

# SRUKTUR DATA

## Struct

## Definition — *Structure*

- Beberapa variabel (dapat berbeda tipe) yang dikelompokkan menjadi satu dengan sebuah nama baru

# struct

- Penting untuk implementasi ADT / membuat tipe data baru
- E.g.,

```
struct motor {  
    float volts;           //voltage of the motor  
    float amps;           //amperage of the motor  
    int phases;           //# of phases of the motor  
    float rpm;            //rotational speed of motor  
};  
  
typedef struct motor motor;
```

# struct

```
struct motor {  
    float volts;  
    float amps;  
    int phases;  
    float rpm;  
};  
typedef struct motor motor;
```

# struct

- E.g.,

```
struct motor {  
    float volts;  
    float amps;  
    int phases;  
    float rpm;  
};  
typedef struct motor motor;
```



Members of the struct

# struct

- Mendefinisikan tipe data baru
- E.g.,

```
struct motor {  
    float volts;  
    float amps;  
    int phases;  
    float rpm;  
};
```

Agar pada saat deklarasi variabel cukup dengan memanggil motor saja



```
typedef struct motor motor;
```

# Menggunakan tipe data baru

```
motor p, q, r;
```

- Mendefinisikan tiga variable – `p`, `q`, dan `r` – masing masing bertipe data `motor`

```
motor M[25];
```

- Mendeklarasikan array `M` berisi 25 data bertipe `motor`

```
motor *m;
```

- Mendeklarasikan variabel pointer yang menyimpan alamat slot memori yang berisi data bertipe `motor`

# Mengakses anggota struct

- Deklarasi

```
motor p;  
motor q[10];
```

- Maka

`p.volts`

— is the voltage

`p.amps`

— is the amperage

`p.phases`

— is the number of phases

`p.rpm`

— is the rotational speed

`q[i].volts`

— is the voltage of the *i*th motor

`q[i].rpm`

— is the speed of the *i*th motor



# Mengakses elemen struct menggunakan pointer

- Deklarasi

`motor *p;`

- Maka

`(*p).volts` — is the voltage of the motor pointed to by `p`

`(*p).phases` — is the number of phases of the motor pointed to by `p`

# Mengakses elemen struct menggunakan pointer

- Notasi `(*p).member` kurang nyaman dipakai
- Cara yang lebih singkat
  - `p->member`, di mana `p` merupakan variabel pointer

# Contoh sebelumnya menjadi ...

- Deklarasi

`motor *p;`

- Maka

`p -> volts` — is the voltage of the motor pointed to by `p`

`p -> phases` — is the number of phases of the motor pointed to by `p`

# contoh

```
struct motor {  
    float volts;  
    float amps;  
};  
  
typedef struct motor motor;  
void main()  
{  
    motor ml;  
    motor *pml;  
  
    ml.volts = 100;  
    ml.amps = 110;  
  
    pml = &ml;  
  
    printf("voltase motor ml : %f\n", ml.volts);  
    printf("amps motor ml : %f\n", ml.amps);  
    printf("voltase motor ml : %f\n", pml->volts);  
    printf("amps motor ml : %f", pml->amps);  
    getch();  
}
```

# Hasil eksekusi program



A screenshot of a Windows command prompt window. The title bar is blue and contains the text "C:\VC5\BIN\NONAME00.exe" followed by standard window control buttons (minimize, maximize, close). The command prompt area has a black background with white text. The text shows the output of a program, displaying voltage and current values for a motor named m1. The output consists of four lines: "voltase motor m1 : 100.000000", "amps motor m1 : 110.000000", "voltase motor m1 : 100.000000", and "amps motor m1 : 110.000000". A vertical scrollbar is visible on the right side of the command prompt window.

```
C:\VC5\BIN\NONAME00.exe
voltase motor m1 : 100.000000
amps motor m1 : 110.000000
voltase motor m1 : 100.000000
amps motor m1 : 110.000000
```

# Operasi pada struct

- Copy/assign

```
struct motor p, q;  
p = q;
```

- Get address

```
struct motor p;  
struct motor *s  
s = &p;
```

- Access members

```
p.volts;  
s -> amps;
```

# Example

```
struct item {  
    char *s;  
    struct item *next;  
}
```

Yes! This is legal!



- Sebuah item dapat berisi alamat item lain...
- ... yang dapat menunjuk item lain
- ... yang juga dapat menunjuk item yang lain lagi
- ... etc.

Dengan demikian membentuk rangkaian item!!!