

ARRAY

Pemrograman

C++

Dasar



LARIK (ARRAY)

- Array disebut juga larik, merupakan kumpulan dari nilai-nilai data bertipe sama dalam urutan tertentu yang menggunakan sebuah nama yang sama
- Elemen-elemen array tersusun secara sekuensial di dalam memori sehingga memiliki alamat yang berdekatan.
- Sifat array :
 - Homogen
Seluruh elemen di dalam struktur array mempunyai tipe data yang sama.
 - Random Access
Setiap elemen di dalam struktur array dapat dicapai secara individual, langsung ke lokasi elemen yang diinginkan, tidak harus melalui elemen pertama.

Array / Larik

- Nilai-nilai data di suatu array disebut dengan elemen-elemen array
- Letak urutan dari elemen-elemen array ditunjukkan oleh suatu *subscript* atau indeks
- Setiap komponen (elemen) array dapat dibedakan dan diakses melalui nomor indeksnya.
- Elemen-elemen dalam array dengan **n elemen** memiliki index dari **0** sampai **n-1**.
- Perhatikan bahwa tidak ada elemen array larik[**n**], karena hal ini akan menyebabkan **array-index-out-of-bounds exception** .

0	A[0] = 100	0x7ffdb0f72130
1	A [1] = 101	0x7ffdb0f72134
2	A [2] = 102	0x7ffdb0f72138
3	A [3] = 103	0x7ffdb0f7213
4	A [4] = 104	0x7ffdb0f72140

index

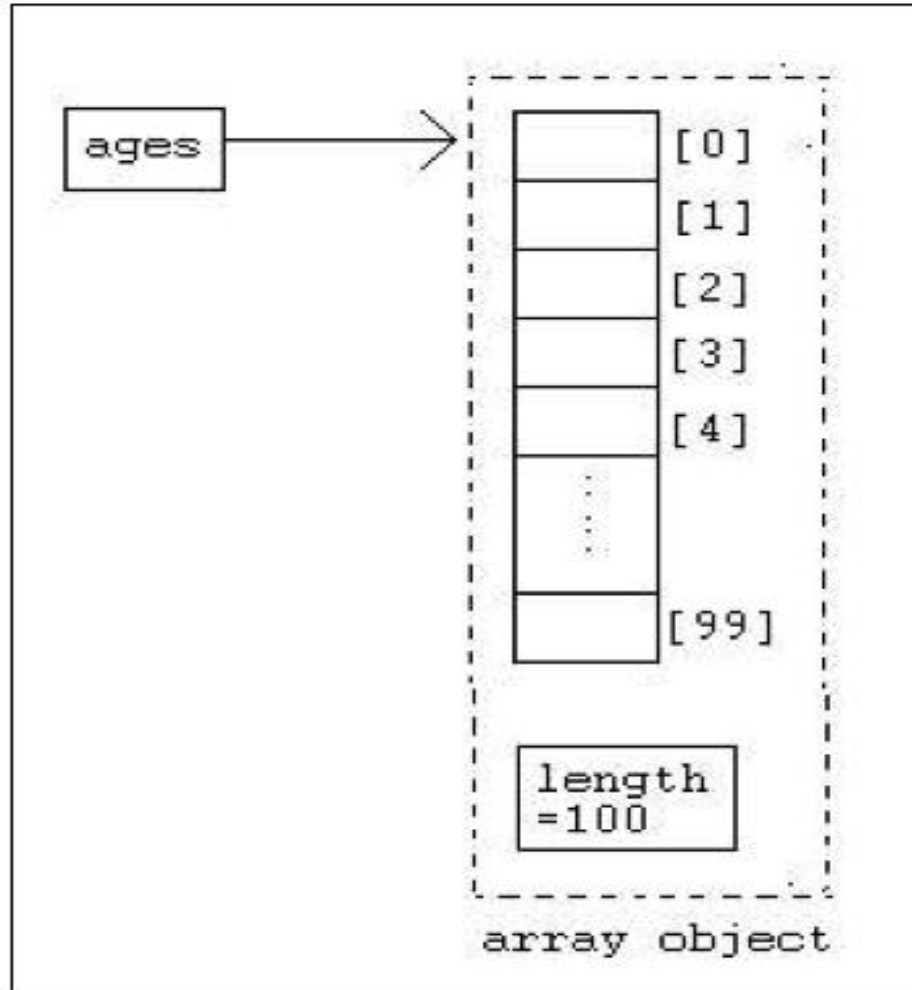
alamat

Nilai elemen atau values

Array

- Array merupakan struktur data yang statis, yaitu jumlah elemen yang ada harus ditentukan terlebih dahulu dan tak bisa di ubah saat program berjalan
- Elemen-elemen array dapat diakses secara berurutan atau random berdasarkan indeks tertentu secara langsung.
- Pengisian dan pengambilan nilai pada indeks tertentu dapat dilakukan dengan mengeset nilai atau menampilkan nilai pada indeks yang dimaksud.

LARIK (ARRAY)



Array

- Dalam bahasa pemrograman, 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.
- Jika programmer mengakses indeks yang salah, maka nilai yang dihasilkan akan berbeda atau rusak karena mengakses alamat memori yang tidak sesuai.
- Tanda [] disebut juga “elemen yang ke- “
- Misalnya “Murid[0]” berarti elemen yang ke nol dari array Murid.

- 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.
- Struktur array dapat digolongkan menjadi:
 - Array berdimensi satu disebut juga List, Vektor
 - Array berdimensi dua disebut juga Tabel, Matriks
 - Array berdimensi banyak

Array merupakan :

- Struktur Data paling mudah
- Struktur Data Statis yang harus diketahui ukurannya terlebih dahulu
- Memori ekonomis, bila semua elemen terisi
- Waktu akses sama ke setiap elemen
- Boros memori jika banyak elemen yang tidak digunakan



Array Satu Dimensi

Deklarasi Array

- Struktur array satu dimensi dapat dideklarasikan dengan bentuk umum berupa:

```
Tipe nama_larik [ukuran];
```

- dengan:
 - tipe: untuk menyatakan jenis tipe data dari elemen array (misalnya *char*, *int*, *unsigned*).
 - ukuran : untuk menyatakan jumlah maksimal elemen array.
 - nama_larik : untuk menyatakan nama array
- contoh pendeklarasian suatu array lima elemen dengan nama *nilai_tes* dan tipe datanya *float* adalah sebagai berikut:
float nilai_tes [10];

Deklarasi Array

```
float nilai_tes[10];
```

tipe data

nama array

ukuran atau jumlah element array

alamat	→	100	104	108	112	116	120	124	128	132	136
Data atau nilai elemen array	→	?	?	?	?	?	?	?	?	?	?
index	→	0	1	2	3	4	5	6	7	8	9

Mengakses Elemen Array

- Pada bahasa C, data array akan disimpan dalam memori pada lokasi yang berurutan.
- Elemen pertama mempunyai indeks bernilai 0.
- Jika pada contoh variabel `nilai_tes` mempunyai 10 elemen, maka elemen pertama mempunyai indeks sama dengan 0, elemen kedua mempunyai indeks 1, dan seterusnya.

Dua Operasi Pengaksesanr Array

1. Mengisi/memasukkan nilai ke dalam elemen array
2. Mengambil/menampilkan nilai yang ada dalam elemen array

Mengisi data pada Array

- Contoh untuk memasukkan data ke dalam elemen array *nilai_tes* ke-0 :
 - `nilai_tes[0] = 70;`
- merupakan contoh pemberian nilai 70 ke elemen array *nilai_tes* ke-0.

Mengambil data Array

- Bentuk umum pengaksesan suatu elemen variabel array adalah:

```
Nama_larik [indeks];
```

- Untuk variabel array nilai_tes,
 - Nilai_tes[1] → elemen ke-1 dari nilai_tes
 - Nilai_tes[3] → elemen ke-3 dari nilai_tes
- Dua cara yang ekuivalen untuk mengakses unsur ke-i dari suatu array. Misal untuk i=2;

**(A+2) atau A[2]*

cout<<A[2]) atau

cout<<(A+2);*

Contoh

nilai_tes[0]

nilai_tes[1]

nilai_tes[2]

nilai_tes[3]

nilai_tes[4]

70
90
80
90
100

tipe float

total 5
elemen

float nilai_tes[5]

- `//deklarasi array`
`int nilai_tes[5]`
- `// mengisi array`
`nilai_tes[3] = 90;`
- `// menampilkan isi array`
`cout<< nilai_tes[3];`

Deklarasi Array sekaligus Inisialisasi

- Data array juga dapat dideklarasikan dalam bentuk variabel yang bersifat statis, serta dapat dilakukan inisialisasi terhadap masing-masing elemen array.
- Contoh proses deklarasi dan inisialisasinya adalah:

```
int nilai_tes[5] = {70, 90, 80, 90, 100};
```

```
int nilai_tes[] = {70, 90, 80, 90, 100};
```

- Dengan deklarasi di atas, maka :
 - nilai_tes*[0] bernilai 70
 - nilai_tes*[1] bernilai 90
 - nilai_tes*[2] bernilai 80
 - nilai_tes*[3] bernilai 90
 - nilai_tes*[4] bernilai 100

Menampilkan data Array

- ```
cout << nilai_tes[0] << endl;
cout << nilai_tes [1] << endl;
cout << nilai_tes [2] << endl;
cout << nilai_tes [3] << endl;
cout << nilai_tes [4] << endl;
```



# Alamat elemen Array

```
#include <iostream>
using namespace std;
int main()
{
int nilai_tes[5] = {10, 20, 30, 40, 50};
int i;
cout<<"Elemen larik\t\t"<<"Nilai elemen\t\t"<<"Alamat memori\n";
for(i=0; i<5; i++)
{
cout<<"nilai_tes["<<i<<"] \t\t\t"<<nilai_tes[i]<<"\t\t\t\t\t"<<&nilai_tes[i]<<"\n";
}
return 0;
}
```

**0x adalah hexadesimal**

| Elemen larik | Nilai elemen | Alamat memori  |
|--------------|--------------|----------------|
| nilai_tes[0] | 10           | 0x7fff8fd69630 |
| nilai_tes[1] | 20           | 0x7fff8fd69634 |
| nilai_tes[2] | 30           | 0x7fff8fd69638 |
| nilai_tes[3] | 40           | 0x7fff8fd6963c |
| nilai_tes[4] | 50           | 0x7fff8fd69640 |

# Alamat elemen Array

```
#include <iostream>
using namespace std;
int main()
{
int nilai_tes[5] = {10, 20, 30, 40, 50};
int i;
cout<<"Elemen larik\t\t"<<"Nilai elemen\t\t"<<"Alamat memori\n";
for(i=0; i<5; i++)
{
cout<<"nilai_tes["<<i<<"] \t\t\t"<<*(nilai_tes+i)<<"\t\t\t\t"<<&nilai_tes[i]<<"\n";
}
return 0;
}
```

**0x adalah hexadesimal**

| Elemen larik | Nilai elemen | Alamat memori  |
|--------------|--------------|----------------|
| nilai_tes[0] | 10           | 0x7fff8fd69630 |
| nilai_tes[1] | 20           | 0x7fff8fd69634 |
| nilai_tes[2] | 30           | 0x7fff8fd69638 |
| nilai_tes[3] | 40           | 0x7fff8fd6963c |
| nilai_tes[4] | 50           | 0x7fff8fd69640 |



# Panjang array

```
#include<iostream>
#include<array>
using namespace std;
int main()
{
 int c;
 int arr[]={1,2,3,4,5,6,7,8,9,0};
 cout<<"The array is: ";
 for(auto i: arr)
 {
 cout<<i<<" ";
 c++;
 }
 cout<<"\nThe length of the given Array is: "<<c;
 return 0;
}
```

```
#include <iostream>
using namespace std;

int main() {
 int arr[] = {10,20,30,40,50,60};
 int arrSize = *(&arr + 1) - arr;
 cout << "The length of the array is: " << arrSize;
 return 0;
}
```



# Operasi pada array

- Copying
- Finding max or min element
- Shifting

# Copying Arrays

Can you copy array using a syntax like this?

```
targetArray = sourceArray;
```

This is not allowed in C++. You have to copy individual elements from one array to the other as follows:

```
for (int i = 0; i < sourceArrays.length; i++)
 targetArray[i] = sourceArray[i];
```

```

#include <iostream>
using namespace std;
int main()
{
int nilai_tes[5] = {10, 20, 30, 40, 50};
int i;
int nilai_uts[5];
cout<<"Elemen larik\t\t"<<"Nilai elemen\t\t"<<"Alamat memori\n";
for(i=0; i<5; i++)
{
nilai_uts[i]=nilai_tes[i];
cout<<"nilai_uts["<<i<<"] \t\t\t"<<nilai_uts[i]<<"\t\t\t\t\t"<<&nilai_uts[i]<<"\n";
}
return 0;
}

```

| Elemen larik | Nilai elemen | Alamat memori  |
|--------------|--------------|----------------|
| nilai_uts[0] | 10           | 0x7fffa6145e30 |
| nilai_uts[1] | 20           | 0x7fffa6145e34 |
| nilai_uts[2] | 30           | 0x7fffa6145e38 |
| nilai_uts[3] | 40           | 0x7fffa6145e3c |
| nilai_uts[4] | 50           | 0x7fffa6145e40 |



# Finding the Largest Element

Use a variable named **max** to store the largest element. Initially **max** is myList[0]. To find the largest element in the array myList, compare each element in myList with **max**, update **max** if the element is greater than max.

```
double max = myList[0];
for (int i = 1; i < ARRAY_SIZE; i++)
{
 if (myList[i] > max)
 max = myList[i];
}
```

```
#include <iostream>
using namespace std;
int main()
{
int nilai_tes[5] = {10, 20, 30, 40, 50};
int i;
int Max_nilai;
Max_nilai = nilai_tes[0];
for(i=1; i<5; i++)
{
if (nilai_tes[i]>Max_nilai)
 Max_nilai=nilai_tes[i];
}
cout<<"Nilai Terbesar adalah : "<<Max_nilai;
}
```

Output :  
Nilai Tes Terbesar 50



# Shifting Right

indeks

0 1 2 3 4 5 6 7

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|



|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|

indeks

0 1 2 3 4 5 6 7



```

#include <iostream>
#include <iostream>
using namespace std;
int main()
{
int nilai_tes[5] = {10, 20, 30, 40, 50};
int i,j;
cout<<"Array sebelum digeser ke kanan : ";
for(i=0; i<5; i++)
 cout<<nilai_tes[i]<<" ";
 cout<<"\n";
int sementara;
sementara = nilai_tes[4];
for(i=4; i>=0; i--)
{
 j=i-1;
 nilai_tes[i] = nilai_tes[j];
}
nilai_tes[0]= sementara;
cout<<"Array sesudah digeser ke kanan : ";
for(i=0; i<5; i++)
 cout<<nilai_tes[i]<<" ";
}

```

Output program :

Array sebelum digeser ke kiri : 10 20 30 40 50  
 Array sesudah digeser ke kiri : 50 10 20 30 40

```
#include<iostream>
using namespace std;
int main()
{
 int arr[5],i;
 cout<<"Enter elements of array: ";
 for(i=0;i<5;i++)
 cin>>arr[i];
 cout<<"\n";
 cout<<"Your array: ";
 for(i=0;i<5;i++)
 cout<<arr[i]<<" ";
 i=4;
 int j=i-1;
 int temp=arr[4];
 while(j>=0 && i>=0)
 {
 arr[i]=arr[j];
 j--;
 i--;
 }
 arr[i]=temp;
 cout<<"\nShifted array: ";
 for(i=0;i<5;i++)
 cout<<arr[i]<<" ";
 return 0;
}
```

Output :

Enter elements of array: 1 2 3 4 5

Your array: 1 2 3 4 5

Shifted array: 5 1 2 3 4



# Shifting Left

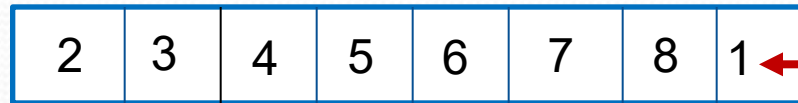
indeks

0 1 2 3 4 5 6 7



indeks

0 1 2 3 4 5 6 7





```
#include <iostream>
using namespace std;
int main()
{
 int nilai_tes[5] = {10, 20, 30, 40, 50};
 int i,j;
 cout<<"Array sebelum digeser ke kiri : ";
 for(i=0; i<5; i++)
 cout<<nilai_tes[i]<<" ";
 cout<<"\n";
 int sementara;
 sementara = nilai_tes[0];
 for(i=0; i<5; i++)
 {
 j=i+1;
 nilai_tes[i] = nilai_tes[j];
 }
 nilai_tes[4]= sementara;
 cout<<"Array sesudah digeser ke kiri : ";
 for(i=0; i<5; i++)
 cout<<nilai_tes[i]<<" ";
}
```

Output program :

Array sebelum digeser ke kiri : 10 20 30 40 50

Array sesudah digeser ke kiri : 20 30 40 50 10

# Deret Bilangan dengan Array satu dimensi

- Program mendapatkan deret 10 bilangan genap pertama menggunakan array satu dimensi.
- Bilangan genap pertama adalah : 0,2,4,6,8,10,12,14,16,18
- Menemukan persamaan deret bilangan genap, jika dasarnya adalah indeks pada array. Untuk mendapatkan deret 10 bilangan pertama bilangan genap maka indeks array yg digunakan adalah dari 0 sampai 9

Misal indeks disimpan dalam variable i, maka akan ada indeks

|        |   |   |   |   |   |   |   |   |   |   |
|--------|---|---|---|---|---|---|---|---|---|---|
| indeks | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|--------|---|---|---|---|---|---|---|---|---|---|

|          |   |   |   |   |   |    |    |    |    |    |
|----------|---|---|---|---|---|----|----|----|----|----|
| Bilangan | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 |
|----------|---|---|---|---|---|----|----|----|----|----|

Sehingga persamaannya adalah  $\text{deret}[i] = 2 * i$



```
#include <iostream>
using namespace std;
int main()
{
 int deret[10];
 int i,j;
 for(i=0; i<10; i++)
 {
 deret[i]=2*i;
 }
 cout<<"Deret bilangan genap : ";
 for(i=0; i<9; i++)
 cout<<deret[i]<<" ";
}
```

Output :

Deret bilangan genap : 0 2 4 6 8 10 12 14 16 18





# Array Dua Dimensi

# Array 2 Dimensi/ Matriks


- array berdimensi dua terdiri dari banyak baris dan banyak kolom yang bertipe sama
- Gambar array berdimensi 2  
(baris x kolom = 3 x 4)

matrix[0][0]  
Baris : 0  
Kolom: 0

|   | 0 | 1  | 2  | 3  |
|---|---|----|----|----|
| 0 | 5 | 20 | 1  | 11 |
| 1 | 4 | 7  | 67 | -9 |
| 2 | 9 | 0  | 45 | 3  |

matrix[3][0]Baris : 3  
Kolom : 0

matrix[0][3]  
Baris: 0  
Kolom: 3

- 
- Elemen-elemen matriks dapat diakses oleh program dengan menggunakan suatu indeks tertentu yang terdiri dari indeks baris dan kolom
  - Elemen-elemen matriks dapat diakses secara berurutan atau random berdasarkan indeks tertentu secara langsung.
  - Pengisian dan pengambilan nilai pada indeks tertentu dapat dilakukan dengan mengeset nilai atau menampilkan nilai pada indeks yang dimaksud.



# Lanjutan ...

- Tanda `[] []` disebut juga “elemen pada baris ke “ “ dan kolom ke “ “.
- Misalnya “`Matriks[1][1]`” berarti elemen matriks pada baris ke 1 dan kolom ke 1

# 2D Array Initialization

- Inisialisasi:

```
int matrix[4][3] = {1,2,3,4,5,6,7,8,9,10,11,12};
```

```
int matrix[4][3] = {{1, 2, 3},
 {4, 5, 6},
 {7, 8, 9},
 {10,11,12}};
```

# 2D Array Initialization

- Memanfaatkan statement nested loop :

```
int matrix[4][3], i, j;
 for (i=0; i<4; i++)
 for (j=0; j<3; j++)
 matrix[i][j] = i * j;
```

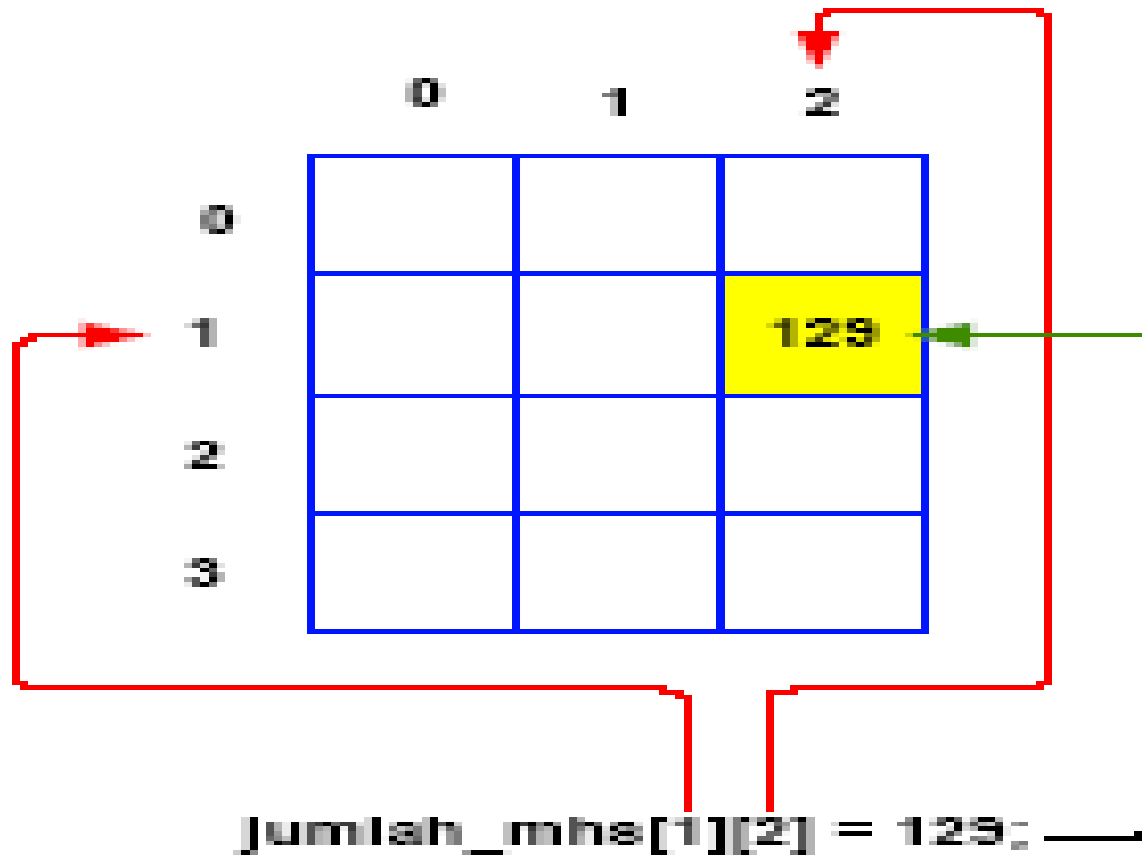
|       |       |       |
|-------|-------|-------|
| (0,0) | (0,1) | (0,2) |
| 0     | 0     | 0     |
| (1,0) | (1,1) | (1,2) |
| 0     | 1     | 2     |
| (2,0) | (2,1) | (2,2) |
| 0     | 2     | 4     |
| (3,0) | (3,1) | (3,2) |
| 0     | 3     | 6     |



# Mengakses Elemen Array 2D

- Untuk mengakses elemen array dua dimensi dapat dilakukan dengan statemen:
  - `nama_variabel[indeks pertama][indeks kedua];`
- Sebagai contoh:
  - `Jumlah_mhs[1][2] = 129;`
  - Merupakan instruksi untuk mengakses elemen array pada baris ke-1, kolom ke-2.

# Pengaksesan elemen array 2D



```
#include <iostream>
using namespace std;
int main () {
 int A[5][5], i, j, k,l; // deklarasi array A bertipe integer dan memiliki ukuran baris 5 dan kolom 5
 // inisialisasi indeks dari array dimulai dari indeks [0][0]
 for (int i = 0; i < 5; i++)
 {
 for (int j = 0; j < 5; j++)
 // set nilai elemen dimulai dari array A[0][0] , baris ke 0 akan diset ke nilai 100 selanjutnya kenaikan baris
 // akan bertambah 1
 A[i][j] = i+100;
 }
 cout << "Indeks\t\t" << "Nilai Elemen\t" << "Alamat" << endl;
 // menampilkan indeks dan nilai dari tiap elemen array A serta lokasi alamat memorinya
 for (int k = 0; k < 5; k++)
 {
 for (int l = 0; l < 5; l++)
 cout << "[" << k << "]" << "[" << l << "]" << "\t\t" << A[k][l] << "\t\t\t\t" << &A[k][l]
 << endl;
 }
}
```



| Indeks | Nilai Elemen | Alamat         |
|--------|--------------|----------------|
| [0][0] | 100          | 0x7ffea71755f0 |
| [0][1] | 100          | 0x7ffea71755f4 |
| [0][2] | 100          | 0x7ffea71755f8 |
| [0][3] | 100          | 0x7ffea71755fc |
| [0][4] | 100          | 0x7ffea7175600 |
| [1][0] | 101          | 0x7ffea7175604 |
| [1][1] | 101          | 0x7ffea7175608 |
| [1][2] | 101          | 0x7ffea717560c |
| [1][3] | 101          | 0x7ffea7175610 |
| [1][4] | 101          | 0x7ffea7175614 |
| [2][0] | 102          | 0x7ffea7175618 |
| [2][1] | 102          | 0x7ffea717561c |
| [2][2] | 102          | 0x7ffea7175620 |
| [2][3] | 102          | 0x7ffea7175624 |
| [2][4] | 102          | 0x7ffea7175628 |
| [3][0] | 103          | 0x7ffea717562c |
| [3][1] | 103          | 0x7ffea7175630 |
| [3][2] | 103          | 0x7ffea7175634 |
| [3][3] | 103          | 0x7ffea7175638 |
| [3][4] | 103          | 0x7ffea717563c |
| [4][0] | 104          | 0x7ffea7175640 |
| [4][1] | 104          | 0x7ffea7175644 |
| [4][2] | 104          | 0x7ffea7175648 |
| [4][3] | 104          | 0x7ffea717564c |
| [4][4] | 104          | 0x7ffea7175650 |

# Operasi pada Matriks

- Pengisian elemen matriks
- 2 cara pemrosesan elemen matriks, melalui baris dan kolomnya
- Pembacaan dan pencetakan elemen matriks
- Pemrosesan elemen matriks dengan operasi-operasi yang berlaku pada matriks
- Searching an matriks (linear or binary search)

## Pemrosesan Matrik Baris demi Baris

- Misalkan matriks  $A[M][N]$ , untuk memroses matriks  $A$  baris demi baris digunakan langkah algoritma sbb:

For Baris 0 to  $M-1$  do

For Kolom 0 to  $N-1$  do

***PROSES MATRIK***

Endfor

Endfor



## Pemrosesan Matrik Kolom demi Kolom

- Misalkan matriks  $A[M][N]$ , untuk memroses matriks  $A$  kolom demi kolom digunakan langkah algoritma sbb:

For Kolom 0 to  $N-1$  do

For Baris 0 to  $M-1$  do

***PROSES MATRIK***

Endfor

Endfor

## Pengisian/Penulisan Elemen Matriks

- Pengisian elemen matriks dengan suatu konstanta

$x[2][4] = \{\{8, 5, 9, 8\},$   
 $\{8, 2, 1, 0\}\};$

atau

$x[0][0]=8; x[0][1]=5; x[0][2]=9; x[0][3]=8;$   
 $x[1][0]=8; x[1][1]=2; x[1][2]=1; x[1][3]=0;$

Hasilnya menjadi sebuah matriks X

$$\mathbf{x} = \begin{pmatrix} 8 & 5 & 9 & 8 \\ 8 & 2 & 1 & 0 \end{pmatrix}$$

## Pengisian matriks dengan harga nol

For Baris = 0 to 1 do

For Kolom = 0 to 2 do

Matriks[Baris][Kolom] = 0

Endfor

Endfor



## Pengisian elemen matriks dengan data tertentu

```
For Baris = 0 to 1 do
```

```
 For Kolom = 0 to 2 do
```

```
 Input (data)
```

```
 Matriks[Baris][Kolom] = data
```

```
 Endfor
```

```
Endfor
```

## Mencetak elemen matriks

For Baris = 0 to 1 do

For Kolom = 0 to 2 do

***print Matriks[Baris][Kolom]***

Endfor

Endfor

```

#include <iostream>
using namespace std;
int main () {
 int A[5][5], i, j, k,l; // deklarasi array A bertipe integer dan memiliki ukuran baris 5 dan
 kolom 5
 // inisialisasi indeks dari array dimulai dari indeks [0][0]
 for (int i = 0; i < 5; i++)
 {
 for (int j = 0; j < 5; j++)
 // set nilai elemen dimulai dari array A[0][0] , baris ke 0 akan diset ke nilai 100 selanjutnya
 kenaikan baris akan bertambah 1
 A[i][j] = i+100;
 }
 cout << "Matriks"<<endl;
 // menampilkan matriks
 for (int k = 0; k < 5; k++)
 {
 for (int l = 0; l < 5; l++)
 cout <<A[k][l]<<" ";
 cout << "\n ";

 }
}

```

### Matriks

|     |     |     |     |     |
|-----|-----|-----|-----|-----|
| 100 | 100 | 100 | 100 | 100 |
| 101 | 101 | 101 | 101 | 101 |
| 102 | 102 | 102 | 102 | 102 |
| 103 | 103 | 103 | 103 | 103 |
| 104 | 104 | 104 | 104 | 104 |



# Pemrosesan elemen matriks dengan operasi-operasi yang berlaku pada matriks

- Penjumlahan Matriks
- Pengurangan Matriks
- Perkalian Matriks
- Diagonal Matriks
- Transpose Matriks

# Penjumlahan matriks

- Agar kedua matriks dapat dijumlahkan harus memiliki jumlah baris dan kolom yang sama.
- Inputkan matriks A dan matriks B
- Siapkan matriks C untuk menampung hasil
- Penjumlahan matriks A dan B sesuai dengan elemen-elemennya.

Elemen matriks A  $[o,o]$  dijumlahkan dengan elemen matriks B  $[o,o]$  juga dan disimpan di elemen matriks C  $[o,o]$  juga. Begitu seterusnya

**Array A**

|   |   |   |   |   |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 3 | 4 | 5 |
| 1 | 2 | 3 | 4 | 5 |

+


**Array B**

|   |   |   |   |   |
|---|---|---|---|---|
| 2 | 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | 3 | 3 |
| 4 | 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 5 | 5 |



- Hasilnya adalah matriks C

|   |   |   |   |    |
|---|---|---|---|----|
| 3 | 4 | 5 | 6 | 7  |
| 4 | 5 | 6 | 7 | 8  |
| 5 | 6 | 7 | 8 | 9  |
| 6 | 7 | 8 | 9 | 10 |



```
For Baris = 0 to 1 do
```


```
 For Kolom = 0 to 1 do
```

```
 C[Baris,Kolom] =
```

```
 A[Baris,Kolom] + B[Baris,Kolom]
```

```
 Endfor
```

```
Endfor
```



```
int i, j;
```

```
int c[i][j], a[i][j], b[i][j];
```

```
for (i = 0; i < rows; i++)
```

```
 for (j = 0; j < cols; j++)
```


```
 c[i][j] = a[i][j] + b[i][j];
```



# Pengurangan 2 buah Matriks

- Agar kedua matriks dapat dikurangkan harus memiliki jumlah baris dan kolom yang sama.
- Inputkan matriks A dan matriks B
- Siapkan matriks C untuk menampung hasil
- pengurangan matriks A dan B sesuai dengan elemen-elemennya.

elemen matriks A  $[o,o]$  dikurangkan dengan elemen matriks B  $[o,o]$  juga dan disimpan di elemen matriks C  $[o,o]$  juga. Begitu seterusnya



For Baris = 0 to 1 do

For Kolom = 0 to 1 do


**C[Baris,Kolom] =**

**A[Baris,Kolom] - B[Baris,Kolom]**

Endfor

Endfor





```
int i, j;
```

```
int c[i][j], a[i][j], b[i][j];
```

```
for (i = 0; i < rows; i++)
```

```
 for (j = 0; j < cols; j++)
```

```
 c[i][j] = a[i][j] - b[i][j];
```




# Perkalian Matriks dengan Konstanta

3 x

|   |   |   |   |    |
|---|---|---|---|----|
| 3 | 4 | 5 | 6 | 7  |
| 4 | 5 | 6 | 7 | 8  |
| 5 | 6 | 7 | 8 | 9  |
| 6 | 7 | 8 | 9 | 10 |

Hasilnya adalah =

|    |    |    |    |    |
|----|----|----|----|----|
| 9  | 12 | 15 | 18 | 21 |
| 12 | 15 | 18 | 21 | 24 |
| 15 | 18 | 21 | 24 | 27 |
| 18 | 21 | 24 | 27 | 30 |



```
For Baris = 0 to 1 do
```


```
 For Kolom = 0 to 2 do
```

```
 C[Baris,Kolom] = 3 * A[Baris,Kolom]
```

```
 Endfor
```

```
Endfor
```





```
int i, j;
```

```
int a[i][j], b[i][j];
```

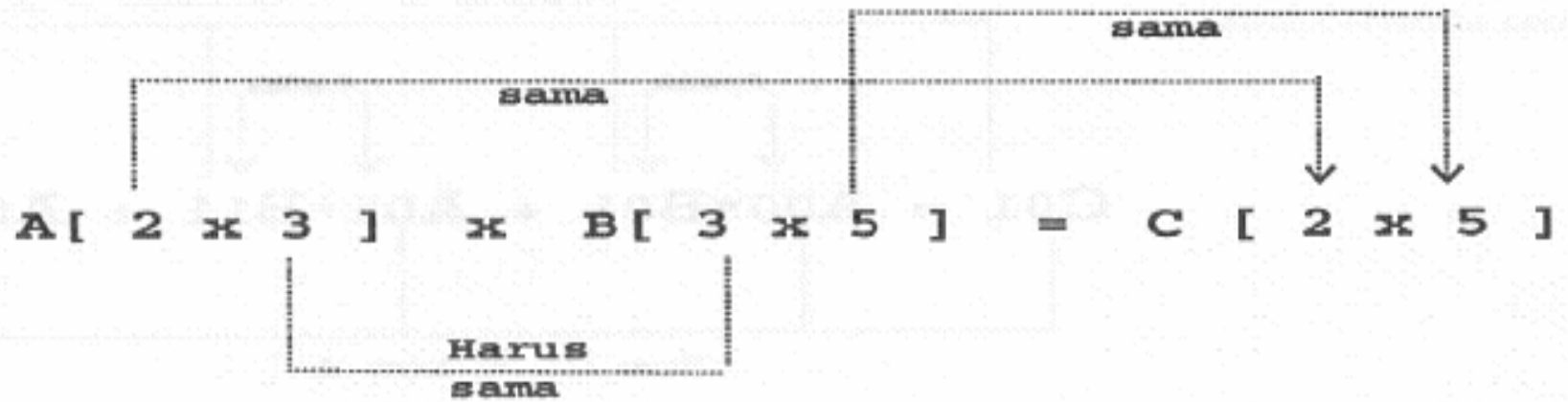
```
for (i = 0; i < rows; i++)
```

```
 for (j = 0; j < cols; j++)
```

```
 b[i][j] = 3 * a[i][j];
```

# Perkalian 2 buah Matriks

- Kedua matriks harus memiliki bentuk  $m \times n$  untuk matriks A dan  $p \times q$  untuk matriks B
- Sehingga matriks hasil akan memiliki bentuk  $m \times q$
- Sehingga :  $(m \times q) = (m \times n) * (p \times q)$
- Inputkan matriks A
- Inputkan matriks B
- Tampung hasil perkalian matriks pada matriks C





|      |   | 0 | 1 | 2 |
|------|---|---|---|---|
| A[ ] | 0 | 2 | 4 | 3 |
|      | 1 | 3 | 2 | 5 |

**X**

|      |   | 0 | 1 | 2 | 3 | 4 |
|------|---|---|---|---|---|---|
| B[ ] | 0 | 3 | 2 | 5 | 7 | 4 |
|      | 1 | 2 | 4 | 6 | 3 | 2 |
|      | 2 | 3 | 3 | 2 | 5 | 4 |

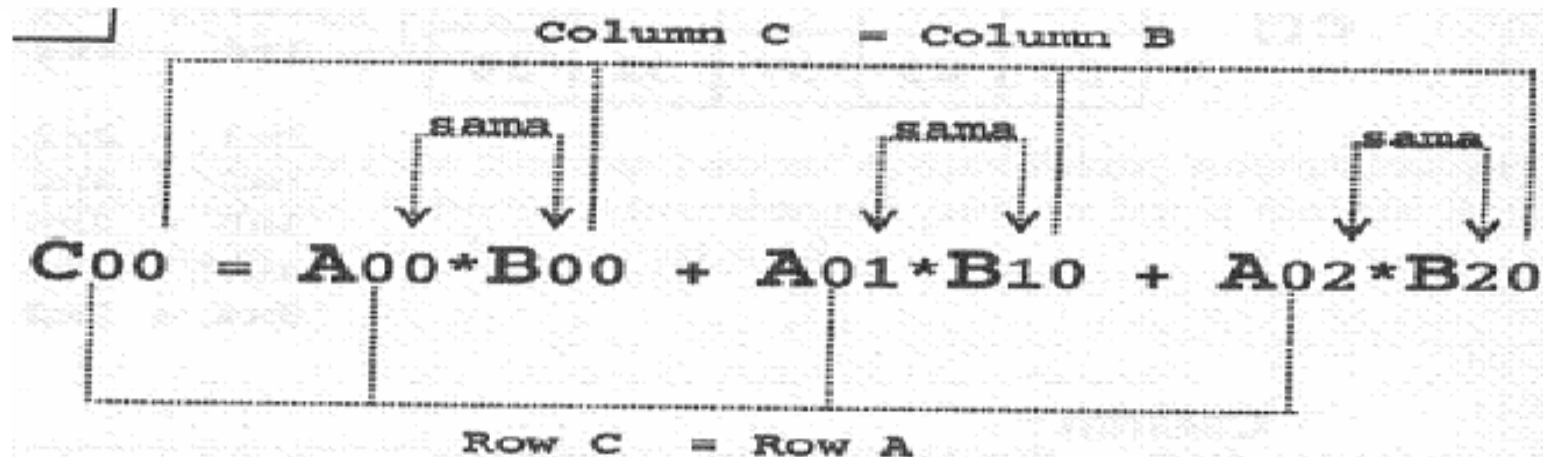
| A[ ] |   |   |
|------|---|---|
| 2    | 4 | 3 |
|      |   |   |

| B[ ] |  |  |  |  |
|------|--|--|--|--|
| 3    |  |  |  |  |
| 2    |  |  |  |  |
| 3    |  |  |  |  |

| C[ ] |  |  |  |  |
|------|--|--|--|--|
| 23   |  |  |  |  |
|      |  |  |  |  |

$$C_{00} = A_{00} * B_{00} + A_{01} * B_{10} + A_{02} * B_{20}$$

$$23 = 2 * 3 + 4 * 2 + 3 * 3$$



| A[ ] |   |   |
|------|---|---|
| 2    | 4 | 3 |
|      |   |   |

| B[ ] |   |  |  |  |
|------|---|--|--|--|
|      | 2 |  |  |  |
|      | 4 |  |  |  |
|      | 3 |  |  |  |

| C[ ] |    |  |  |  |
|------|----|--|--|--|
|      | 29 |  |  |  |
|      |    |  |  |  |

$$C_{01} = A_{00} * B_{01} + A_{01} * B_{11} + A_{02} * B_{21}$$

Column C = Column B

$$C_{01} = A_{00} * B_{01} + A_{01} * B_{11} + A_{02} * B_{21}$$

Row C = Row A





```
For i = 0 to 2 do
```

```
 For j = 0 to 5 do
```

```
 C[i,j] = 0
```

```
 For k = 0 to 5 do
```

```
 C[i,j] = C[i,j] + A[i,k] * B[k,j]
```

```
 Endfor
```

```
 Endfor
```

```
 Endfor
```

```
int i, j, k
int rowsa = 2;
int colsb = 5 ;
Int colsa = 5;
 for (i = 0; i < rowsa; i++)
 for (j = 0; j < colsb; j++) {
 c[i][j] = 0;
 for (k = 0; k < colsa; k++)
 c[i][j] += a[i][k] * b[k][j];
 }
```

# Diagonal Matriks

- Matriks harus bujur sangkar
- Diagonal adalah elemen matriks yang baris dan kolomnya sama

For Baris = 0 to 1 do

For Kolom = 0 to 1 do

**if Baris= Kolom**

**print ( A[Baris,Kolom])**

Endfor

Endfor



# Transpose

- Transpose adalah elemen baris matriks akan menjadi kolom matriks dan sebaliknya kolom matriks akan menjadi baris matriks.

For Baris = 0 to 1 do

For Kolom = 0 to 2 do

**MatriksTranspose[Baris,Kolom]=  
A[Kolom,Baris]**

Endfor

Endfor

```
#include <iostream>
using namespace std;
```

```
int main() {
```

```
 int i, j, rows;
 cout<<"Enter number of rows: "<<"\n";
 cin>>rows;
 for(i=1; i<=rows; ++i)
 {
 for(j=1; j<=i; ++j)
 {
 cout<<j;
 }
 cout<<"\n";
 }
 return 0;
}
```

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```



```
#include <iostream>
using namespace std;
```

```
int main() {
```

```
 int i, j;
```

```
 char inputan, alphabet = 'A';
```

```
 cout<<"Enter the uppercase character
you want to print in last row: "<<"\n";
```

```
 cin>>inputan;
```

```
 for(i=1; i <= (inputan-'A'+1); ++i)
```

```
 {
```

```
 for(j=1;j<=i;++j)
```

```
 {
```

```
 cout<<alphabet;
```

```
 }
```

```
 ++alphabet;
```

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

```
 }
```

```
 return 0;
```

```
}
```

A

B B

C C C

D D D D

E E E E E





1

2 3

4 5 6

7 8 9 10

# Questions?

