

# STRUKTUR DATA DAN ALGORITMA

## Overview : Array, Pointer, Struct



Oleh:

Indra Hermawan

PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TINGGI TEKNOLOGI TERPADU NURUL FIKRI

# Outline

---

- Array
- Struktur
- Pointer

## Array

- Array adalah sekelompok data yang bertipe sama yang menggunakan sebuah nama yang sama dan disimpan dalam urutan tertentu.
- Data di suatu array disebut dengan elemen array.
- Letak urutan dari elemen-elemen array ditunjukkan oleh suatu indeks.

## Array dimensi 1

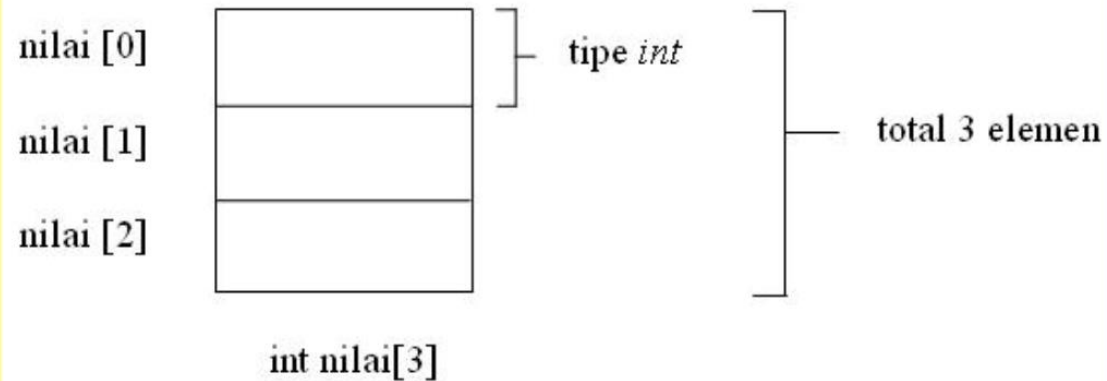
- Deklarasi

```
tipe_data nama_var[panjang_element];
```

- Contoh:

```
int nilai[3];
```

## Mengakses elemen array



- Bentuk umum pengaksesan array adalah:

```
nama_var[indeks]
```

nilai[0] → elemen 1, indek ke-0 dari nilai

nilai[2] → elemen 3, indek ke-2 dari nilai

- Contoh:

```
nilai[0] = 70;
```

```
scanf("%d", &nilai[2]);
```

## Inisialisasi array

- Inisialisasi array dapat dilakukan dengan cara memasukkan nilai-nilai dari setiap elemen array setelah deklarasi array dan harus diapit dengan tanda { dan }

- Contoh:

```
int jum_hari[12] = { 31, 28, 31, 30, 31, 30,  
                    31, 31, 30, 31, 30, 31};
```

## Beberapa variasi dalam deklarasi dan inisialisasi array

- `int numbers[10];`
- `int numbers[10] = { 34, 27, 16 };`
- `int numbers[] = { 2, -3, 45, 79, -14, 5, 9, 28, -1, 0 };`
- `char text[] = "Welcome to New Zealand.";`
- `float radix[12] = { 134.362, 1913.248 };`
- `double radians[1000];`



## Array berdimensi 2

- Bentuk deklarasi:

```
tipe_data nama_var[pjg_elm1][pjg_elm2];
```

- Contoh:

```
int nilai[3][5];
```

## Mengakses array dimensi 2

- Bentuk umum untuk mengakses elemen array berdimensi 2 adalah:

```
nama_var[indeks_dim1, indeks_dim2]
```

- Contoh:

```
nilai[0][1] = 540;  
printf("%d", nilai[1][2]);
```

## Inisialisasi array dimensi 2


- Hampir mirip dengan menginisialisasi array berdimensi 1, hanya saja dituliskan pengelompokan nilai dalam tiap dimensi dengan tanda { }.

- Contoh:

```
int nilai[2][3]={ {7, 6, 8},  
                 {5, 7, 6} };
```

## Array sebagai parameter

```
int nilai[3];  
...  
proses_data(nilai);
```



```
void proses_data(int x[]) {  
    ...  
}
```

## Struct

- Cara pengelompokan data dengan nama yang sama, akan tetapi dapat mempunyai tipe-tipe yang berbeda
- Struktur biasa dipakai untuk mengelompokkan beberapa informasi yang berkaitan menjadi sebuah satu kesatuan.

## Deklarasi struct

```
struct nama_struct {  
    tipe data_1;  
    ...  
    tipe data_n;  
} var_1, ..., var_n;
```

## Contoh deklarasi struct

```
struct datasiswa {  
    int nrp;  
    char nama[20];  
};
```

```
Struct datasiswa siswa;
```

```
struct datatanggal {  
    int tanggal;  
    int bulan;  
    int tahun;  
} tanggal_lahir;
```

```
Struct datatanggal tanggal_lahir;
```

## Contoh deklarasi struct dan deklarasi variabel struct

```
struct datasiswa {  
    int nrp;  
    char nama[20];  
} siswa;
```

```
struct datatanggal {  
    int tanggal;  
    int bulan;  
    int tahun;  
} tanggal_lahir;
```



## Contoh deklarasi struct, deklarasi variabel struct, dan inisialisasi

```
struct datasiswa {  
    int nrp;  
    char nama[20];  
} siswa = {7, "Agus"};
```

```
struct datatanggal {  
    int tanggal;  
    int bulan;  
    int tahun;  
} tanggal_lahir =  
    {17, 8, 1945};
```

## Mengakses struct

- Mengakses struct dapat dilakukan dengan menyebutkan nama variabel struct-nya dan diiringi dengan data pada struct yang ingin diakses (dipisahkan dengan tanda . )
- Bentuk umum:  
`var_struct.data_i`

## Contoh

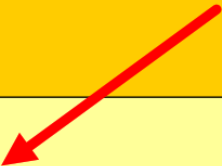
```
struct datatanggal {  
    int tanggal;  
    int bulan;  
    int tahun;  
} tanggal_lahir;
```

```
tanggal_lahir.tanggal=17;  
tanggal_lahir.bulan=8;  
tanggal_lahir.tahun=1945;
```

## Struct dalam struct

```
struct datatanggal {  
    int tanggal;  
    int bulan;  
    int tahun;  
};
```

```
struct datasiswa {  
    int nrp;  
    char nama[20];  
    struct datatanggal tgllahir;  
} siswa;
```



siswa.tgllahir.tanggal = 17

## Pointer

- Suatu variabel yang dipakai untuk menyimpan alamat dari suatu data
- Pointer tidak berisi nilai dari data, akan tetapi berisi alamat dari data.
- Berguna untuk pengalokasian memori secara dinamis

## Deklarasi pointer

- Bentuk umum:

```
tipe *var_pointer;
```

```
int *alamatdata;
```


- Untuk mengetahui alamat dari data, dapat dipakai tanda &.

```
alamatdata=&data;
```

## Contoh

```
int x;  
int *y;  
...  
x=2;  
y=&x;
```

	alamat	data
x		
	1000	2
y		
		1000



## Type pointer

- Untuk membuat suatu tipe berbentuk pointer
- Menambahkan kata kunci Typedef
- Contoh:

```
Typedef int *IntegerPointer;
```

```
IntegerPointer a;
```

← Variabel pointer  
bertipe int



- Pendahuluan
- Array Berdimensi Satu
  - Mendeklarasikan Array
  - Mengakses Elemen Array
  - Menginisialisasi Array
  - Variasi dalam Mendeklarasikan Array
- Array Berdimensi Dua
  - Mendeklarasikan Array
  - Mengakses Elemen Array
- Array Berdimensi Banyak
- Inisialisasi Array Tak Berukuran
- Array Sebagai Parameter Fungsi

- Pendahuluan
- Mendefinisikan Struktur
- Mendeklarasikan Struktur
- Mengakses Elemen Struktur
- Menginisialisasi Struktur
- Array dan Struktur (*array of struct*)

