

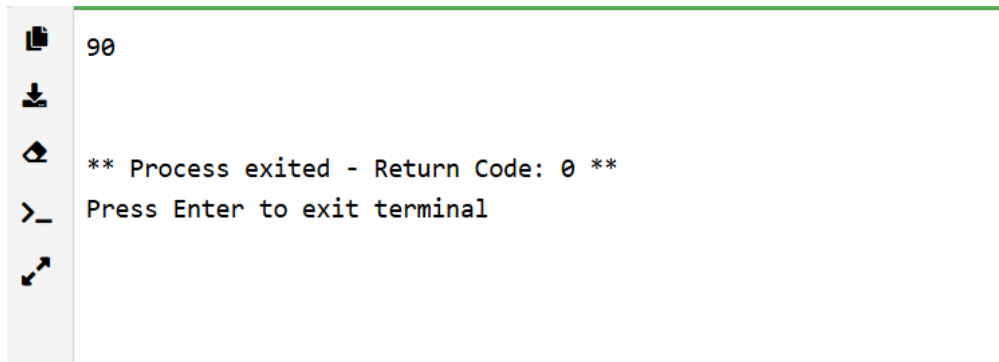
## Praktek 1:

```
# 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])
```

Hasilnya:



```
90

** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Penjelasan:

Baris ke-1: Baris ini mengimpor library numpy, yaitu library Pythin yang digunakan untuk operasi numerik, Khususnya array dan matriks. Kata as np digunakan untuk menyingkat penulisan numpy menjadi np, sehingga lebih praktis digunakan.

Baris ke-2: Baris ini membuat sebuah array numpy yang berisi data nilai siswa: 85, 55, 40, dan 90. Fungsi np.array([...]) digunakan untuk membuat array dari daftar (list) Python.

Baris ke-3: Baris ini mencetak elemen ke-4 dari array nilai\_siswa, yaitu 90. Perlu diingat bahwa indeks dimulai dari 0, sehingga:

- Nilai\_siswa[0] adalah 85
- Nilai\_siswa[1] adalah 55
- Nilai\_siswa[2] adalah 40
- Nilai\_siswa[3] adalah 90

## Praktek 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, 99]])

# 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)
```

Hasilnya:

	75
	40
	88
	70
	Ukuran Array : (4,)
	Ukuran Array : (2, 3)
	Dimensi Array : 2

Penjelasan:

Baris ke-1: Mengimpor library numpy dan disingkat menjadi np agar lebih ringkas saat digunakan.

Baris ke-2: Membuat array dengan numpy.

- nilai\_siswa\_1: array 1 dimensi yang berisi nilai-nilai siswa.
- nilai\_siswa\_2: array 2 dimensi (matriks 2 x 3), yang terdiri dari 2 baris dan 3 kolom.

Baris ke-3: Cara akses elemen array

- nilai\_siswa\_1[0]: mengakses elemen pertama dari array 1 dimensi (75).
- nilai\_siswa\_2[1][1]: mengakses elemen baris ke-2, kolom ke-2 dari array 2 dimensi (40 sebelum diubah)

Baris ke-4: Mengubah Nilai elemen array

- mengubah nilai pertama di nilai\_siswa\_1 menjadi 88.
- Mengubah nilai pada baris ke-2, kolom ke-2 di nilai\_siswa\_2 menjadi 70.

Baris ke-5: Mengecek perubahannya dengan akses elemen array

- Menampilkan nilai yang sudah diperbarui 88 dan 70.

Baris ke-6: Mengecek ukuran dan dimensi array

- shape: menunjukkan ukuran/tata letak array.
  - o nilai\_siswa\_1. shape = (4, ) artinya 1 dimensi dengan 4 elemen.
  - o nilai\_siswa\_2. shape = (2, 3) artinya 2 baris dan 3 kolom.
- ndim: menunjukkan jumlah dimensi array.
  - o nilai\_siswa\_2. Ndim = 2 karena array 2 dimensi.

### Praktek 3:

```
# impor library numpy
import numpy as np

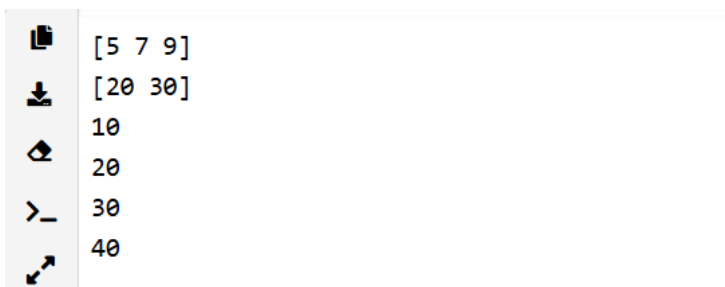
# membuat array
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

# 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)
```

Hasilnya:



```
[5 7 9]
[20 30]
10
20
30
40
```

Penjelasan:

Baris ke-1: Komentar bahwa baris selanjutnya akan mengimpor library numpy.

Baris ke-2: Mengimpor library numpy dan menyingkatnya menjadi np agar lebih ringkas saat dipakai.

Baris ke-3: Komentar bahwa bagian ini akan membuat array menggunakan numpy.

Baris ke-4: Membuat array a berisi elemen [1,2,3].

Baris ke-5: Membuat array b berisi elemen [4,5,6].

Baris ke-6: Komentar bahwa bagian ini melakukan operasi penjumlahan antar array.

Baris ke-7: Menjumlahkan elemen-elemen array a dan b satu per satu:

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

Baris ke-8: Komentar bahwa bagian ini akan menunjukkan cara mengambil Sebagian elemen dari array (slicing).

Bagian ke-9: Membuat array arr berisi elemen [10,20,30,40].

Bagian ke-10: Mengambil elemen dari indeks ke-1 sampai sebelum indeks ke-3 (slicing)

- Indeks ke-1 = 20
- Indeks ke-2 = 30

Bagian ke-11: Komentar bahwa bagian ini akan melakukan perulangan pada elemen array.

Bagian ke-12: Melakukan iterasi (perulangan) untuk setiap elemen x di dalam array arr.

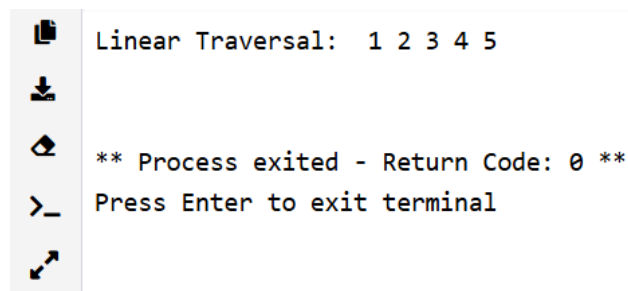
Bagian ke-13: Menampilkan setiap nilai x (elemen array).

#### Praktek 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()
```

Hasilnya:



```
Linear Traversal: 1 2 3 4 5

** Process exited - Return Code: 0 **

Press Enter to exit terminal
```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa baris berikutnya akan membuat array atau daftar.

Baris ke-2: Membuat array (list) bernama arr yang berisi lima elemen angka, yaitu 1 sampai 5.

Baris ke-3: Komentar yang menjelaskan bahwa bagian ini akan melakukan penelusuran (traversal) secara linear ke setiap elemen array.

Baris ke-4: Mencetak teks "Linear Traversal:" ke layar tanpa berpindah ke baris baru, agar elemen-elemen array dapat dicetak di baris yang sama.

Baris ke-5: Memulai perulangan for untuk menelusuri setiap elemen di dalam array arr.

Baris ke-6: Mencetak elemen saat ini (i) tanpa berpindah baris, dan memberi spasi setelah setiap angka.

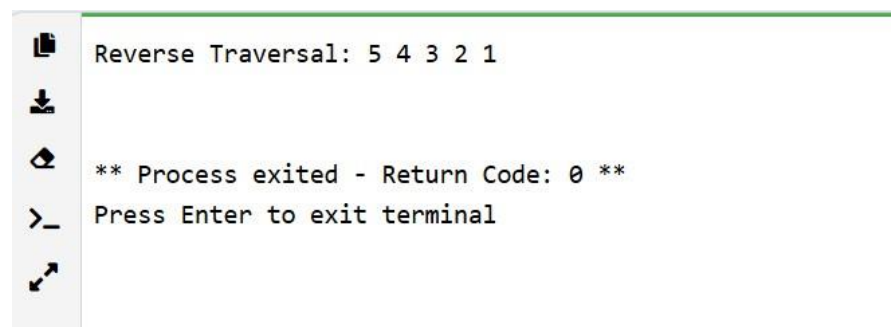
Baris ke-7: Mencetak baris baru setelah semua elemen array ditampilkan, agar tampilan rapi dan tidak menempel ke output berikutnya.

### 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()
```

Hasilnya:



```
Reverse Traversal: 5 4 3 2 1

** Process exited - Return Code: 0 **

Press Enter to exit terminal
```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa baris selanjutnya akan membuat array atau list.

Baris ke-2: Membuat array (list) bernama arr yang berisi lima elemen: 1, 2, 3, 4, dan 5.

Baris ke-3: Komentar yang menjelaskan bahwa bagian ini akan menelusuri (traversal) elemen array dari belakang ke depan (reverse).

Baris ke-4: Mencetak teks "Reverse Traversal:" ke layar tanpa ganti baris, supaya elemen-elemen bisa muncul di baris yang sama.

Baris ke-5: Melakukan perulangan for untuk mengambil indeks dari elemen terakhir (`len(arr) - 1`) sampai indeks pertama (0) secara mundur (-1 sebagai langkah).

Baris ke-6: Mencetak elemen array berdasarkan indeks `i`, dengan spasi di antaranya dan tetap di baris yang sama.

Baris ke-7: Mencetak baris kosong setelah semua elemen ditampilkan agar output rapi dan tidak menempel ke output berikutnya.

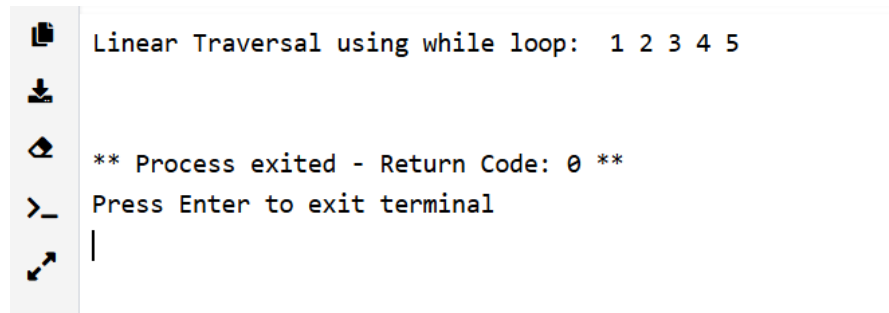
### Praktek 7:

```
# 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()
```

Hasilnya:



```
Linear Traversal using while loop: 1 2 3 4 5

** Process exited - Return Code: 0 **

Press Enter to exit terminal
```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa baris berikutnya akan membuat array (list).

Baris ke-2: Membuat array (list) bernama `arr` yang berisi lima elemen: 1, 2, 3, 4, dan 5.

Baris ke-3: Komentar yang menjelaskan bahwa nilai awal akan dideklarasikan sebelum melakukan perulangan.

Baris ke-4: Menyimpan panjang array `arr` ke dalam variabel `n`. Ini akan digunakan sebagai batas akhir perulangan.

Baris ke-5: Menginisialisasi variabel penghitung `i` dengan nilai 0. Ini sebagai indeks awal untuk mengakses elemen array.

Baris ke-6: Mencetak teks "Linear Traversal using while loop:" ke layar tanpa ganti baris, agar hasil traversal muncul di baris yang sama.

Baris ke-7: Komentar yang menjelaskan bahwa bagian ini menggunakan while loop untuk traversal array.

Baris ke-8: Memulai perulangan while selama i masih lebih kecil dari n (panjang array).

Baris ke-9: Mencetak elemen array pada indeks ke-i di baris yang sama, dipisahkan oleh spasi.

Baris ke-10: Menambahkan nilai i sebanyak 1 untuk melanjutkan ke elemen berikutnya.

Baris ke-11: Mencetak baris kosong setelah traversal selesai, supaya output rapi dan tidak menempel pada baris selanjutnya.

### Praktek 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)
```

Hasilnya:

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

** Process exited - Return Code: 0 **
Press Enter to exit terminal
|
```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa baris berikutnya membuat array (list).

Baris ke-2: Membuat array bernama arr yang berisi lima elemen: 1, 2, 3, 4, dan 5.

Baris ke-3: Komentar yang menunjukkan bahwa kita akan mendeklarasikan nilai awal sebelum perulangan.

Baris ke-4: Membuat variabel start dengan nilai 0 sebagai indeks awal.

Baris ke-5: Membuat variabel end dengan nilai indeks terakhir array (panjang array dikurangi 1).

Baris ke-6: Mencetak teks "Reverse Traversal using while loop:" di layar tanpa ganti baris, agar output berada di baris yang sama.

Baris ke-7: Komentar yang menjelaskan bahwa perulangan dilakukan secara mundur menggunakan while.

Baris ke-8: Memulai perulangan while selama nilai start masih lebih kecil dari end.

Baris ke-9: Menukar elemen array pada indeks start dan end. Ini akan membalik posisi elemen dari ujung ke tengah.

Baris ke-10: Menambahkan nilai start sebanyak 1 untuk maju ke indeks berikutnya dari depan.

Baris ke-11: Mengurangi nilai end sebanyak 1 untuk mundur ke indeks sebelumnya dari belakang.

Baris ke-12: Mencetak array setelah proses pembalikan selesai.

### Praktek 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))
```

Hasilnya:



```
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

** Process exited - Return Code: 0 **
Press Enter to exit terminal
```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa baris berikutnya membuat array.

Baris ke-2: Membuat array bernama arr yang berisi enam elemen: 12, 16, 20, 40, 50, dan 70.

Baris ke-3: Komentar yang menjelaskan bahwa baris selanjutnya akan mencetak array sebelum dilakukan penyisipan elemen.

Baris ke-4: Mencetak array arr sebelum elemen baru disisipkan.

Baris ke-5: Komentar yang menyatakan bahwa baris berikut akan mencetak panjang array sebelum penyisipan.

Baris ke-6: Mencetak jumlah elemen dalam array arr sebelum penyisipan.

Baris ke-7: Komentar yang menjelaskan bahwa penyisipan akan dilakukan di akhir array menggunakan metode .append().

Baris ke-8: Menyisipkan elemen 26 ke akhir array menggunakan metode .append().

Baris ke-9: Komentar yang menyatakan bahwa array akan dicetak setelah elemen disisipkan.

Baris ke-10: Mencetak array arr setelah elemen 26 ditambahkan.

Baris ke-11: Komentar yang menyatakan bahwa panjang array akan dicetak setelah penyisipan.

Baris ke-12: Mencetak jumlah elemen array setelah elemen baru ditambahkan.

## Praktek 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))

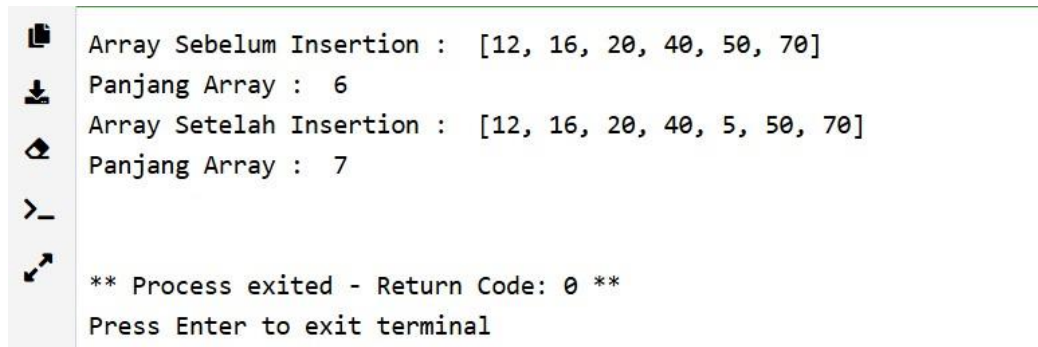
# menyisipkan array pada tengah elemen menggunakan .insert(pos, x)
arr.insert(4, 5)

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

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

```

Hasilnya:



```

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

** Process exited - Return Code: 0 **
Press Enter to exit terminal

```

Penjelasan:

Baris ke-1: Komentar yang menjelaskan bahwa array (list) akan dibuat.

Baris ke-2: Membuat array bernama arr dengan isi: 12, 16, 20, 40, 50, 70.

Baris ke-3: Komentar bahwa array akan dicetak sebelum elemen baru disisipkan.

Baris ke-4: Mencetak array sebelum proses penyisipan dilakukan.

Baris ke-5: Komentar bahwa panjang array akan dicetak sebelum penyisipan.

Baris ke-6: Mencetak jumlah elemen array sebelum disisipkan elemen baru.

Baris ke-7: Komentar yang menjelaskan bahwa elemen akan disisipkan di tengah array menggunakan metode .insert(posisi, nilai).

Baris ke-8: Menyisipkan angka 5 ke indeks ke-4 (yaitu sebelum angka 50). Jadi, 5 akan berada di antara 40 dan 50.

Baris ke-9: Komentar bahwa array akan dicetak setelah elemen berhasil disisipkan.

Baris ke-10: Mencetak array setelah elemen 5 disisipkan.

Baris ke-11: Komentar bahwa panjang array akan dicek kembali setelah penyisipan.

Baris ke-12: Mencetak panjang array setelah elemen 5 ditambahkan, kini bertambah menjadi 7 elemen.

### Praktek 11 :

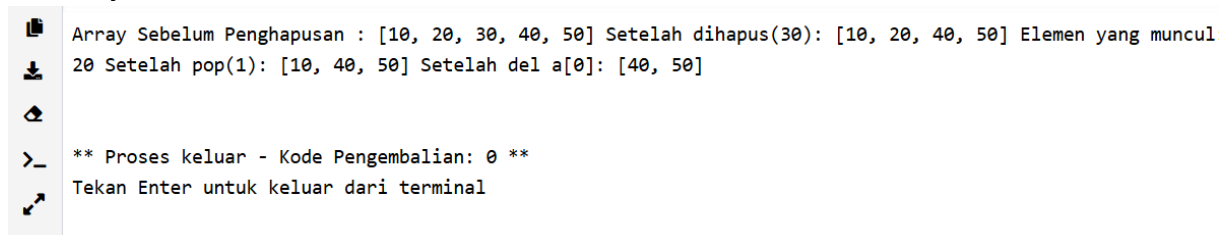
```
# membuat array
a = [10, 20, 30, 40, 50]
print("Array Sebelum Deletion : ", a)

# 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)
```

### Hasilnya :



```
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
```

### Penjelasan

baris 1 = membuat array a dengan isi [10, 20, 30, 40, 50]

baris 2 = menampilkan array sebelum dihapus elemen

baris 3 = menghapus elemen bernilai 30

baris 4 = menampilkan array setelah remove(30)

baris 5 = menghapus elemen di index 1 (yaitu 20) dan menyimpannya ke popped\_val

baris 6 = menampilkan elemen yang di-pop

baris 7 = menampilkan array setelah pop(1)

baris 8 = menghapus elemen di index 0 (yaitu 10)

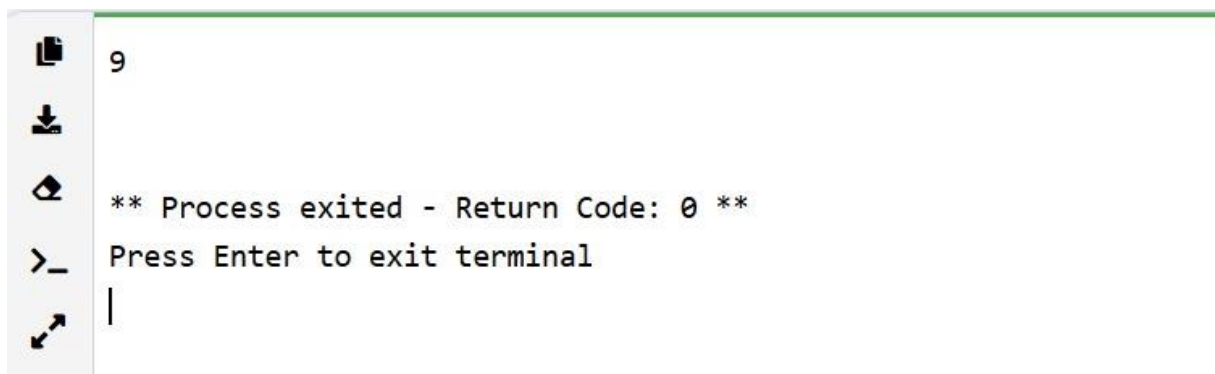
baris 9 = menampilkan array setelah del a[0]

## Praktek 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 :



```
9

** Process exited - Return Code: 0 **
>_ Press Enter to exit terminal
|
```

## Penjelasan :

misalnya kita ingin mengambil nilai 5 maka yang perlu kita lakukan pertama adalah menggenatuhi berada di indeks baris dan kolom berapa lokasi nilai yang ingin kita ambil.

Nilai 5 terlihat berada pada baris ke-1 dan kolom ke-1. Kok indeks-nya 1 (baris/kolom)? Seperti list indeks array juga dimulai dari nol. Lalu bagaimana jika kita ingim mengakses angka atau nilai 9 dari matriks?

Nilai 9 berada pada baris ke-3 dan kolom ke-3, karna indeks matrik dimulai dari nol maka berada pada indeks ke [2][2].

### Praktek 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:
```

### Hasil :



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

### Penjelasan :

baris 1 = komentar, memberi tahu bahwa ini program penjumlahan matriks dari list

baris 2 = mendefinisikan matriks X berisi 3 baris dan 3 kolom

baris 3 = mendefinisikan matriks Y berisi 3 baris dan 3 kolom

baris 4 = membuat matriks kosong 'result' dengan nilai awal semua 0, ukurannya sama dengan X dan Y

baris 5 = memulai perulangan untuk baris (loop luar), sebanyak jumlah baris di X

baris 6 = memulai perulangan untuk kolom (loop dalam), sebanyak jumlah kolom di X

baris 7 = menjumlahkan elemen dari X dan Y pada posisi [i][j], lalu disimpan ke result[i][j]

baris 8 = mencetak judul hasil penjumlahan

baris 9 = perulangan untuk mencetak setiap baris dari matriks hasil penjumlahan

baris 24 = mencetak baris dari matriks result

#### Praktek 14 :

```
# impor library numpy
import numpy as np

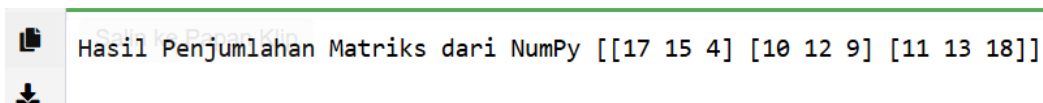
# Membuat matriks dengan numpy
X = np.array([
    [12,7,3],
    [4,5,6],
    [7,8,9]])

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 :



```
Hasil Penjumlahan Matriks dari NumPy [[17 15 4] [10 12 9] [11 13 18]]
```

#### Penjelasan :

baris 1 = mengimpor library numpy dengan alias np

baris 2 = membuat matriks X sebagai array NumPy 3x3

baris 3 = membuat matriks Y sebagai array NumPy 3x3

baris 4 = menjumlahkan matriks X dan Y langsung (karena NumPy mendukung operasi ini secara otomatis per elemen)

baris 5 = mencetak judul hasil penjumlahan

baris 6 = mencetak hasil penjumlahan matriks

#### praktek 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 :**

```
Hasil Pengurangan Matriks dari NumPy
[[ 7 -1  2]
 [-2 -2  3]
 [ 3  3  0]]
```

**Penjelasan :**

baris 1 = mengimpor library numpy dengan nama alias np

baris 2 = membuat matriks X dalam bentuk array NumPy 3x3

baris 3 = membuat matriks Y dalam bentuk array NumPy 3x3

baris 4 = melakukan operasi pengurangan: setiap elemen X dikurangi elemen yang sesuai di Y

baris 5 = mencetak judul hasil pengurangan

baris 6 = mencetak hasil pengurangan matriks

**Praktek 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 :**

```
Hasil Perkalian Matriks dari NumPy
[[60 56  3]
 [24 35 18]
 [28 40 81]]
>_
```

**Penjelasan :**

baris 1 = mengimpor library numpy dengan alias np

baris 2 = membuat matriks X sebagai array NumPy 3x3

baris 3 = membuat matriks Y sebagai array NumPy 3x3

baris 4 = melakukan operasi perkalian elemen-per-elemen (bukan perkalian matriks matematika!)

baris 5 = mencetak judul hasil perkalian

baris 6 = mencetak hasil perkalian elemen-per-elemen

**Praktek 17 :**



```

# 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]])

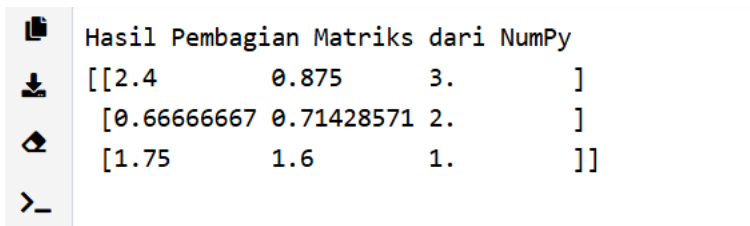
Y = np.array([
    [5,8,1],
    [6,7,3],
    [4,5,9]])

# Operasi pembagian dua matrik numpy
result = X / Y

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

```

**Hasil :**



```

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

```

**Penjelasan :**

baris 1 = komentar judul program (Praktek 17 - pembagian matriks dengan NumPy)

baris 2 = mengimpor library numpy dengan alias np

baris 3 = membuat matriks X dalam bentuk array NumPy 3x3

baris 4 = membuat matriks Y dalam bentuk array NumPy 3x3

baris 5 = melakukan pembagian elemen-per-elemen antara X dan Y ( $X[i][j] / Y[i][j]$ )

baris 6 = mencetak judul hasil pembagian

baris 7 = mencetak hasil pembagian matriks

## Praktek 18 :

```
# impor library numpy
import numpy as np

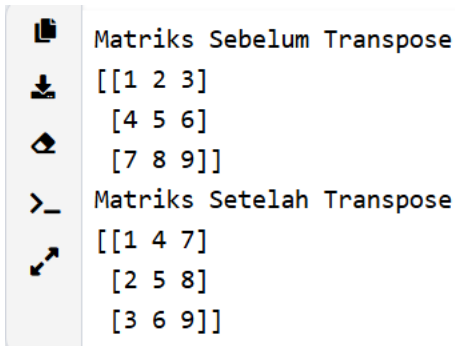
# membuat matriks
matriks_a = np.array([
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
])

# 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 :



```
Matriks Sebelum Transpose
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Matriks Setelah Transpose
[[1 4 7]
 [2 5 8]
 [3 6 9]]
```

## Penjelasan :

baris 1 = mengimpor library numpy dengan alias np

baris 2 = membuat matriks 3x3 bernama matriks\_a menggunakan np.array

baris 3 = mencetak teks "Matriks Sebelum Transpose"

baris 4 = mencetak isi dari matriks\_a

baris 5 = melakukan transpose (baris jadi kolom, kolom jadi baris) dan disimpan ke variabel balik

baris 6 = mencetak teks "Matriks Setelah Transpose"

baris 7 = mencetak isi dari matriks yang sudah ditranspose

## Praktek 19 :

```
# impor library numpy
import numpy as np

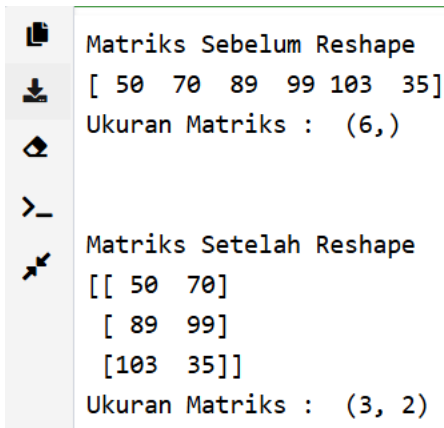
# membuat array 1 dimensi
arr_1d = np.array([50, 70, 89, 99, 103, 35])

# 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 :



The screenshot shows the output of a Jupyter Notebook. It displays the array before and after reshaping. The first part shows the 1D array [50, 70, 89, 99, 103, 35] and its shape (6,). The second part shows the 2D array after reshaping to (3, 2), which is displayed as three rows: [50, 70], [89, 99], and [103, 35].

```
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)
```

## Penjelasan :

baris 1 = mengimpor library numpy dengan alias np

baris 2 = membuat array 1 dimensi bernama arr\_1d dengan 6 elemen

baris 3 = mencetak teks "Matriks Sebelum Reshape"

baris 4 = mencetak isi dari arr\_1d (masih 1 dimensi)

baris 5 = mencetak ukuran (shape) dari arr\_1d → hasil: (6,)

baris 6 = baris kosong untuk pemisah tampilan

baris 7 = mengubah bentuk array menjadi matriks 3 baris x 2 kolom (reshape(3, 2)) dan disimpan ke variabel `ubah`

baris 8 = mencetak teks "Matriks Setelah Reshape"

baris 7 = mencetak isi matriks setelah diubah bentuknya jadi 3x2

baris 9 = mencetak ukuran (shape) dari matriks hasil reshape → hasil: (3, 2)

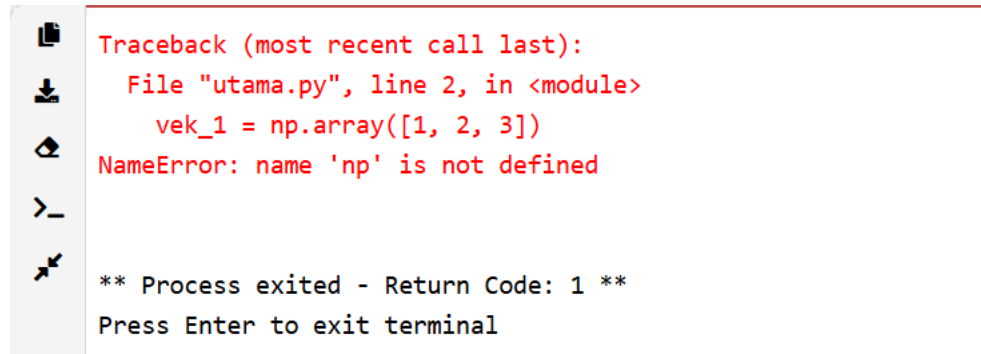
#### Praktek 20 :

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

# vektor kolom
vek_2 = np.array([1,
                  [2],
                  [3])
# atau menggunakan 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 :



```
Traceback (most recent call last):
  File "utama.py", line 2, in <module>
    vek_1 = np.array([1, 2, 3])
NameError: name 'np' is not defined

>_

** Process exited - Return Code: 1 **
Press Enter to exit terminal
```

#### Penjelasan :

**Sistem error**

#### Praktek 21 :

```

import numpy as np

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

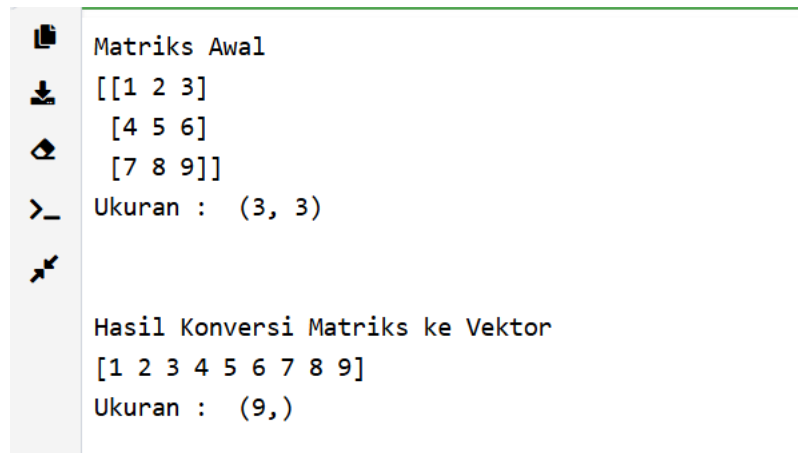
# cetak matriks awal
print("Matriks Awal")
print(matriks_a)
print("Ukuran : ", matriks_a.shape)
print("\n")

# 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 :**



```

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,)

```

**Penjelasan :**

baris 1 = mengimpor library numpy dengan alias np

baris 2 = membuat matriks 3x3 bernama matriks\_a menggunakan np.array

baris 3 = mencetak teks "Matriks Awal"

baris 4 = mencetak isi matriks\_a

baris 5 = mencetak ukuran (shape) dari matriks\_a → hasil: (3, 3)

baris 6 = baris kosong untuk pemisah tampilan

baris 7 = mengubah matriks menjadi vektor 1 dimensi dengan fungsi flatten(), disimpan ke jd\_vektor

baris 8 = mencetak teks "Hasil Konversi Matriks ke Vektor"

baris 9 = mencetak isi vektor hasil konversi

baris 10 = mencetak ukuran (shape) dari vektor → hasil: (9,)