flatten().

Baris -8 Menampilkan teks "Hasil Konversi Matriks ke Vektor" sebagai informasi.

Baris -9 Mencetak isi vektor hasil konversi `jd_vektor`.

Baris -10 Menampilkan ukuran (dimensi) vektor `jd_vektor` menggunakan `.shape` (hasil: (9,)).