



Institut Teknologi
Telkom
Purwokerto

Teknik Informatika - Fakultas Informatika

Pertemuan 4 – Struct dan Pointer

Author: **Wahyu Andi Saputra [WAA]**

Co-Author: **Condro Kartiko [CKO]**



STRUCT

Tipe Data Bentuk User

- Bahasa pemrograman bisa memiliki tipe data:
 - **Built-in** : sudah tersedia oleh bahasa pemrograman tersebut
 - Tidak berorientasi pada persoalan yang dihadapi.
 - **UDT : User Defined Type**, dibuat oleh pemrogram.
 - Mendekati penyelesaian persoalan yang dihadapi
 - Contoh: record pada Pascal, **struct pada C/C++**, class pada Java
 - **ADT : Abstract Data Type**
 - memperluas konsep UDT dengan menambahkan pengkapsulan atau enkapsulasi, berisi sifat-sifat dan operasi-operasi yang bisa dilakukan terhadap kelas tersebut.
 - Contoh: class pada Java

Tipe Data Bentukan User

•STRUCT

- adalah kumpulan data yang saling berhubungan, yang disimpan dalam satu unit penyimpanan.

Struct

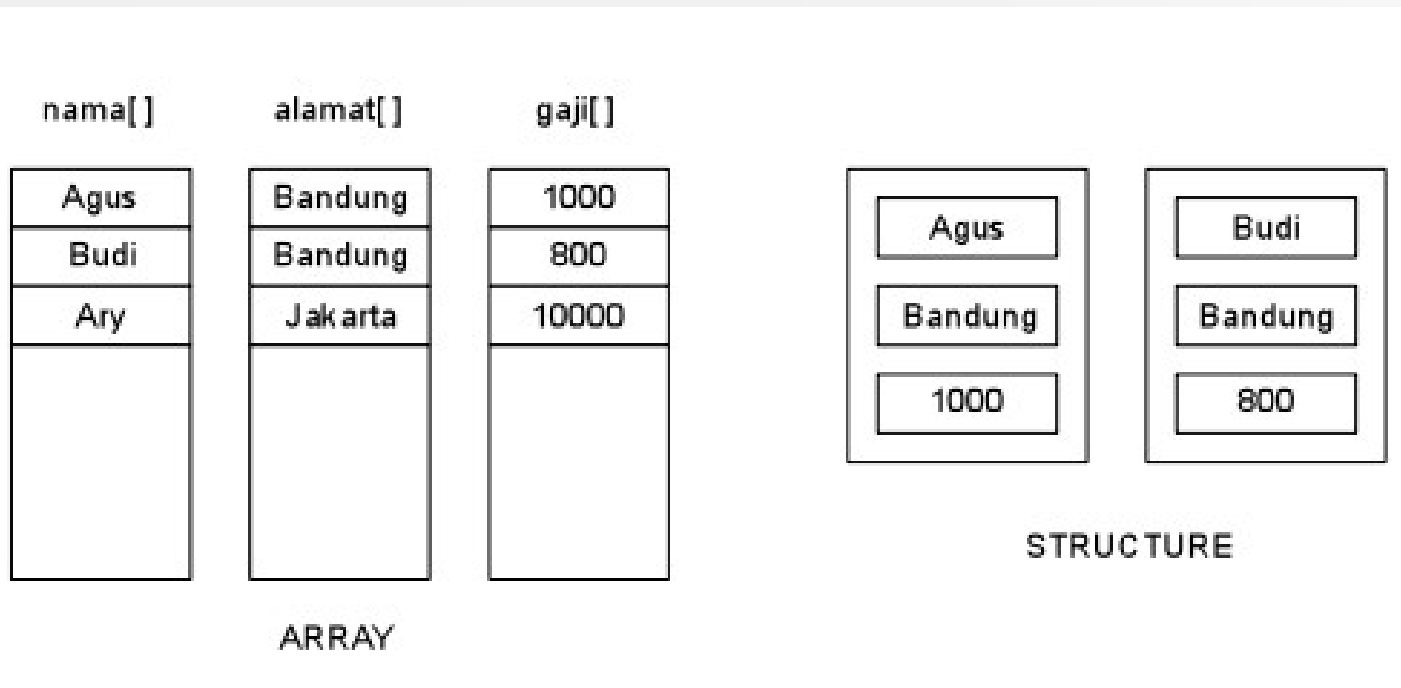
- nama,
- alamat ,
- Gaji,
- Jabatan,
- Umur,
- Pendidikan terakhir



Struct

- nama,
 - alamat ,
 - Gaji,
 - Jabatan,
 - Umur,
 - Pendidikan terakhir
- Bila menggunakan **array biasa**, maka diperlukan 6 variable yang bebas satu dengan yang lain, yaitu variabel **nama, alamat dan gaji, jabatan, umur, Pendidikan terakhir**
 - Dengan menggunakan **structure**, data tersebut diorganisasikan dalam satu kesatuan

Struct vs Array



Deklarasi structure

```
//Deklarasi struct  
struct data_mahasiswa  
{  
    long int nim;  
    char nama[100];  
    char fakultas[100];  
};
```

```
//Deklarasi variabel struct  
data_mahasiswa mahasiswa1, mahasiswa2;
```


Deklarasi structure

```
int main()
{
    //Input struct data mahasiswa
    cout<<" Data Mahasiswa Pertama\n";
    cout<<"-----\n";
    cout<<" NIM      : "; cin>>mahasiswa1.nim;
    cout<<" Nama      : "; fflush(stdin); gets(mahasiswa1.nama);
    cout<<" Fakultas : "; fflush(stdin); gets(mahasiswa1.fakultas);
    cout<<"\n\n";
    cout<<" Data Mahasiswa Kedua\n";
    cout<<"-----\n";
    cout<<" NIM      : "; cin>>mahasiswa2.nim;
    cout<<" Nama      : "; fflush(stdin); gets(mahasiswa2.nama);
    cout<<" Fakultas : "; fflush(stdin); gets(mahasiswa2.fakultas);
    cout<<"\n\n";
}
```

Deklarasi structure

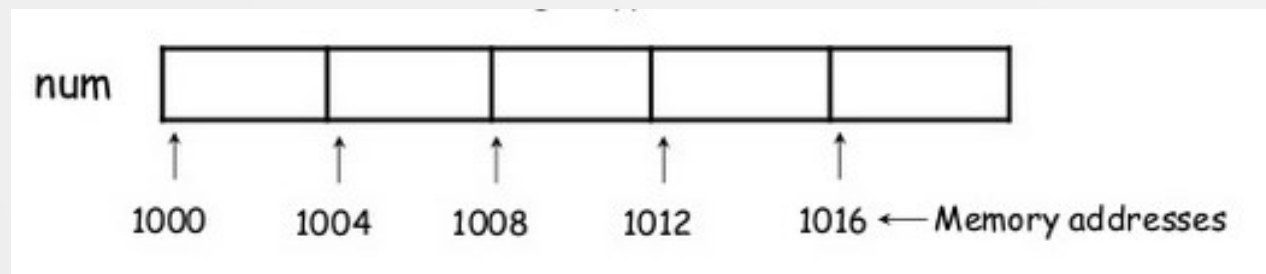
```
//Output struct data mahasiswa
cout<<" Data Mahasiswa Pertama\n";
cout<<"-----\n";
cout<<" NIM      : "<<mahasiswa1.nim<<endl;
cout<<" Nama      : "<<mahasiswa1.nama<<endl;
cout<<" Fakultas : "<<mahasiswa1.fakultas<<endl;
cout<<"\n\n";
cout<<" Data Mahasiswa Kedua\n";
cout<<"-----\n";
cout<<" NIM      : "<<mahasiswa2.nim<<endl;
cout<<" Nama      : "<<mahasiswa2.nama<<endl;
cout<<" Fakultas : "<<mahasiswa2.fakultas<<endl;
cout<<"\n\n";
getch();
}
```



POINTER

Mengapa pointer disebut dinamis?

- Sebuah urutan variabel dengan nama dan tipe data yang sama
int num [5]
- 5 buah urutan variabel dengan tipe integer dengan nama variabel num



Mengapa pointer disebut dinamis?

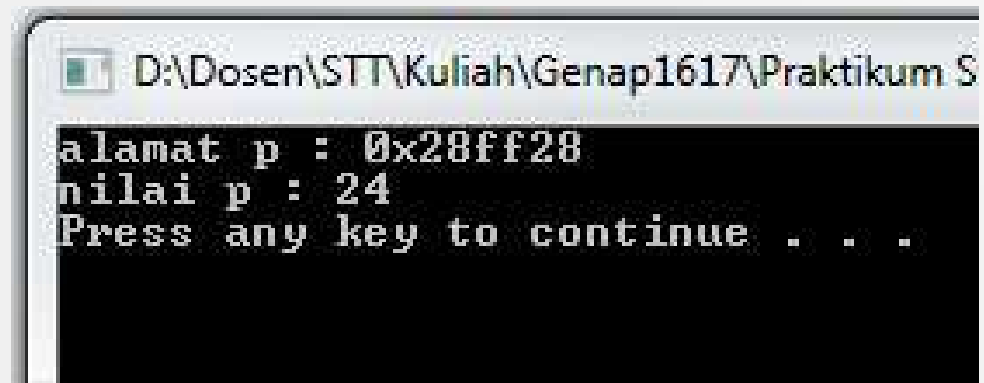
- bersifat statis (ukuran dan urutannya sudah pasti).
- ruang memori yang dipakai olehnya tidak dapat dihapus bila variabel bertipe array tersebut sudah tidak digunakan lagi pada saat program dijalankan.
- **Sedangkan**, pointer bersifat dinamis, **variabel akan dialokasikan hanya pada saat dibutuhkan** dan sesudah tidak dibutuhkan dapat dihapus kembali.

Operator pointer

- & → menghasilkan **alamat**
- * → menghasilkan reference dari sebuah alamat (**nilai/value**)

Try it!

```
int a[5];  
  
    int *p;  
  
    a[0]=24;  
  
    a[1]=32;  
  
    a[2]=81;  
  
    a[3]=44;  
  
    a[4]=23;  
  
    p=&a[0];  
    cout<<"alamat p : "<<p<<endl;  
    cout<<"nilai p : "<<*p<<endl;
```



```
D:\Dosen\STT\Kuliah\Genap1617\Praktikum S  
alamat p : 0x28ff28  
nilai p : 24  
Press any key to continue . . .
```

Try it!

```
Int a = 5 ;  
Int b = 10 ;  
Int c = a + b ;  
  
Int *d = &c ;
```

```
Cout << d ; -> 0x03  
Cout << *d ; -> 15
```

```
Int *e = d ;
```

Variabel	Nilai	Alamat
a	5	0x01
b	10	0x02
c	a+b=15	0x03
d	0x03	0x04
e	0x03	0x05



Challenge!

```
E:\Download\Documents\aa\DEVCC\strukdat\yan.exe
nilai index ke-0 dari a[0] adalah 24
nilai index ke-1 dari a[1] adalah 32
nilai index ke-2 dari a[2] adalah 81
nilai index ke-3 dari a[3] adalah 44
nilai index ke-4 dari a[4] adalah 23

===>> untuk p=&a[3]
alamat p : 0x6ffe2c
nilai p : 44
ampersand p : 0x6ffe10

===>> untuk p=&b
alamat p : 0x6ffe1c
nilai p : 56
ampersand p : 0x6ffe10
```



Pointer Bertipe Void

- Pada C++ terdapat pointer yang dapat menunjuk ke tipe data apapun, pointer semacam ini dideklarasikan dengan

tipe **void** sehingga sering dikenal dengan istilah **Void**

Pointer.

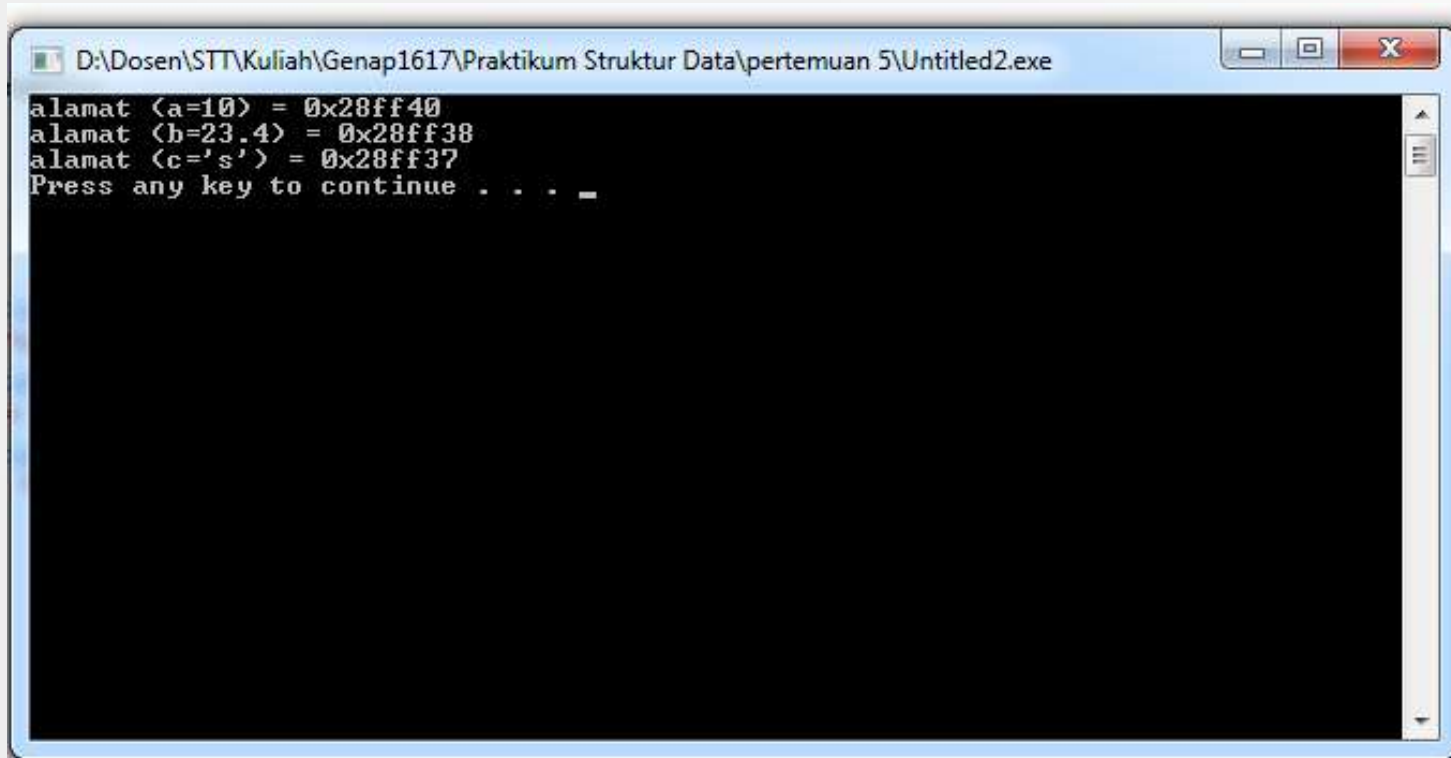
Pointer Bertipe Void

```
#include <iostream>
#include <stdlib.h>
#include <string>
#include <conio.h>
#include <stdio.h>

int main(int argc, char *argv[])
{
    void *p;
    int a=10;
    double b=23.4;
    char c='s';
    p=&a; //p menunjuk ke tipe data int
    cout<<"alamat (a=10) = "<<p<<endl;
    p=&b; //p menunjuk ke tipe data double
    cout<<"alamat (b=23.4) = "<<p<<endl;
    p=&c; //p menunjuk ke tipe data char
    cout<<"alamat (c='s') = "<<p<<endl;

    system("PAUSE");
    return 0;
}
```

Pointer Bertipe Void



```
D:\Dosen\STT\Kuliah\Genap1617\Praktikum Struktur Data\pertemuan 5\Untitled2.exe  
alamat (a=10) = 0x28ff40  
alamat (h=23.4) = 0x28ff38  
alamat (c='s') = 0x28ff37  
Press any key to continue . . . _
```

Challenge!

```
nilai index ke-0 dari a[0] adalah 24  
nilai index ke-1 dari a[1] adalah 32  
nilai index ke-2 dari a[2] adalah 81  
nilai index ke-3 dari a[3] adalah 44  
nilai index ke-4 dari a[4] adalah 23
```

```
alamat p : 0x6ffe2c  
nilai p : 44  
ampersand p : 0x6ffe10
```

```
alamat p : 0x6ffe1c  
nilai p : 56  
ampersand p : 0x6ffe10
```

Hubungan array dan pointer

Manakah alamat masing-masing elemen?

```
int num[5] ;
```

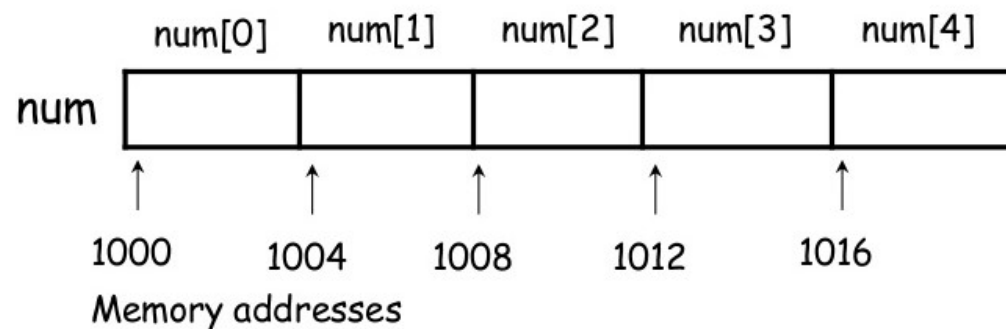
```
&num[0] == 1000
```

```
&num[1] == 1004
```

```
&num[2] == 1008
```

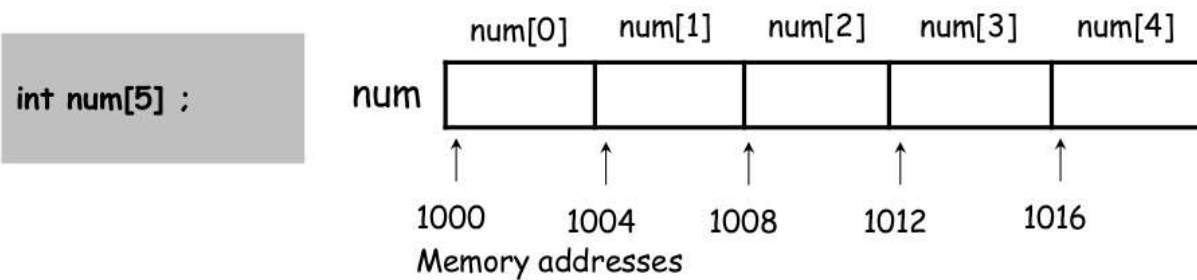
```
&num[3] == 1012
```

```
&num[4] == 1016
```



Hubungan array dan pointer

Apa itu num?



- *num* is the constant pointer of which value is the start address of the array.

```
num == &num[0] == 1000
```

Pointer Aritmatika

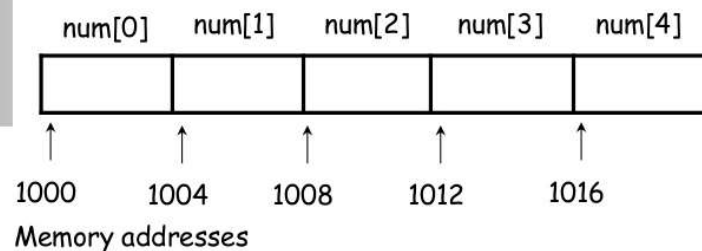
■ Example : Arithmetic of pointers

- "*pointer* + 1" does not mean increasing *pointer* by 1.
- "*pointer* + 1" is "the address of the next element".
- "*pointer* - 1" is "the address of the prior element".

```
num == &num[0] == 1000
```

```
(num+0) == &num[0]  
(num+1) == &num[1]  
(num+2) == &num[2]  
(num+3) == &num[3]  
(num+4) == &num[4]
```

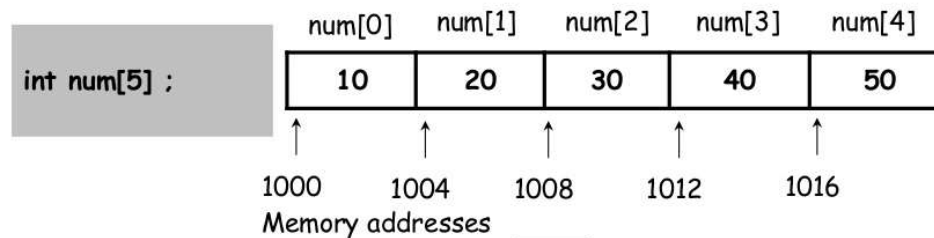
```
int num[5];
```



Pointer Aritmatika

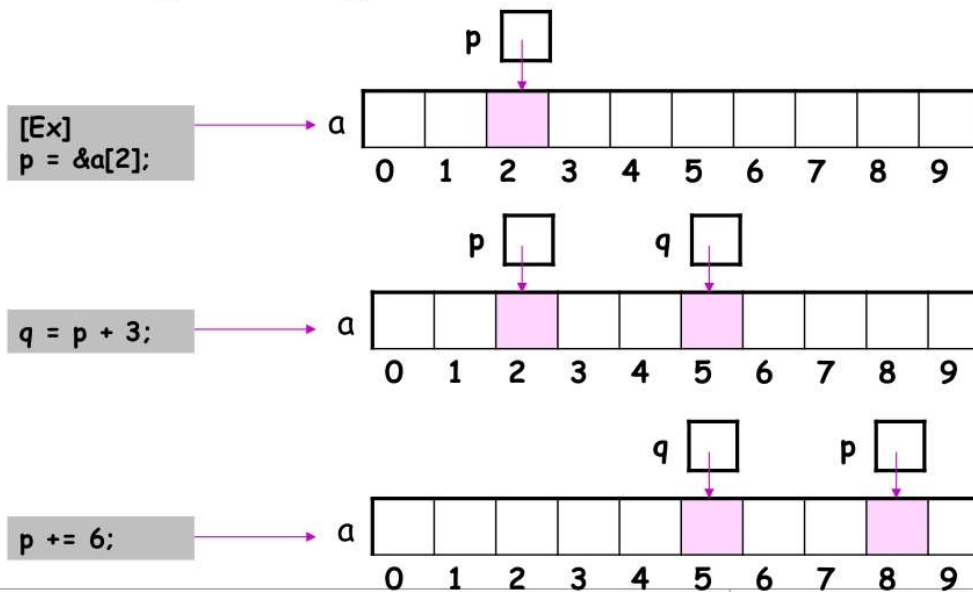
Example : Arithmetic of pointers

<code>int num[5] ;</code>	<code>int num[5] ;</code>	<code>int num[5], *p = num ;</code>	<code>int num[5], *p = num ;</code>
<code>num[0] = 10 ;</code>	<code>*num = 10 ;</code>	<code>*p = 10 ;</code>	<code>p[0] = 10 ;</code>
<code>num[1] = 20 ;</code>	<code>*(num+1) = 20 ;</code>	<code>*(p+1) = 20 ;</code>	<code>p[1] = 20 ;</code>
<code>num[2] = 30 ;</code>	<code>*(num+2) = 30 ;</code>	<code>*(p+2) = 30 ;</code>	<code>p[2] = 30 ;</code>
<code>num[3] = 40 ;</code>	<code>*(num+3) = 40 ;</code>	<code>*(p+3) = 40 ;</code>	<code>p[3] = 40 ;</code>
<code>num[4] = 50 ;</code>	<code>*(num+4) = 50 ;</code>	<code>*(p+4) = 50 ;</code>	<code>p[4] = 50 ;</code>



Pointer Aritmatika

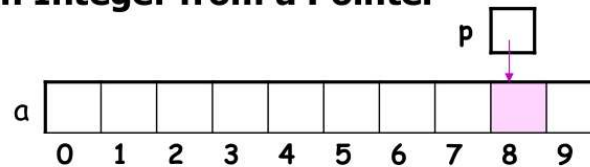
■ Adding an Integer to a Pointer



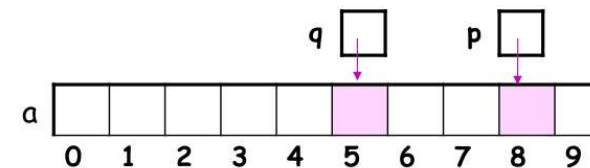
Pointer Aritmatika

▪ Subtracting an Integer from a Pointer

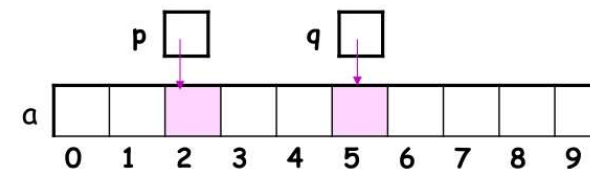
[Ex]
`p = &a[8];`



`q = p - 3;`



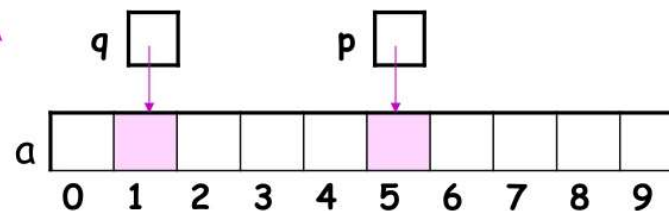
`p -= 6;`



Pointer Aritmatika

■ Subtracting Pointers

```
[Ex]  
p = &a[5];  
q = &a[1];  
  
i = p - q;    /* i == 4 */  
i = q - p;    /* i == -4 */
```



Pointer Aritmatika

■ Comparing Pointers

- Relational operators (<, <=, >, >=) can be applied
- Equality operators (==, !=) can be applied

```
[Ex]
p = &a[5];
q = &a[1];

p <= q;  /* result is 0 */
p >= q;  /* result is 1 */
```

Pointer Aritmatika

- **Example: Pointer Operation**

```
int a[ ] = { 5,15,25,43,12,1,7,89,32,11}  
int *p = &a[1], *q = &a[5] ;
```

1. $*(p + 3)$?

2. $*(q - 2)$?

3. $q - p$?

4. $\text{if } (p > q)$?

5. $\text{if } (*p > *q)$?

Pointer Aritmatika

- **Example: Pointer Operation**

```
int a[ ] = { 5,15,25,43,12,1,7,89,32,11}  
int *p = &a[1], *q = &a[5] ;
```

1. $*(p + 3)$?

2. $*(q - 2)$?

3. $q - p$?

4. $\text{if } (p > q)$?

5. $\text{if } (*p > *q)$?



Institut Teknologi
Telkom
Purwokerto

TERIMA KASIH