

ARRAY

OLEH : ANDRI HERYANDI, M.T.



09

MATERI

01153 - Algoritma dan Struktur Data 1

Part #1

- Pengertian Array
- Dimensi Array
 - Array 1 Dimensi
 - Array Multidimensi
- Array 1 Dimensi
 - Pendeklarasian Array
 - Pengaksesan Array
 - Operasi-Operasi Dalam Array

Part #2

- Array Multidimensi
 - 2 Dimensi
 - Dimensi banyak
- Numpy (detail)
 - Operasi-operasi array dengan numpy



Oleh : Andri Heryandi, M.T.

ARRAY 1 DIMENSI

OLEH : ANDRI HERYANDI, M.T.



PENGERTIAN ARRAY

01153 - Algoritma dan Struktur Data 1

- Array adalah data terstruktur. Dalam Bahasa Indonesia disebut larik.
- Array adalah struktur data yang bisa menampung banyak data yang mempunyai tipe data yang sama tapi setiap elemennya boleh mempunyai nilai yang berbeda.
- Array menyediakan akses acak (random access) untuk setiap elemennya
- Setiap elemen array mempunyai index.
- Dengan array anda bisa menyimpan banyak data dalam sebuah variable.
- Elemen array disimpan di memori secara berurutan.
- Semua tipe data bisa disusun dalam bentuk array. Anda boleh memiliki array dari data integer, array dari data string, array dari data Boolean, atau array dari data pointer sekali pun. Bahkan anda pun bisa membuat array dari array.

Contoh Susunan Array

Index	[0]	[1]	[2]	[3]	[4]
Elemen/Isi	6	8	12	34	7

Index array biasanya berupa angka yang bisa dimulai dari angka berapa pun. Tetapi banyak Bahasa pemrograman (C, Java, Python) yang menggunakan index 0 sebagai index pertama. Oleh karena itu dalam materi perkuliahan ini juga akan menggunakan index 0 sebagai index pertama



PENGERTIAN ARRAY

01153 - Algoritma dan Struktur Data 1

■ Array dalam kehidupan sehari-hari

Mencari total dari 5 data.

$$Total = \sum_{i=1}^n D_i$$

D_i menyatakan “Data ke- i ” (data di posisi ke- i), ini berarti data yang mempunyai index (Array). Kalau $n=5$ maka i dimulai dari 1 s.d 5 sehingga Total : $D_1 + D_2 + D_3 + D_4 + D_5$

Mencari Standar Deviasi/Simpangan Baku.

$$s_x = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}}$$

X : Keseluruhan Data (Array)
 X_i : menyatakan “Data X index ke- i ”
 N : Banyaknya Data
 \bar{X} : Rata-Rata Data X

PENGERTIAN ARRAY

01153 - Algoritma dan Struktur Data 1

■ Array dalam kehidupan sehari-hari (Array 2 Dimensi)



Collaborative Filtering (System Recommendation)

- Tanda centang menandakan bahwa user tersebut menyukai film tersebut.
- Kategori pengguna bisa diklasifikasikan berdasarkan film yang disukai.

Apa yang bisa didapat dari sistem seperti ini?

- Rekomendasi Film Untuk Setiap User
- Target marketing promo film baru

Sumber : <https://developers.google.com/machine-learning/recommendation/images/1Dmatrix.svg>

DIMENSI ARRAY

01153 - Algoritma dan Struktur Data 1

- Array harus memiliki dimensi.
- Dimensi array paling sedikit adalah 1 dimensi. Sehingga anda bisa membuat :
 - Array 1 dimensi
 - Array multidimensi (2 dimensi, 3 dimensi atau lebih)

1D Array

3	2
---	---

2D Array

1	0	1
3	4	1

3D Array

1	7	9
5	9	3
7	9	9

Di dunia matematik, array 2 dimensi dikenal sebagai matriks



Sumber Gambar : https://miro.medium.com/max/1400/1*X0Dg7QfSYtWhSAu-afi8-g.png

Oleh : Andri Heryandi, M.T.

ARRAY 1 DIMENSI

OLEH : ANDRI HERYANDI, M.T.



PENDEKLARASIAN ARRAY (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

■ Pendefinisian Array 1 Dimensi

`NamaVariable : Array[indexawal.. Indexakhir] of typedata`

Contoh :

`Data : Array[0 .. 4] of integer`

Index	[0]	[1]	[2]	[3]	[4]
Isi	5	7	3	4	9



Oleh : Andri Heryandi, M.T.

PENDEKLARASIAN ARRAY (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

- **Konstanta penyimpan maksimal data array**
 - Terkadang nilai maksimal data array sering diperlukan untuk pemeriksaan (apakah sudah penuh atau belum), oleh karena itu sebaiknya anda menyimpan data ke sebuah nilai konstanta, misalnya maks.
- **Variable penyimpan kondisi banyaknya data dalam array**
 - Ketika sebuah array memiliki batas 100 elemen, belum tentu banyaknya data akan terisi 100. Ketika program dijalankan ada kemungkinan array mempunyai isi kosong (banyaknya 0 elemen), setelah ada penambahan data maka banyak data akan bertambah 1.
 - Oleh karena itu, ada baiknya setiap anda membuat array maka sertakan sebuah variable bertipe integer yang menyimpan kondisi berapa elemen data yang tersimpan di array tersebut.

Contoh :

Kanus :

```
const Maks=5
```

```
Data : Array[0 .. Maks-1] of integer
```

```
BanyakData : integer
```

Karena index array yang digunakan dimulai dari 0, maka jika banyak data ingin Maks elemen, maka ketika pendeklarasiannya nilai maksimal harus dikurangi 1



MENGAKSES ELEMEN ARRAY (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

■ Mengakses Elemen Array 1 Dimensi

- Mengakses elemen array dilakukan dengan memanggil nama variabelnya disertai dengan index-nya.
- Index ditulis diapit didalam kurung siku ([]).

■ Contoh Array :

Data : Array[0..4] of Integer;

Index	[0]	[1]	[2]	[3]	[4]
Isi	5	7	3	4	9

■ Cara mengakses elemen array:

- `Data[1] ← 7` // pengisian data pada index ke-1 dengan nilai 7
- `Input(Data[0])` // pengisian data pada index ke-0 dengan cara meminta masukkan dari pengguna.
- `Data[i] ← 100` // mengisi data pada index ke-i dengan nilai 100.
- `Data[4] ← Data[0] + Data[3]` // data pada index ke-4 diisi dengan hasil penjumlahan data index ke-0 dan ke-3.
- `Output(Data[2])` // menampilkan data pada index ke-2 ke layar.



REPRESENTASI ARRAY DALAM MEMORI (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

■ Bagaimana membayangkan struktur Array?

Vertikal

Index	Isi
[1]	5
[2]	7
[3]	3
[4]	4
[5]	9
[6]	8
[7]	7
[8]	5
[9]	2
[10]	6

Horizontal

Index	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
Isi	5	7	3	4	9	8	7	5	2	6

Cara anda membayangkan array sebenarnya tergantung kasus anda.

Contoh :

1. Jika anda menggunakan array sebagai stack (tumpukan) data, maka lebih baik membayangkannya secara vertikal,
2. Jika anda menggunakan array untuk queue (antrian), maka lebih baik anda membayangkannya secara horizontal.



ARRAY DI PYTHON

01153 - Algoritma dan Struktur Data 1

- **TIDAK ADA BUILT-IN TYPE ARRAY DI PYTHON**
- Jika anda ingin menggunakan array di python, setidaknya ada 3 cara yang bisa anda lakukan yaitu :
 - Menggunakan List sebagai array
 - Mengimport module array
 - Mengimport module numpy
- Ketiga cara di atas menggunakan array dinamis (dimana ukuran array akan berkembang sesuai dengan keperluan ketika program dijalankan). Berbeda dengan di Bahasa C, Java, Pascal yang ukuran array ditentukan sebelum program dijalankan.



PENDEFINISIAN ARRAY DI PYTHON MENGUNAKAN LIST

01153 - Algoritma dan Struktur Data 1

- Untuk membuat sebuah objek list yang akan diasumsikan sebagai array adalah dengan membuat objek yang ditulis diapit dengan tanda kurung siku ([])
- Tidak perlu melakukan import, karena list merupakan struktur data/class built-in dari python.
- Contoh :
 - `data = []` # membuat list/array kosong
 - `data = [24, 23, 27, 12]` # list yang sudah terisi 4 buah elemen
- Contoh Program :

```
data1 = []  
print("data 1 : ", data1)  
data2 = [24, 23, 27, 12]  
print("data 2 : ", data2)
```

Hasil eksekusi

```
data 1 :  []  
data 2 :  [24, 23, 27, 12]
```



PENDEFINISIAN ARRAY DI PYTHON MENGUNAKAN MODULE ARRAY (BUILT-IN)

01153 - Algoritma dan Struktur Data 1

- Module array dapat digunakan untuk mendefinisikan objek yang berisi array dari nilai-nilai dasar (karakter, bilangan bulat dan bilangan pecahan).
- Untuk membuat sebuah objek array menggunakan module array, langkahnya adalah :
 - Import module array
 - Buat objek array dengan menggunakan function array.
- Sintak function array.

```
array(typecode [, initializer])
```

- Keterangan :
 - typecode : Jenis tipe data (i : integer, l: long, f : float, d : double). Lebih lengkap lihat di <https://docs.python.org/3.8/library/array.html>. Tipe data ini menggunakan tipe data yang digunakan di Bahasa C.
 - initializer : Nilai awal array. Bersifat optional.



PENDEFINISIAN ARRAY DI PYTHON MENGUNAKAN MODULE ARRAY (BUILT-IN)

01153 - Algoritma dan Struktur Data 1

■ Contoh Program :

```
import array as ar
data1 = ar.array("i") # array integer
print("data 1 : ", data1)
data2 = ar.array("i", [24, 23, 27, 12]) # array int
print("data 2 : ", data2)
data3 = ar.array("f", [1.2, 2.5, 3.8]) # array float
print("data 3 : ", data3)
data4 = ar.array("d", [1.2, 2.5, 3.8]) # array double
print("data 4 : ", data4)
```

Hasil eksekusi

```
data 1 :  array('i')
data 2 :  array('i', [24, 23, 27, 12])
data 3 :  array('f', [1.2000000476837158, 2.5, 3.799999952316284])
data 4 :  array('d', [1.2, 2.5, 3.8])
```



PENDEFINISIAN ARRAY DI PYTHON MENGUNAKAN MODULE NUMPY

01153 - Algoritma dan Struktur Data 1

- Module numpy adalah :
 - Singkatan dari “numerical python”
 - Library python yang digunakan untuk bekerja dengan array.
- Untuk membuat sebuah objek array menggunakan module array, langkahnya adalah :
 - Import module numpy
 - Buat objek array dengan menggunakan function array.
- Sintak function array.

`array(object)`

- Keterangan :
 - *object* : Nilai awal array.



PENDEFINISIAN ARRAY DI PYTHON MENGUNAKAN MODULE ARRAY (BUILT-IN)

01153 - Algoritma dan Struktur Data 1

■ Contoh Program :

```
import numpy as np
data1 = np.array([])
print("data 1 : ", data1)
data2 = np.array([24, 23, 27, 12])
print("data 2 : ", data2)
data3 = np.array([1.2, 2.5, 3.8])
print("data 3 : ", data3)
data4 = np.array(["Aa", "Bebe", "Cece"])
print("data 3 : ", data4)
```

Hasil eksekusi

```
data 1 :  []
data 2 :  [24 23 27 12]
data 3 :  [1.2 2.5 3.8]
data 3 :  ['Aa' 'Bebe' 'Cece']
```



MENGAKSES ELEMEN ARRAY DI PYTHON

01153 - Algoritma dan Struktur Data 1

- Untuk mengakses array di python, gunakan nama objek dan indexnya.
- Index selalu dimulai dari 0

```
data = [ 10, 25, 30, 40]
print(data[1]) # elemen ke-2
data[2] = 75 # ubah isi elemen ke-3
print(data)
```

Hasil eksekusi

```
25
[10, 25, 75, 40]
```

```
import array as ar
data = ar.array("i",[ 10, 25, 30, 40])
print(data[1]) # elemen ke-2
data[2] = 75 # ubah isi elemen ke-3
print(data)
```

Hasil eksekusi

```
25
array('i', [10, 25, 75, 40])
```

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
print(data[1]) # elemen ke-2
data[2] = 75 # ubah isi elemen ke-3
print(data)
```

Hasil eksekusi

```
25
[10 25 75 40]
```



NUMPY ARRAY

01153 - Algoritma dan Struktur Data 1

- Pada perkuliahan ini, implementasi array yang akan digunakan adalah numpy array.
- Alasannya :
 - Numpy array lebih cepat dibandingkan list.
 - Numpy array menyediakan function-function dan operasi-operasi matematika yang lengkap
 - Numpy dijadikan basis dari module-module lain (seperti Pandas, SciPy)



Oleh : Andri Heryandi, M.T.

OPERASI-OPERASI DASAR ARRAY

01153 - Algoritma dan Struktur Data 1

■ Operasi-operasi dasar array

- Traversal
- Penambahan
- Penyisipan
- Penghapusan



Oleh : Andri Heryandi, M.T.


TRAVERSAL ARRAY (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

- Traversal array artinya menelusuri array dari elemen pertama sampai terakhir.
- Traversal bisa digunakan untuk banyak hal seperti :
 - Menampilkan seluruh elemen array
 - Mencari total data dari seluruh elemen
 - Melakukan pencarian nilai terkecil / terbesar
- Algoritma traversal array (pseudo-code)

Algoritma:

```
for i ← 0 to banyak_elemen_array-1 do  
    operasi data ke-i (data[i])  
End for
```



Index	[0]	[1]	[2]	[3]	[4]
Isi	5	7	3	4	9

TRAVERSAL ARRAY (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

- Algoritma traversal array (pseudo-code) – Menampilkan data ke layar

Algoritma:

```
for i  $\leftarrow$  0 to banyak_elemen_array-1 do  
    output(data[i])  
End for
```

- Algoritma traversal array (pseudo-code) – Menghitung total data

Algoritma:

```
total  $\leftarrow$  0  
for i  $\leftarrow$  0 to banyak_elemen_array-1 do  
    total  $\leftarrow$  total + data[i]  
End for
```



TRAVERSAL ARRAY (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Ada beberapa cara untuk melakukan traversal array dengan python, yaitu :
 - Menggunakan pengulangan for in
 - Menggunakan pengulangan for in dan range
 - Menggunakan pengulangan while
 - Menggunakan enumerate



Oleh : Andri Heryandi, M.T.

TRAVERSAL ARRAY (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Traversak array menggunakan pengulangan for in

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
for i in data:
    print(i)
```

Hasil eksekusi

```
10
25
30
40
```

- Keterangan :

- Setiap melakukan pengulangan for, maka nilai variable i akan berisi nilai dari elemen pertama dan elemen selanjutnya.



TRAVERSAL ARRAY (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Traversal array menggunakan pengulangan for in dan range

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
banyakElemen = data.size # atau len(data)
for i in range(banyakElemen):
    print(data[i])
```

Hasil eksekusi

```
10
25
30
40
```

- Keterangan :

- Pengulangan for digunakan untuk menghasilkan deret index yang akan diakses.



TRAVERSAL ARRAY (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Traversal array menggunakan pengulangan while

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
banyakElemen = len(data)
i = 0
while i < banyakElemen:
    print(data[i])
    i = i + 1
```

Hasil eksekusi

```
10
25
30
40
```



TRAVERSAL ARRAY (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Traversal array menggunakan enumerate().
 - Enumerate() digunakan untuk menghasilkan index (counter) dan nilai yang berada dalam array.

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
for i, isi in enumerate(data):
    print(i, ' : ', data[i])
```

Hasil eksekusi

0	:	10
1	:	25
2	:	30
3	:	40

PENAMBAHAN DATA (ALGORITMA)

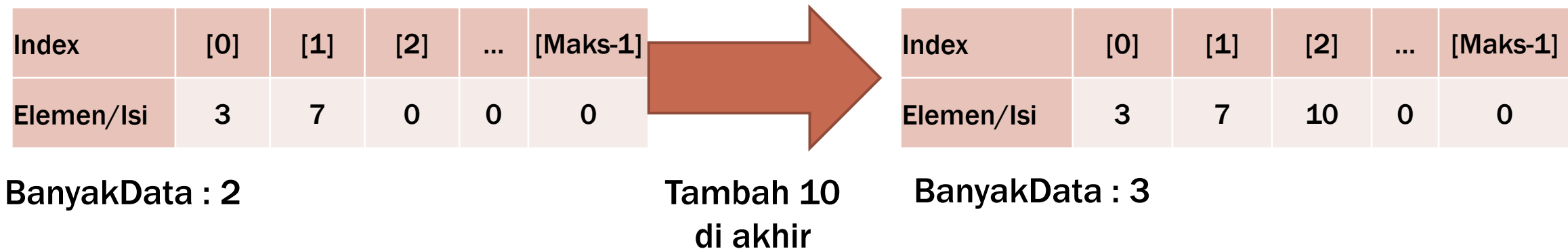
01153 - Algoritma dan Struktur Data 1

■ Penambahan data adalah proses menambahkan elemen baru di bagian akhir.

■ Algoritmanya :

1. Periksa apakah array belum penuh (banyak data < maks-1).
2. Jika array belum penuh maka lakukan :
 1. Banyak data ditambah 1
 2. Simpan data baru di posisi akhir ($\text{data}[\text{banyakdata}-1] \leftarrow \text{baru}$)

■ Ilustrasi penambahan data



PENAMBAHAN DATA (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

■ Algoritma pseudo-code procedure penambahan data

Program Demo_Penambahan_Data

Kamus:

```
const maks = 5  
data : array [0 .. Maks-1] of integer  
banyakdata : integer
```

Procedure tambah(input/output data:array of integer, input/output banyakdata, input baru:integer)

Algoritma:

```
if banyakdata < maks-1 then  
    banyakdata ← banyakdata + 1  
    data[banyakdata-1] ← baru  
else  
    output("data penuh")  
end if
```

EndProcedure

Algoritma (Program Utama) :

```
banyakdata ← 0  
tambah(data, banyakdata, 5) # tambahkan 5 di akhir array → [ 5 ]  
Tambah(data, banyakdata, 10) # tambahkan 10 di akhir array → [ 5, 10 ]
```



PENAMBAHAN DATA (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Untuk melakukan penambahan data menggunakan Bahasa python di numpy array adalah dengan memanggil method append.
- Method append akan menghasilkan sebuah array baru yang berisi array lama yang sudah ditambah dengan elemen baru di posisi akhir
- Contoh Program :

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
for i, isi in enumerate(data):
    print(i, ' : ', data[i])
```

Hasil eksekusi

0	:	10
1	:	25
2	:	30
3	:	40



PENYISIPAN DATA (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

- Penyisipan data adalah proses menambahkan elemen baru di bagian awal atau tengah.
- Algoritmanya :
 1. Periksa, apakah masih ada elemen kosong (belum penuh). Hal ini ditandai dengan nilai BanyakData < Maks-1.
 2. Jika masih ada elemen kosong maka lakukan langkah 3. Jika telah penuh tampilkan pesan bahwa array telah penuh.
 3. Periksa apakah posisi sisip berada di range 0 s.d BanyakData-1. Jika tidak, maka tampilkan pesan bahwa posisi sisip tidak sah, tetapi jika posisi sisip berada dalam range tersebut maka eksekusi langkah 4 s.d 6.
 4. Geserkan data dari posisi sisip sampai posisi terakhir ke posisi berikutnya (Data index ke-i disikan ke indek ke-(i+1).
 5. Tempatkan data baru di posisi setelah posisi sisip
 6. Variable BanyakData ditambah 1 (menyatakan bahwa data bertambah)



PENYISIPAN DATA (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

- Ilustrasi Penyisipan data di tengah (Contoh : Menambahkan elemen 15 di posisi 2)

Kondisi Awal

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	17	23	30	0	0	0

BanyakData : 5

Geserkan data

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	17	17	23	30	0	0

BanyakData : 5

Simpan data baru di posisi sisip

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	15	17	23	30	0	0

BanyakData : 6

PENAMBAHAN DATA (ALGORITMA)

01153 - Algoritma dan Struktur Data 1

■ Algoritma pseudo-code procedure penyisipan data

Program Demo_Penambahan_Data

Kamus:

```
const maks = 5
data : array [0 .. Maks-1] of integer
banyakdata : integer
```

Procedure sisip(input/output data:array of integer, input/output banyakdata,
input posisisisip, input baru:integer)

Kamus :

```
i : integer
```

Algoritma:

```
if banyakdata < maks-1 then
    if posisisisip >= 0 and posisisisip < banyakdata-1 then
        for i ← posisisisip to banyakdata-1 then banyakelemen
            data[i+1] ← data[i]
        data[posisisisip] ← baru
        banyakdata ← banyakdata + 1
    else
        output("Posisi sisip tidak sah")
    end if
else
    output("data penuh")
end if
```

EndProcedure

Algoritma (Program Utama) :

```
Data[0] = 5
Data[1] = 6
Data[2] = 7
Data[3] = 3
Banyakdata = 4
Sisip(data, banyakdata, 2, 99) # [5 6 99 7 3]
```



PENYISIPAN DATA (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Untuk melakukan penyisipan data menggunakan Bahasa python di numpy array adalah dengan memanggil method insert.
- Method insert akan menghasilkan sebuah array baru yang berisi array lama yang sudah ditambah dengan elemen baru di posisi sisip
- Contoh Program :

```
import numpy as np
data = np.array([ 10, 25, 30, 40])
print("Sebelum insert : ", data)
data = np.insert(data,2, 77) # sisip 77 di posisi 2
print("Setelah insert 77 di posisi 2 : ", data)
```

Hasil eksekusi

```
Sebelum insert : [10 25 30 40]
Setelah insert 77 di posisi 2 : [10 25 77 30 40]
```



PENGHAPUSAN DATA

01153 - Algoritma dan Struktur Data 1

- Penghapusan Data adalah proses menghilangkan sebuah elemen yang sudah ada dalam array.
- Ada 3 cara melakukan penghapusan data
 - Penghapusan data di posisi data awal (beban komputasi paling banyak karena melibatkan pergeseran sebanyak $\text{BanyakData} - 1$).
 - Penghapusan data di posisi data tengah (beban komputasi agak banyak karena melibatkan pergeseran)
 - Penghapusan data di posisi data akhir (beban komputasi paling ringan karena tidak ada pergeseran)



PENGHAPUSAN DATA (DELETE)

01153 - Algoritma dan Struktur Data 1

- Algoritma penghapusan data di posisi data di posisi tertentu adalah :
 1. Periksa apakah array memiliki elemen. Ini dilakukan dengan memeriksa variable BanyakData apakah lebih dari 0. Jika tidak (array masih kosong) maka tuliskan pesan bahwa “Data Kosong”, tetapi jika array tidak kosong (memiliki elemen atau $\text{BanyakData} > 0$) maka lakukan langkah 2 s.d. 5.
 2. Jika posisi hapus di posisi akhir ($\text{banyakdata} - 1$) maka lanjutkan ke langkah 4 dan 5. Tetapi jika penghapusan dilakukan di posisi sebelum posisi akhir (index 0 s.d $\text{banyakdata} - 1$) maka lakukan Langkah 3 (pergeseran data)
 3. Geserkan data di posisi setelah posisi hapus ($\text{posisi hapus} + 1$) sampai data terakhir ke posisi sebelumnya ($\text{Data}[i-1] := \text{Data}[i]$).
 4. Isi data di posisi akhir dengan nilai default (misalnya 0). Langkah ini sebenarnya langkah opsional, boleh dilakukan atau tidak.
 5. Banyak Data dikurangi 1.



PENGHAPUSAN DATA (DELETE)

01153 - Algoritma dan Struktur Data 1

- Ilustrasi hapus data di posisi akhir.

Kondisi Awal

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	17	23	30	0	0	0

BanyakData : 5



Hapus Data Di Posisi Data Akhir

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	17	23	0	0	0	0

BanyakData : 4

PENGHAPUSAN DATA (DELETE)

01153 - Algoritma dan Struktur Data 1

- Ilustrasi penghapusan data di tengah (Contoh : Menghapus data di posisi 2)

Kondisi Awal

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	17	23	30	0	0	0

BanyakData : 5

Geserkan data

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	23	30	30	0	0	0

BanyakData : 5

Defaultkan data terakhir, Banyak Data Kurangi 1

Index	[0]	[1]	[2]	[3]	[4]	[5]	...	[Maks-1]
Elemen/Isi	3	20	23	30	0	0	0	0

BanyakData : 4

PENGHAPUSAN DATA (DELETE)

01153 - Algoritma dan Struktur Data 1

■ Contoh procedure hapus data di posisi tengah

Program Demo Hapus Data

Kamus:

```
const maks = 5
data : array [0 .. Maks-1] of integer
banyakdata : integer
Procedure hapus(input/output data:array of integer,
               input/output banyakdata, input posisihapus)
```

Kamus :

```
i : integer
```

Algoritma:

```
if banyakdata>0 then
    if posisihapus>=0 and posisihapus<banyakdata-1 then
        for i ← posisihapus+1 to banyakdata-1 then banyakelemen
            data[i-1] ← data[i]
        end if
        data[banyakdata-1] = 0 # mendefaultkan nilai
        banyakdata ← banyakdata - 1
    else
        output("data kosong")
    end if
EndProcedure
```

Algoritma (Program Utama)

```
Data[0] ← 7
Data[1] ← 9
Data[2] ← 10
Data[3] ← 15
Data[4] ← 20
Banyakdata ← 5
Hapus(data,banyakdata,0) # [9, 10, 15, 20]
Hapus(data,banyakdata,2) # [9, 10, 20]
Hapus(data,banyakdata,2) # [9, 10]
```



PENGHAPUSAN DATA (PYTHON)

01153 - Algoritma dan Struktur Data 1

- Untuk melakukan penghapusan data menggunakan Bahasa python di numpy array adalah dengan memanggil method delete.
- Method delete akan menghasilkan sebuah array baru yang berisi salinan array lama yang sudah mengalami proses penghapusan.
- Contoh Program :

```
import numpy as np
data = np.array([4, 6, 8, 2, 5, 7])
print("Sebelum hapus : ", data)
data = np.delete(data,0) # hapus depan
print("Setelah hapus posisi 0 : ", data)
data = np.delete(data,2) # hapus tengah
print("Setelah hapus posisi 2 : ", data)
data = np.delete(data,3) # hapus akhir
print("Setelah hapus posisi 3 : ", data)
```

Hasil eksekusi

```
Sebelum hapus : [4 6 8 2 5 7]
Setelah hapus posisi 0 : [6 8 2 5 7]
Setelah hapus posisi 2 : [6 8 5 7]
Setelah hapus posisi 3 : [6 8 5]
```



MEMBUAT FUNCTION DENGAN ARRAY SEBAGAI PARAMETER

01153 - Algoritma dan Struktur Data 1

- Di python banyak sekali function-function yang menggunakan array sebagai parameternya.
 - Contoh :
 - Function sum, atau `numpy.sum`
 - Function `numpy.mean`
 - Function min, atau `numpy.min`
 - Function max, atau `numpy.max`
 - Function sorted, atau `numpy.sort`
 - Dan lain-lain
- Tetapi, selengkap apa pun function-function yang telah disediakan, terkadang kita membutuhkan function yang belum/tidak tersedia di function-function tersebut.



MEMBUAT FUNCTION DENGAN ARRAY SEBAGAI PARAMETER

01153 - Algoritma dan Struktur Data 1

■ Contoh :

```
import numpy as np

def my_sum(data):
    total = 0
    for i in data:
        total += i
    return total

def my_mean(data):
    return my_sum(data)/len(data)

def my_min(data):
    terkecil = data[0]
    for i in range(1,len(data)):
        if data[i]<terkecil:
            terkecil=data[i]
    return terkecil

def my_max(data):
    terbesar = data[0]
    for i in range(1,len(data)):
        if data[i]>terbesar:
            terbesar=data[i]
    return terbesar
```

```
# program utama
data = np.array([12, 45, 23, 12, 22])
print("Data : ", data)
print("Total : ", my_sum(data)," -- ", np.sum(data))
print("Rata-Rata : ", my_mean(data)," -- ", np.mean(data))
print("Terkecil : ", my_min(data)," -- ", np.min(data))
print("Terbesar : ", my_max(data)," -- ", np.max(data))
```

Hasil eksekusi

```
Data : [12 45 23 12 22]
Total : 114 -- 114
Rata-Rata : 22.8 -- 22.8
Terkecil : 12 -- 12
Terbesar : 45 -- 45
```



STANDAR DEVIASI

01153 - Algoritma dan Struktur Data 1

- Buatlah sebuah function untuk menghitung standar deviasi dari sekumpulan data.
- Rumus perhitungan standar deviasi adalah :

$$s_x = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}}$$

Keterangan :

- x adalah data
- n adalah banyak data
- x adalah rata-rata

```
import numpy as np

def my_std(data):
    # silahkan diisi functionnya

# program utama
data = np.array([12, 45, 23, 12, 22])
print("Data : ", data)
print("Standar Deviasi : ", my_dev(data), " -- ", np.std(data))
```

Hasil eksekusi

```
Data : [12 45 23 12 22]
Standar deviasi : 12.056533 -- 12.056533
```

PERKALIAN ARRAY

01153 - Algoritma dan Struktur Data 1

- Buatlah sebuah function untuk mengkalikan seluruh elemen array.
- Fungsi ini akan mereturnkan array yang berisi hasil perkalian seluruh elemen array dengan suatu nilai

```
import numpy as np

def my_multiply(data, pengkali):
    # silahkan isi functionnya

# program utama
data = np.array([12, 45, 23, 12, 22])
print("Data : ", data)
print("Data * 3: ", my_multiply(data, 3))
```

Hasil eksekusi

```
Data : [12 45 23 12 22]
Data * 3: [ 36. 135.  69.  36.  66.]
```



ARRAY MULTI DIMENSI

OLEH : ANDRI HERYANDI, M.T.



FORUM DISKUSI

01153 - Algoritma dan Struktur Data 1



Selamat Datang di LMS
UNIKOM

LMS UNIKOM merupakan media pembelajaran daring untuk memudahkan proses pengajaran di lingkungan Universitas Komputer Indonesia

LMS UNIKOM

<https://lms.unikom.ac.id>



**Group Whatsapp
Perkuliahahan**



Youtube Playlist

<https://unikom.id/YT-ASD1>



Oleh : Andri Heryandi, M.T.