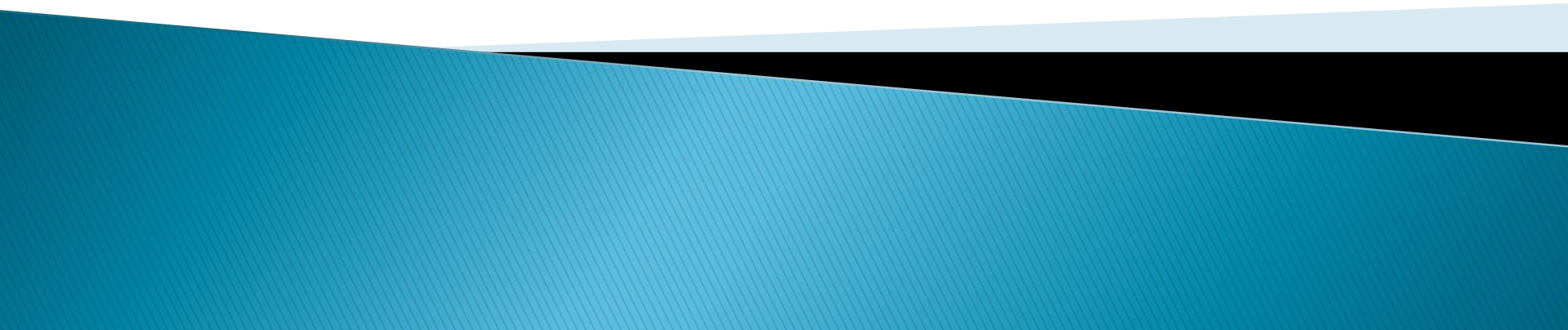


# Array Multi Dimensi

Algoritma



# Multidimension Array

Dimungkinkan untuk memiliki array dengan dimensi lebih dari satu.

Setiap dimensi direpresentasikan oleh subscript. Array dua dimensi memiliki dua subscript, 3 dimensi memiliki 3 subscript, dst.

Array dapat berdimensi berapapun, tapi umumnya digunakan satu atau dua dimensi.

```
tipe_data nama_array [ukuran 1][ukuran 2] ... [ukuran N];
```

# Array Multi Dimensi

- ▶ Array multi dimensi adalah suatu array yang mempunyai lebih dari satu subskrip.
- ▶ Array multi dimensi ini aplikasinya antara lain untuk matrik.
- ▶ Array berdimensi banyak pada kenyataannya jarang dipergunakan dalam aplikasi. Array berdimensi banyak yang sering digunakan adalah array dengan 2 dimensi atau lebih dikenal dengan nama **Matriks**

# Cara Pendeklarasian Array Multi Dimensi

- ▶ Cara pendeklarasian array multi dimensi mirip dengan cara array 1 dimensi.
- ▶ Hanya terdapat penambahan tanda kurung siku (“[“ dan “]”) untuk menunjukkan jumlah maksimum data yang dapat ditampung oleh variabel array tersebut.

Pada C/C++, untuk mendeklarasikan variable array multi dimensi kita dapat menuliskannya sebagai berikut :

**Type\_Data Nama\_Array[Jumlah\_Elemen1] ...[Jumlah\_Elemen n];**

Contoh :

```
int Array[10][5]; //deklarasi array 2 dimensi
```

```
double Jumlah[7][1][3]; //deklarasi array 3 dimensi
```

```
float Total[5][6][1][2]; //deklarasi array 4 dimensi, dst...
```

# Cara Pendeklarasian Array Tak Berukuran

- ▶ Ada kalanya kita tidak mengetahui jumlah elemen maksimum (atau dengan kata lain jumlah elemen dalam array sifatnya tidak konstan atau dinamis), untuk keperluan inilah dalam bahasa C/C++ kita bisa mendefinisikan suatu array tanpa mencantumkan berapa ukuran atau jumlah elemen maksimal yang bisa disimpan dalam array tersebut

# Cara Pendeklarasian Array Tak Berukuran

Pada C/C++, untuk mendeklarasikan variable array tak berukuran kita dapat menuliskannya sebagai berikut :

**Tipe\_Data Nama\_Array[ ][ ]...[ ];**

Contoh :

```
int Array[ ]; //deklarasi array 1 dimensi tak berukuran
```

```
char Angka[ ][ ]; //deklarasi array 2 dimensi tak berukuran
```

```
float Total[ ][ ]...[ ]; //deklarasi array dimensi tertentu dan tak berukuran
```

# Cara Akses Array Multi Dimensi

- ▶ Untuk dapat memasukkan suatu nilai atau melihat isi dari suatu array kita harus menentukan posisi dimana nilai tersebut disimpan. Untuk mengakses elemen array dapat kita lakukan dengan perintah

Pada C/C++, untuk mengakses elemen array dapat kita lakukan dengan perintah sebagai berikut :

**Nama\_Array[indeks\_Elemen1] ...[indeks\_Elemen n];**

Contoh :

Ary[10]; //akses elemen array Ary pada indeks ke-10

Jumlah[7][1]; // akses elemen array Jumlah pada indeks baris ke-7 dan indeks kolom ke-1

Total[5][6][1]; // akses elemen array Total pada indeks x ke-5 dan indeks y ke-6 dan indeks z ke-1, dst...

- ▶ Setelah posisi ini kita ketahui, kemudian kita bisa melakukan operasi pada array atau nilai dalam array tersebut. Operasi yang dapat dilakukan pada sebuah array sangat beragam tergantung kebutuhan pengguna program tersebut.

# Array 2 Dimensi

- ▶ Terdiri dari baris dan kolom

	0	1	2	3	→ Kolom, 4
0	12	17	22	14	Dimensi Array dinyatakan dalam Baris x Kolom
1	10	5	13	5	

↓  
Baris, 2

Array 2 x 4



# 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

► Contoh :

int matrik[2][4]; Merupakan matrik 2 X 4.

# 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

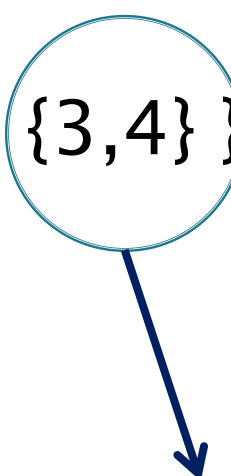
```
int data[][] = { {10,20,30}, {40,50,60} };
```

→ Array berukuran 2x3, bertipe integer

# Inisialisasi Array 2 Dimensi

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

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

A blue circle highlights the second row of the array initialization, {3,4}. A blue arrow points from this circle down to the text 'Kurang 1 elemen'.

**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

- ▶ 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 :

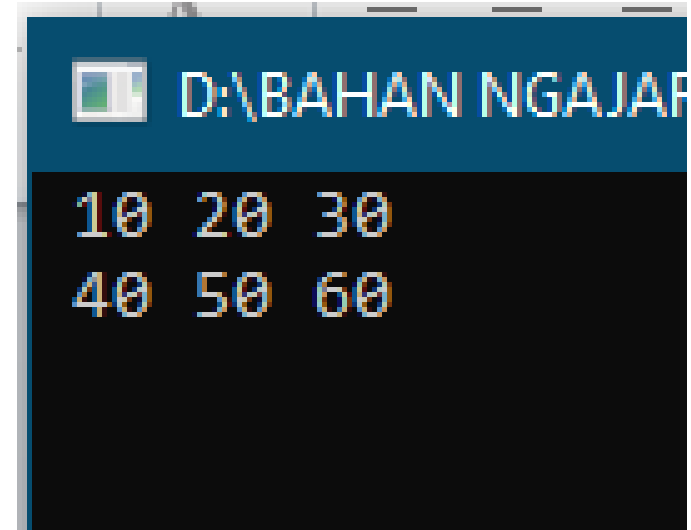
→ diakses dengan : `data[0][1];`

	0	1	2
0	10	20	30
1	40	50	60

# Contoh Array 2 Dimensi- Tipe Data Angka

```
• #include <stdio.h>
• #include <iostream.h>
• #include <conio.h>

• void main()
• {
•     int data1[2][3] = {{10, 20, 30},{40, 50, 60}};
•     for(int b=0; b<2; b++)
•     {
•         for(int k=0; k<3; k++)
•         {
•             cout<<" "<<data1[b][k];
•         }
•         cout<<endl;
•     }
•     cout<<endl;
•     getch();
• }
```



```
D:\BAHAN NGAJAR
10 20 30
40 50 60
```

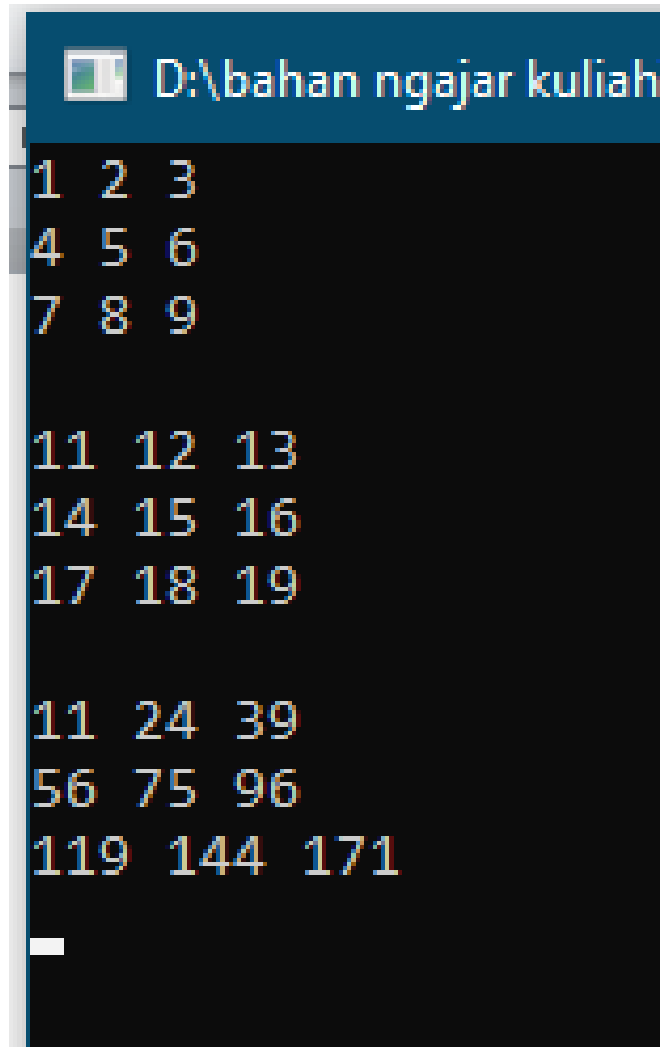
# Contoh Perkalian Matriks

- `#include <stdio.h>`
- `#include <conio.h>`
- `#include <iostream.h>`
- `void main() {`
- `int matriksA[3][3] = {{1,2,3},{4,5,6},{7,8,9}};`
- `int matriksB[3][3] =`  
`{{11,12,13},{14,15,16},{17,18,19}};`
- `int matriksC[3][3];`
- `for(int b=0; b<3; b++)`
- `{`
- `for(int k=0; k<3; k++)`
- `{`
- `cout<<matriksA[b][k]<<" ";`
- `}`
- `cout<<endl;`
- `}`
- `cout<<endl;`
- `}`

```
for(int b=0; b<3; b++)
{
    for(int k=0; k<3; k++)
    {
        cout<<matriksB[b][k]<<" ";
    }
    cout<<endl;
    for(int b=0; b<3; b++)
    {
        for(int k=0; k<3; k++)
        {
            matriksC[b][k]=matriksA[b][k]*matriksB[b][k];
            cout<<matriksC[b][k]<<" ";
        }
        cout<<endl;
    }
    getch();
}
```

# Contoh Perkalian Matriks

- Output



A screenshot of a Windows command prompt window. The title bar is dark blue with a small icon and the text "D:\bahan ngajar kuliah". The command prompt has a black background with yellow text. It displays the output of a matrix multiplication program. The first matrix is a 3x3 matrix with values 1, 2, 3; 4, 5, 6; 7, 8, 9. The second matrix is a 3x3 matrix with values 11, 12, 13; 14, 15, 16; 17, 18, 19. The resulting 3x3 matrix is displayed below, with values 11, 24, 39; 56, 75, 96; 119, 144, 171. A white cursor is visible at the bottom left of the command prompt.

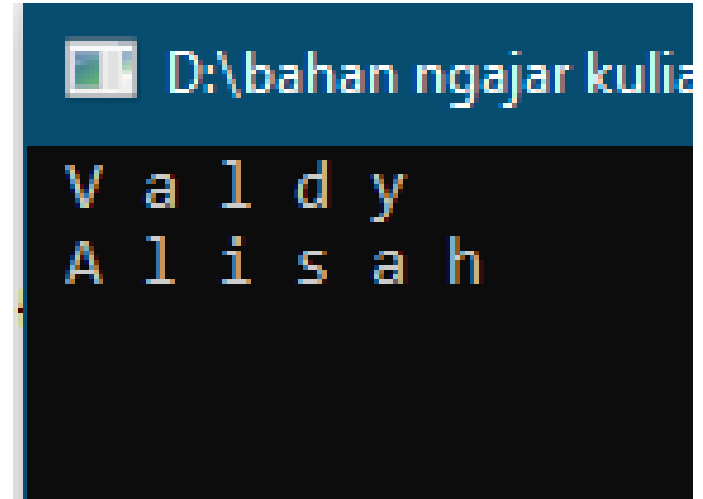
```
D:\bahan ngajar kuliah
1 2 3
4 5 6
7 8 9

11 12 13
14 15 16
17 18 19

11 24 39
56 75 96
119 144 171
_
```

# Contoh Array 2 Dimensi- Tipe Data Char

- `#include <stdio.h>`
- `#include <iostream.h>`
- `#include <conio.h>`
- `void main()`
- `{`
- `char data1[2][6] = {{'V','a','l','d','y'},`
- `{'a','l','i','s','a','h'}};`
- `for(int b=0; b<2; b++)`
- `{`
- `for(int k=0; k<6; k++)`
- `{`
- `cout<<" "<<data1[b][k];`
- `}`
- `cout<<endl;`
- `}`
- `cout<<endl;`
- `getch();`
- `}`

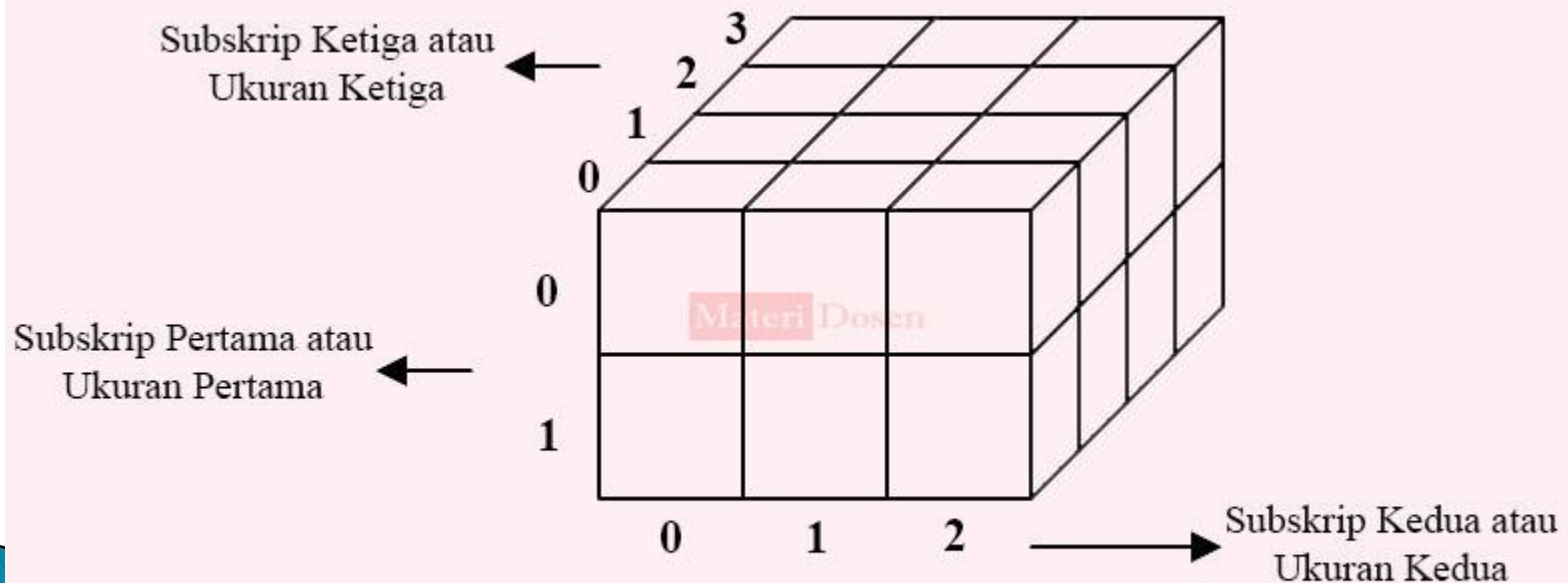


D:\bahan ngajar kuliah

V a l d y  
A l i s a h

# Array 3 Dimensi

- ▶ Digunakan untuk mengolah data dalam bentuk 3 dimensi
- ▶ Cth : `int LARIK [2][3][4];`
- ▶ Menentukan jumlah elemen =  $(2) \times (3) \times (4) = 24$  elemen



**Ilustrasi Array Tiga dimensi dengan 24 elemen**

## Contoh Array 3 Dimensi- Tipe Data Angka

```
#include <iostream.h>
```

```
#include <conio.h>
```

```
int main(){
```

```
// Deklarasi array tiga dimensi dengan nama "angka"
```

// Dengan jumlah ukuran pertama / subskrip pertama = 2

// Jumlah Ukuran kedua = 3 & jumlah ukuran ketiga = 4

```
int angka [2][3][4];
```

```
// Mendeklarasi variabel untuk indeks perulangan
```

```
int i,j,k;
```

```
cout<<"\t=====";
```

```
cout<<"\n\t== Contoh Array Tiga Dimensi ==\n";
```

```
cout<<"\t===== \n\n";
```

```
// Mengisi nilai kedalam elemen-elemen array angka
```

```
cout<<"== Masukkan elemen-elemen array angka ==\n";
```

```
for(i=0;i<2;i++){
```

```
for(j=0;j<3;j++){
```

```
for(k=0;k<4;k++){
```

```
cout<<"angka indeks ke ["<<i<<"]["<<j<<"]["<<k<<"]"<<" = ";
```

```
cin>>angka[i][j][k];
```

}

}

}

```
cout<<"\n\n=====\n";
```

```
cout<<"== Tampil nilai elemen Array ==\n";
```

```
cout<<"=====\\n";
```

```
//menampilkan nilai dari setiap elemen array angka
```

```
for(i=0;i<2;i++){
```

```
for(j=0;j<3;j++){
```

```
for(k=0;k<4;k++){
```

```
cout<<"angka indeks ke ["<<i<<"]["<<j<<"]["<<k<<"]<<" =
"<<angka[i][j][k]<<endl;
```

}

}

}

```
getch();
```

}

# Contoh Array 3 Dimensi

- Output

```
C:\BC5\BIN\NONAME00.exe

=====
== Contoh Array Tiga Dimensi ==
=====

== Masukkan elemen-elemen array angka ==
angka indeks ke [0][0][0] = 1
angka indeks ke [0][0][1] = 2
angka indeks ke [0][0][2] = 3
angka indeks ke [0][0][3] = 4
angka indeks ke [0][1][0] = 5
angka indeks ke [0][1][1] = 6
angka indeks ke [0][1][2] = 7
angka indeks ke [0][1][3] = 8
angka indeks ke [0][2][0] = 9
angka indeks ke [0][2][1] = 10
angka indeks ke [0][2][2] = 11
angka indeks ke [0][2][3] = 12
angka indeks ke [1][0][0] = 13
angka indeks ke [1][0][1] = 14
angka indeks ke [1][0][2] = 15
angka indeks ke [1][0][3] = 16
angka indeks ke [1][1][0] = 17
angka indeks ke [1][1][1] = 18
angka indeks ke [1][1][2] = 19
angka indeks ke [1][1][3] = 20
angka indeks ke [1][2][0] = 21
angka indeks ke [1][2][1] = 22
angka indeks ke [1][2][2] = 23
angka indeks ke [1][2][3] = 24
```

```
C:\BC5\BIN\NONAME00.exe

=====
== Tampil nilai elemen Array ==
=====

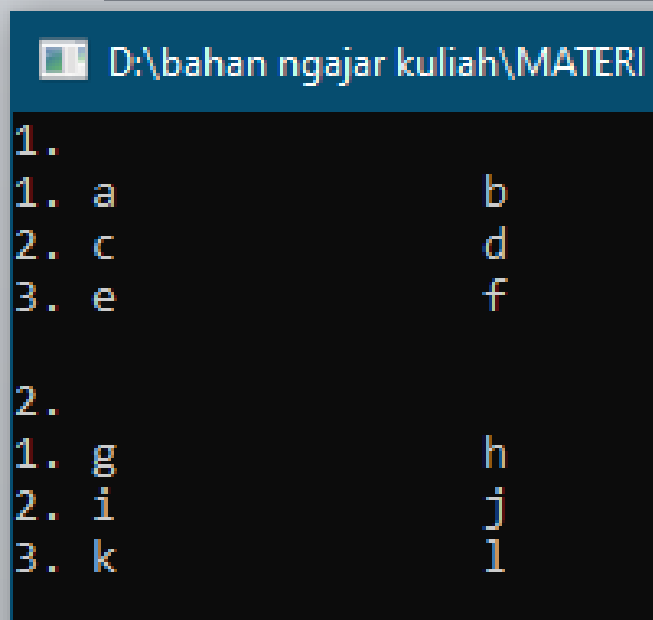
angka indeks ke [0][0][0] = 1
angka indeks ke [0][0][1] = 2
angka indeks ke [0][0][2] = 3
angka indeks ke [0][0][3] = 4
angka indeks ke [0][1][0] = 5
angka indeks ke [0][1][1] = 6
angka indeks ke [0][1][2] = 7
angka indeks ke [0][1][3] = 8
angka indeks ke [0][2][0] = 9
angka indeks ke [0][2][1] = 10
angka indeks ke [0][2][2] = 11
angka indeks ke [0][2][3] = 12
angka indeks ke [1][0][0] = 13
angka indeks ke [1][0][1] = 14
angka indeks ke [1][0][2] = 15
angka indeks ke [1][0][3] = 16
angka indeks ke [1][1][0] = 17
angka indeks ke [1][1][1] = 18
angka indeks ke [1][1][2] = 19
angka indeks ke [1][1][3] = 20
angka indeks ke [1][2][0] = 21
angka indeks ke [1][2][1] = 22
angka indeks ke [1][2][2] = 23
angka indeks ke [1][2][3] = 24
```



# Contoh Array 3 Dimensi- Tipe Data Char

- `#include <iostream.h>`
- `#include <conio.h>`
- `#include <stdio.h>`
- `void array_huruf()`
- `{`
- `char`
- `huruf[2][3][2]={{'a','b'},{'c','d'},{'e','f'},{'g','h'},{'i','j'},{'k','l'}};`
- `//Memanggil array`
- `for(int i=0;i<=1;i++)`
- `{`
- `cout<<i+1<<".\n";`
- `for(int j = 0; j<=2; j++)`
- `{`
- `cout<<j+1<<". ";`
- `for(int k = 0; k<=1; k++)`
- `{`
- `cout<<" "<<huruf[i][j][k]<<"\t\t ";`
- `}`
- `cout<<endl;`
- `}`
- `cout<<endl;`
- `}`
- `}`

```
int main()
{
    array_huruf();
    getch();
}
```



```
D:\bahan ngajar kuliah\MATERI P
1.
1. a b
2. c d
3. e f
2.
1. g h
2. i j
3. k l
```