

# Array

# Array

Array digunakan untuk:

- ❑ Menyimpan data-data yang diinputkan masing- masing kedalam memory komputer secara bersebelahan/homogen
- ❑ Ukuran atau jumlah elemen maksimum array telah diketahui dari awal yaitu ketika array dibuat.
- ❑ Sekali ukuran array ditentukan maka tidak dapat diubah. Ukuran array adalah bilangan bulat positif.
- ❑ Array harus diberi nama sebagai identifikasi
- ❑ Contoh kasus yang membutuhkan array:
  - ✓ Daftar pegawai perusahaan tertentu
  - ✓ Daftar nilai tes dari suatu matakuliah tertentu
  - ✓ Daftar customer dan nomor teleponnya

# Array

- ❑ Di dalam C dan pemrograman yang lain, terdapat suatu fasilitas untuk menyimpan data-data yang bertipe data sama dengan suatu nama tertentu = ARRAY/LARIK
- ❑ Array adalah suatu tipe data terstruktur yang berupa sejumlah data sejenis (bertipe data sama) yang jumlahnya tetap dan diberi suatu nama tertentu.
- ❑ Elemen-elemen array tersusun secara sekuensial di dalam memori sehingga memiliki alamat yang berdekatan/bersebelahan.
- ❑ Array dapat berupa array 1 dimensi, 2 dimensi, bahkan n- dimensi.
- ❑ Elemen-elemen array bertipe data sama tapi bisa bernilai sama atau berbeda-beda.

# Bentuk Array dalam Memory (int)

0	1	2	3	4	5	6	7	indeks
8	10	6	-2	11	7	1	100	value
ffea	ffeb	ffec	ffed	ffef	fffa	fffb	fffc	alamat

# Array (2)

- ❑ Elemen-elemen array dapat diakses oleh program menggunakan suatu indeks tertentu
- ❑ Pengaksesan elemen array dapat dilakukan berurutan atau random berdasarkan indeks tertentu secara langsung.
- ❑ Cara mengaksesnya adalah dengan menyebutkan nama array dan indeksinya. Indeks array dimulai dari 0 sampai dengan  $n-1$  ( $n$  adalah ukuran array).
- ❑ Dalam C, tidak terdapat error handling terhadap batasan nilai indeks, apakah indeks tersebut berada di dalam indeks array yang sudah didefinisikan atau belum.
- ❑ Hal ini merupakan tanggung jawab programmer.
- ❑ Di bahasa pemrograman lain: array index out of bounds exception

# Deklarasi

```
tipe_data nama_var_array[ukuran];
```

tipe\_data : menyatakan jenis tipe data elemen larik (int, char, float, dll)

nama\_var\_array : menyatakan nama variabel yang dipakai.

ukuran : menunjukkan jumlah maksimal elemen larik.

Tipe data sejenis

- Ada indeks yang teratur dan berurutan
- Bersifat statis, harus diketahui ukurannya terlebih dahulu

# Contoh dan Arti

```
char huruf[9];  
int umur[10];  
int kondisi[2] = {0,1}  
int arr_dinamis[] = {1,2,3}
```

- `char huruf[9]` berarti akan memesan tempat di memori komputer sebanyak 9 tempat dengan indeks dari 0-8, dimana semua elemennya bertipe data karakter semuanya. Kalau satu karakter berukuran 1 byte, berarti membutuhkan memori sebesar 9 byte.
- `int umur[10]`: berarti akan memesan tempat di memori komputer sebanyak 10 tempat dengan indeks dari 0-9, dimana semua elemennya bertipe data integer semuanya. Kalau satu integer berukuran 4 bytes, berarti membutuhkan memori sebesar  $4 \times 10 = 20$  bytes.

# Keunggulan dan Kelemahan Array

- ❑ Keunggulan array adalah sebagai berikut:

1. Array sangat cocok untuk pengaksesan acak. Sembarang elemen di array dapat diacu secara langsung tanpa melalui elemen-elemen lain.
2. Jika telah berada di suatu lokasi elemen, maka sangat mudah menelusuri ke elemenelemen tetangga, baik elemen pendahulu atau elemen penerus.

- ❑ Kelemahan array adalah sebagai berikut:

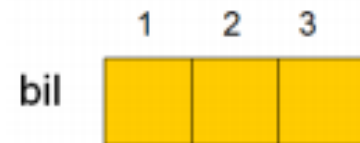
1. Array mempunyai fleksibilitas rendah, karena array mempunyai batasan harus bertipe homogen. Kita tidak dapat mempunyai array dimana satu elemen adalah karakter, elemen lain bilangan, dan elemen lain adalah tipe-tipe lain
2. Kebanyakan bahasa pemrograman mengimplementasikan array dengan ukuran statik yang sulit diubah ukurannya di waktu eksekusi. Bila penambahan dan pengurangan terjadi terus-menerus, maka representasi statis ini bersifat tidak efisien dalam penggunaan memori.



# Contoh dan Arti

Contoh penggunaan array dalam permasalahan sederhana adalah pengurutan 3 buah bilangan

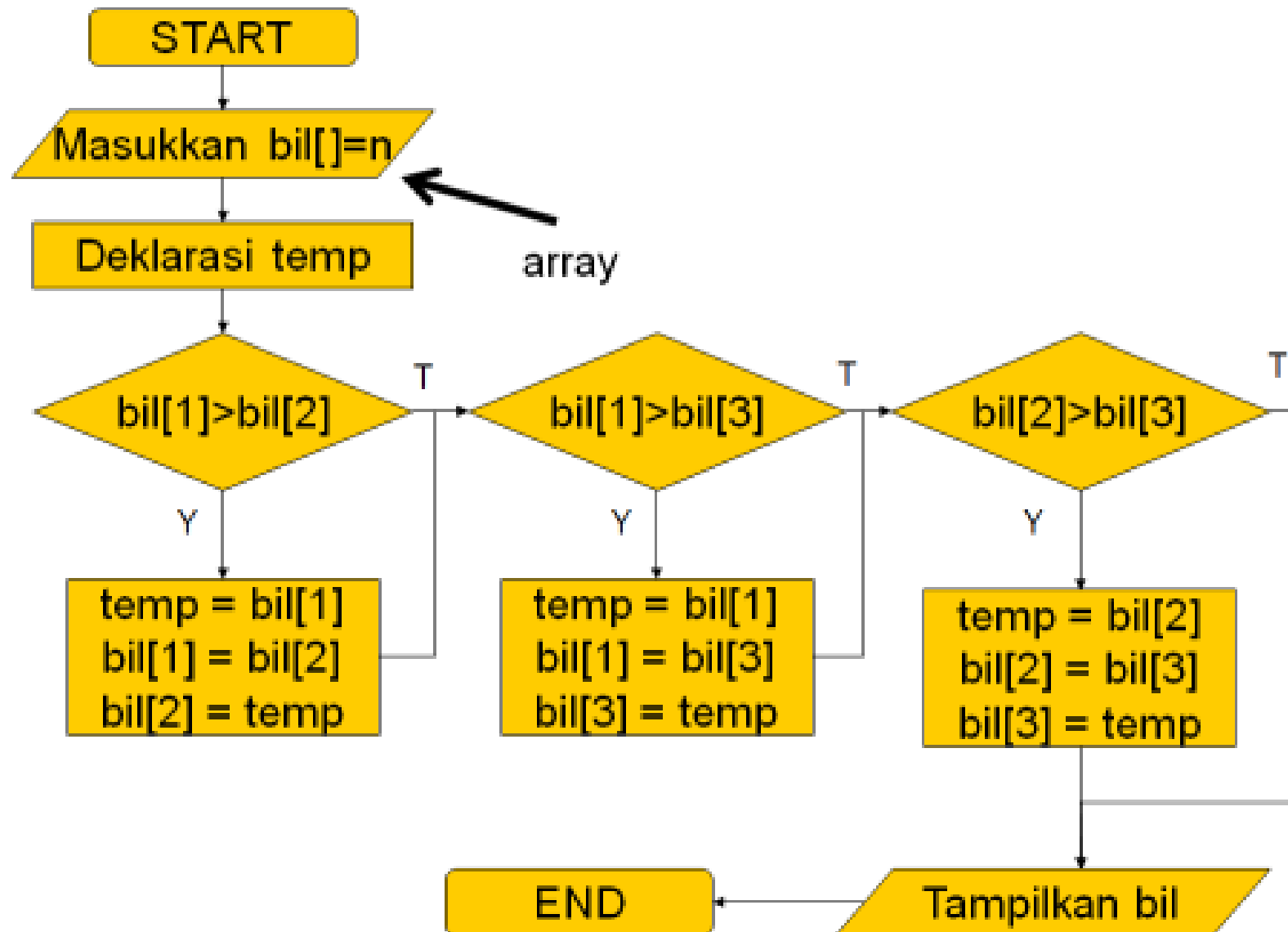
- Untuk mengurutkan tiga buah bilangan dibutuhkan operasi perbandingan yang menghasilkan kondisi benar atau salah ( $>$  atau  $<$ ).



Jika ( $\text{bil}[2] < \text{bil}[1]$ )  
temp = bil[1]  
bil[1] = bil[2]  
bil[2] = temp

Jika ( $\text{bil}[3] < \text{bil}[1]$ )  
temp = bil[1]  
bil[1] = bil[3]  
bil[3] = temp

Jika ( $\text{bil}[3] < \text{bil}[2]$ )  
temp = bil[2]  
bil[2] = bil[3]  
bil[3] = temp



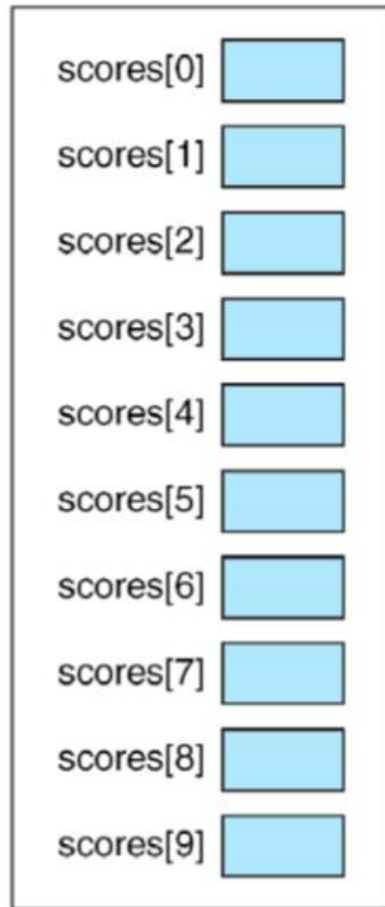
## Contoh dan Arti (2)

```
char huruf[9];  
int umur[10];  
int kondisi[2] = {0,1}  
int arr_dinamis[] = {1,2,3}
```

- `int kondisi[2]` berarti akan memesan tempat di memori komputer sebanyak 2 tempat dengan indeks 0-1, dimana semua elemennya bertipe data integer semuanya. Dan pada contoh di atas isi elemen-elemennya yang sebanyak 2 buah diisi sekaligus (diinisialisasi) yaitu pada elemen `kondisi[0]` bernilai 0, dan elemen `kondisi[1]` bernilai 1.
- `int arr_dinamis[]` berarti mendeklarasikan array dengan ukuran maksimum array tidak diketahui, namun ukuran tersebut diketahui berdasarkan inisialisasi yaitu sebanyak 3 elemen, yang isinya 1,2, dan 3. Kita tidak dapat mendeklarasikan array dinamis tanpa inisialisasi.

# Penjelasan Lebih Lanjut

- ❑ Tanda [] disebut juga “elemen yang ke- „. Misalnya “kondisi[0]“ berarti elemen yang ke nol.
- ❑ Array yang sudah dipesan, misalnya 10 tempat tidak harus diisi semuanya, bisa saja hanya diisi 5 elemen saja, baik secara berurutan maupun tidak.
- ❑ Namun pada kondisi yang tidak sepenuhnya terisi tersebut, tempat pemesanan di memori tetap sebanyak 10 tempat, jadi tempat yang tidak terisi tetap akan terpesan dan dibiarkan kosong.



scores

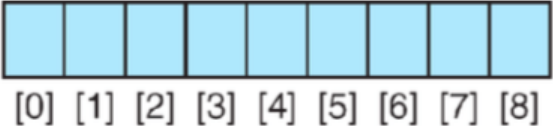
(b) Index Format

- 
- the subscript value in **square brackets**.
  - This is known as **indexing**

# Deklarasi dan Definisi Array

Declaration and definition tell the compiler the name of the array, the type of each element, and the number of elements(size) in the array.


`int scores [9];`



type of each element

scores

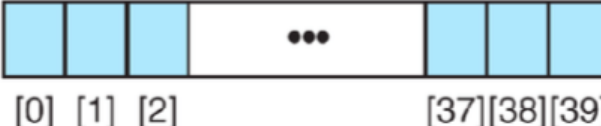
`char name [10];`



name of the array

name

`float gpa [40];`



number of elements

gpa

# Contoh Proses

## ALGORITMA

For Indeks ← 0 to N-1 do  
    Proses Array  
End For

- ✓ Mengisi elemen larik dengan 0 (inisialisasi)
- ✓ Mengisi elemen larik dari keyboard
- ✓ Mencetak elemen larik ke layar

A[indeks]=0

Input A[indeks]

Print A[indeks]

# Inisialisasi Array / Larik

## ALGORITMA

For Indeks  $\leftarrow$  0 to 7 do

A[indeks]=0

End For





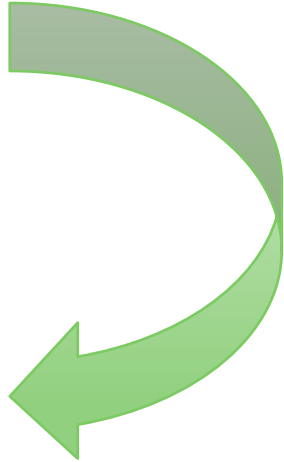
# Input Element Array / Larik

## ALGORITMA

For Indeks  0 to 7 do

    Scanf A[indeks]

End For



1	3	5	7	2	9	4	7
0	1	2	3	4	5	6	7

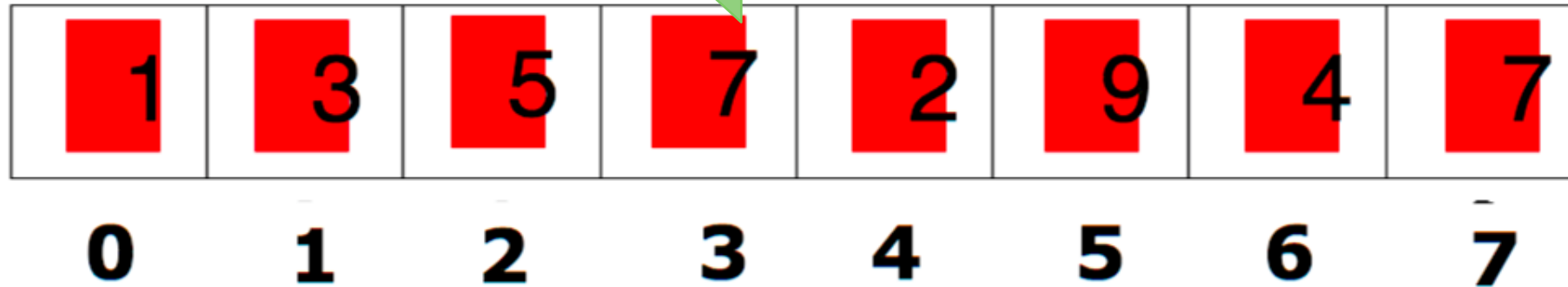
# Input Element Array / Larik

## ALGORITMA

For Indeks  $\leftarrow$  0 to 7 do

    Scanf A[indeks]

End For



# Input Element Array / Larik

## ALGORITMA

For Indeks  $\leftarrow$  0 to 7 do


    Printf A[indeks]

End For



1	3	5	7	2	9	4	7
0	1	2	3	4	5	6	7

# Array 1 Dimensi

- ▶ 

1	11	32	17	25	12	66
---	----	----	----	----	----	----
- ▶ Terdiri dari 1 baris, berisi beberapa data, semuanya memiliki tipe data yang sama

# Array 2 Dimensi

▶ P

12	17	22	14
10	5	13	5

- ▶ Terdiri lebih dari 1 baris dan 1 kolom, berisi beberapa data yang semuanya memiliki tipe data yang sama

# Array 2 Dimensi

- ▶ Terdiri dari baris dan kolom

	0	1	2	3
0	12	17	22	14
1	10	5	13	5

↓  
Baris, 2

→ Kolom, 4

Dimensi Array dinyatakan  
dalam Baris x Kolom

Array 2 x 4

## Array 2 Dimensi

12	17
10	5

12
10
17

12	17
10	5
17	11

1	11	32
---	----	----

# Deklarasi Array 2 Dimensi

**Tipe-data** **nama-array**[jumlah baris][jumlah kolom]

**tipe-data** : tipe data dari elemen array

**nama-array** : nama dari variabel array

**jumlah baris** : jumlah baris elemen array

**jumlah kolom** : jumlah kolom elemen array



# Deklarasi Array 2 Dimensi

► Contoh :

```
int arrayku[2][3];
```

→ Mendeklarasikan array berukuran 2x3, bertipe integer

	0	1	2
0			
1			

# Deklarasi Array 2 Dimensi

► Contoh :

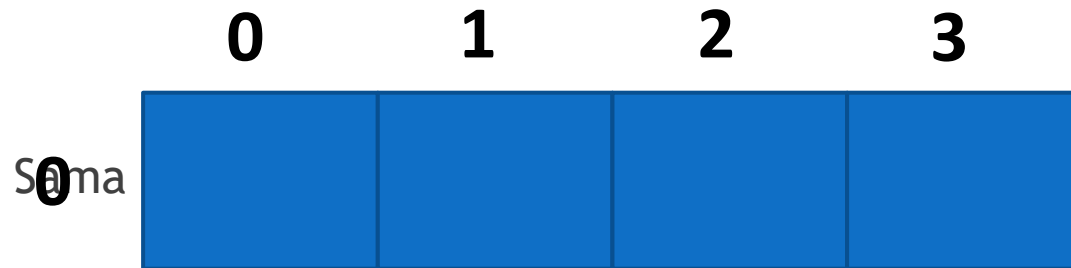
```
char peserta[10][50];
```

→ Mendeklarasikan array berukuran 10x50 bertipe char

# Deklarasi Array 2 Dimensi

► Contoh :

`int data[1][4];` → bagaimanakah array yang terbentuk ?



# Inisialisasi Array 2 Dimensi

- ▶ Inisialisasi bisa dilakukan saat variabel dideklarasikan
- ▶ Untuk Array 1 Dimensi, pemberian nilai dengan tanda '{ }'
- ▶ Dengan Array 2 Dimensi sama saja, hanya ada tambahan tanda '{ }' untuk masing-masing barisnya

# Inisialisasi Array 2 Dimensi

- ▶ Array 1 Dimensi :

```
int data[3] = {30, 40, 50};
```

30	40	50
----	----	----

- ▶ Array 2 Dimensi :

```
int data[2][3] = { {10,20,30}, {40,50,60} };
```

10	20	30
40	50	60

# Inisialisasi Array 2 Dimensi

```
int data[2][3] = { {10,20,30}, {40,50,60} };
```

Baris ke 0

Baris ke 1

	0	1	2
0	10	20	30
1	40	50	60

# Inisialisasi Array 2 Dimensi

- ▶ Jumlah baris dan kolom bisa tidak dicantumkan asalkan array langsung diinisialisasikan

```
int data[][] = { {10,20,30}, {40,50,60} };
```

→ Array berukuran 2x3, bertipe integer

# Inisialisasi Array 2 Dimensi

- Bagaimana deklarasi dan inisialisasi array-array berikut ?

30	40	50	30	40	50
----	----	----	----	----	----

10	20
40	50
10	20

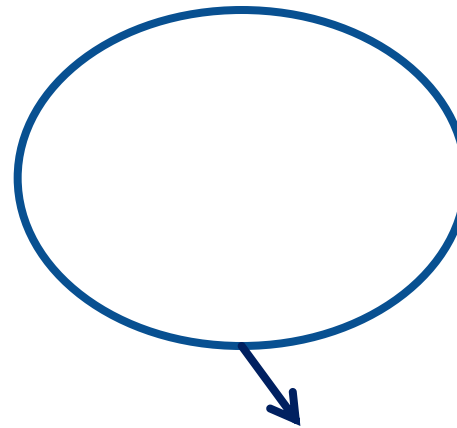
Z	X	c	V	B	N
A	S	D	F	G	H
q	W	e	R	t	Y



# Inisialisasi Array 2 Dimensi

- ▶ Bisa saja tidak seluruh elemen diinisialisasi
- ▶ Contoh :

```
int data[2][3] = { {3,2,3}, {3,4} }
```



**Kurang 1 elemen**

# Inisialisasi Array 2 Dimensi

- ▶ Jika ada beberapa elemen yang tidak diinisialisasi, maka isinya akan menjadi **NULL** atau karakter **\0**

```
int data[2][3] = { {3,2,3}, {3,4} }
```

3	2	3
3	4	NL

# Inisialisasi Array 2 Dimensi

- ▶ Benar atau salah inisialisasi berikut :
  - ▶ `int data[2][3] = { {10, 20, 30}};`
  - ▶ `int data[2][3] = { {10, 20, 30}, { } };`
  - ▶ `int data[2][3] = { {10, 20, 30}, {10} };`

# Inisialisasi Array 2 Dimensi

```
int data[2][3] = {{10, 20, 30}, {40, 50, 60}};
```

- ▶ Untuk mempermudah penulisan dan pembacaan, inisialisasi dapat dilakukan dengan penulisan berikut :

```
int data[2][3] = {{10, 20, 30},  
                  {40, 50, 60}};
```

# Inisialisasi Array 2 Dimensi

- Khusus untuk array 2 dimensi bertipe char, inisialisasi dapat dilakukan dengan cara-cara berikut :

```
char nama[2][6] = {{'m', 'a', 'r', 'k'},  
                  {'k', 'e', 'v', 'i', 'n'}};
```

```
char nama[2][6] = {"mark",  
                  "kevin"};
```

# Pengaksesan Array 2 Dimensi

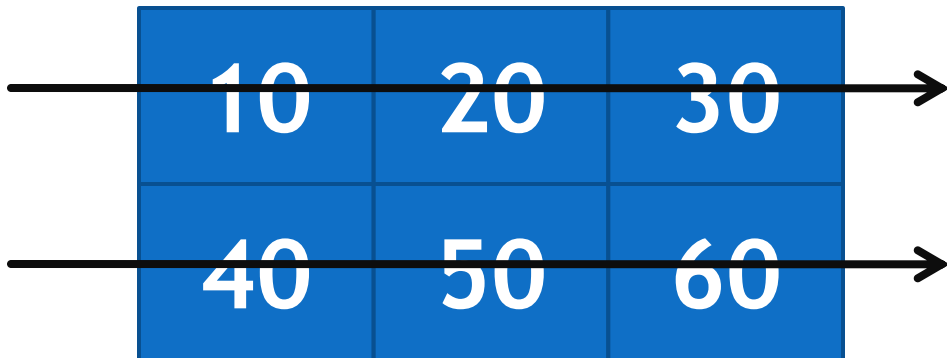
- ▶ Elemen dalam array 2 dimensi diakses dengan penanda baris dan kolom
- ▶ Contoh :

→ diakses dengan : `data[0][1];`

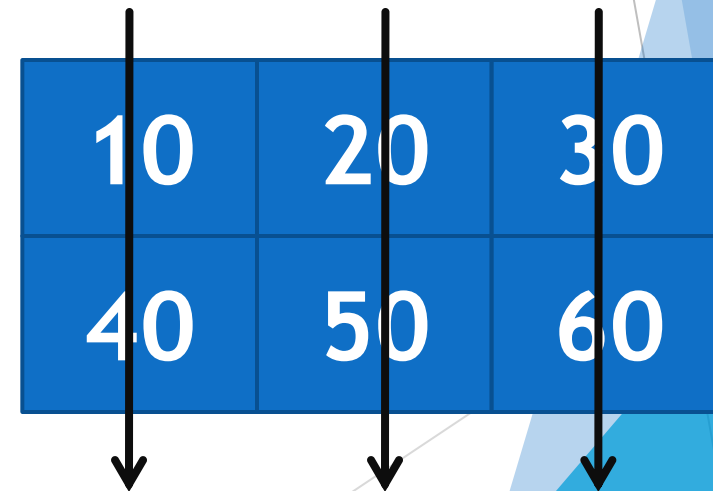
	0	1	2
0	10	20	30
1	40	50	60

# Pengaksesan Array 2 Dimensi

- ▶ Urutan pengaksesan tidak harus baris-per-baris, tapi bisa kolom-per-kolom sesuai kebutuhan
- ▶ Baris-per-baris :            Kolom-per-kolom



10	20	30
40	50	60



10	20	30
40	50	60