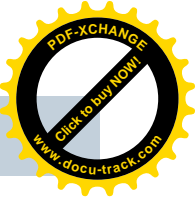




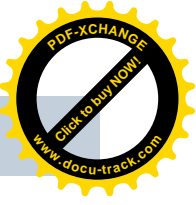
Bab 15. Struktur 2

Konsep Pemrograman
Politeknik Elektronika Negeri Surabaya
2006



Overview

- Struktur dan Fungsi
- Melewatkan Elemen Struktur ke dalam Fungsi
- Melewatkan Struktur ke dalam Fungsi
 - Melewatkan Elemen Struktur ke dalam Fungsi
 - Pass by value
 - Pass by reference
 - Melewatkan Struktur ke dalam Fungsi
 - Pass by value
 - Pass by reference (*pointer to struct*)



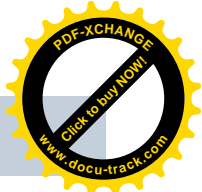
Struktur dan Fungsi

- Melewatkan sebuah struktur untuk menjadi parameter dalam sebuah fungsi dapat dilakukan sama dengan pengiriman parameter berupa variabel biasa.
- Fungsi yang mendapat kiriman parameter tersebut juga bisa mengirimkan hasil baliknya yang juga berupa sebuah struktur (*pass by reference*).



Melewatkan Elemen Struktur ke dalam Fungsi

- Melewatkan parameter berupa elemen struktur dapat dilakukan sebagaimana pengiriman parameter berupa variabel biasa, dapat dilakukan baik secara :
 - Pengiriman nilai (*pass by value*)
 - Pengiriman acuan/address (*pass by reference*).



Melewatkan Elemen Struktur dalam Fungsi: *pass by value* (cetak1.c)

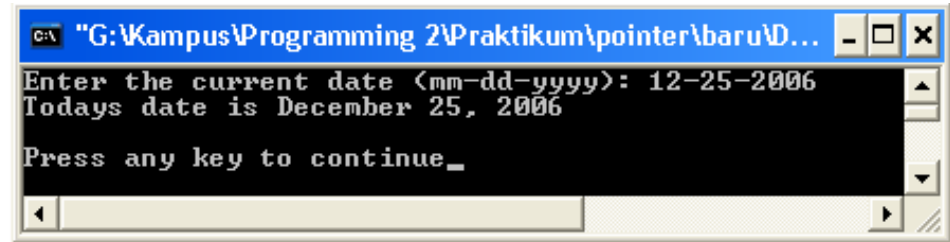
```
#include <stdio.h>
void cetak_tanggal(int, int, int);

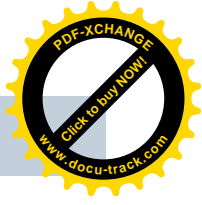
main() {
    struct date {
        int month, day, year;
    } today;

    printf("Enter the current date (mm-dd-yyyy): ");
    scanf("%d-%d-%d", &today.month, &today.day, &today.year);

    cetak_tanggal(today.month, today.day, today.year);
}

void cetak_tanggal(int mm, int dd, int yy){
    char *nama_bulan[] = {
        "Wrong month", "January", "February", "March",
        "April", "May", "June", "July", "August",
        "September", "October", "November", "December"
    };
    printf("Todays date is %s %d, %d\n\n", nama_bulan[mm], dd, yy);
}
```



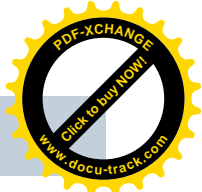


Melewatkan Elemen Struktur ke dalam Fungsi

- Tampak bahwa elemen dari struktur dilewatkan ke fungsi memakai bentuk pengaksesan elemen struktur, berupa :

```
cetak_tanggal(today.month, today.day, today.year) ;
```

- Apabila nilai suatu elemen struktur diharapkan akan diubah oleh fungsi, maka yang dilewatkan haruslah berupa alamat dari elemen struktur (*pass by reference*).
- Untuk keperluan ini, operator alamat ditempatkan di depan nama variabel struktur (bukan di depan nama elemen struktur).



Melewatkan Elemen Struktur ke dalam Fungsi: *pass by reference* (posisi1.c)

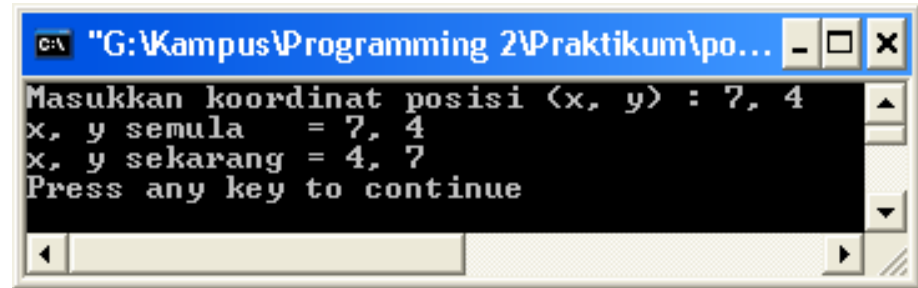
```
#include <stdio.h>
void tukar_xy(int *, int *);

main() {
    struct koordinat {
        int x, y;
    } posisi;

    printf("Masukkan koordinat posisi (x, y) : ");
    scanf("%d, %d", &posisi.x, &posisi.y);
    printf("x, y semula    = %d, %d\n", posisi.x, posisi.y);
    tukar_xy(&posisi.x, &posisi.y);
    printf("x, y sekarang = %d, %d\n", posisi.x, posisi.y);
}

void tukar_xy(int *a, int *b) {
    int z;

    z = *a;
    *a = *b;
    *b = z;
}
```



Melewatkan Struktur ke dalam Fungsi

- Pada program `cetak1.c` di atas misalnya, semua elemen dari struktur dikirimkan ke fungsi `cetak_tanggal()`, dengan maksud nilai elemen dari struktur akan ditampilkan di layar.
- Untuk keadaan seperti ini (semua elemen dikirim sebagai parameter), lebih baik kalau parameter fungsi diubah menjadi bentuk struktur, sehingga parameter fungsi tidak lagi sebanyak tiga buah, melainkan hanya satu yaitu variabel strukturnya



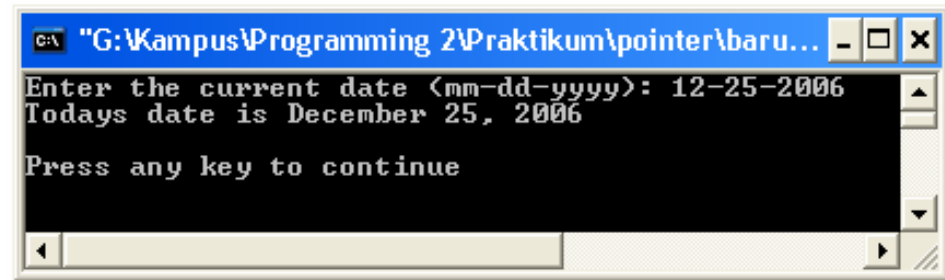
Melewatkan Struktur ke dalam Fungsi : *pass by value* (cetak2.c)

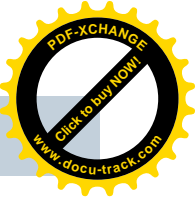
```
#include <stdio.h>
struct date {
    int month, day, year;
};
void cetak_tanggal(struct date);
main() {
    struct date today;

    printf("Enter the current date (mm-dd-yyyy): ");
    scanf("%d-%d-%d", &today.month, &today.day, &today.year);
    cetak_tanggal(today);
}

void cetak_tanggal(struct date now){
    char *nama_bulan[] = {
        "Wrong month", "January", "February", "March", "April", "May", "June",
        "July", "August", "September", "October", "November", "December"
    };

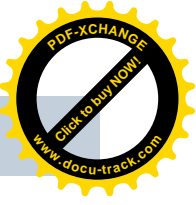
    printf("Todays date is %s %d, %d\n\n",
        nama_bulan[now.month], now.day, now.year);
}
```





Melewatkan Struktur ke dalam Fungsi: *pass by reference à pointer to struct*

- Jika sebuah struktur mengandung banyak *field* dan diputuskan bahwa keseluruhan *field*-nya akan diubah oleh fungsi, maka cara yang efisien adalah dengan melewati alamat dari struktur (*pass by reference*).
- Dengan demikian pada pendefinisian fungsi, parameter formalnya berupa pointer yang menunjuk ke struktur (*pointer to struct*).



Melewatkan Struktur ke dalam Fungsi : *pass by reference* (posisi2.c)

```
#include <stdio.h>
struct koordinat {
    int x, y;
};

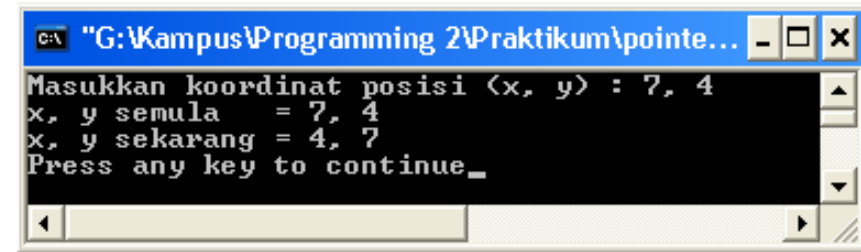
void tukar_xy(struct koordinat *);

main() {
    struct koordinat posisi;

    printf("Masukkan koordinat posisi (x, y) : ");
    scanf("%d, %d", &posisi.x, &posisi.y);
    printf("x, y semula    = %d, %d\n", posisi.x, posisi.y);
    tukar_xy(&posisi);
    printf("x, y sekarang = %d, %d\n", posisi.x, posisi.y);
}

void tukar_xy(struct koordinat *pos_xy) {
    int z;

    z = (*pos_xy).x;
    (*pos_xy).x = (*pos_xy).y;
    (*pos_xy).y = z;
}
```





Melewatkan Struktur ke dalam Fungsi : *pass by reference*

- Parameter dari fungsi `tukar_xy()` disederhanakan menjadi satu parameter saja, yakni sebagai berikut :

```
void tukar_xy(struct koordinat *pos_xy) { }
```

- Pada definisi fungsi di atas,

```
struct koordinat *pos_xy
```

menyatakan bahwa `pos_xy` adalah pointer yang menunjuk ke obyek bertipe `struct koordinat`.

- Adapun penulisan :

```
(*pos_xy).x
```

menyatakan : elemen/field bernama `x` yang ditunjuk oleh pointer `pos_xy`

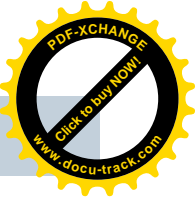
- Perlu diperhatikan bahwa penulisan tanda kurung seperti pada contoh `(*pos_xy).x` merupakan suatu keharusan, sebab

```
*pos_xy.x
```

mempunyai makna yang berbeda dengan

```
(*pos_xy).x
```

- Ungkapan `*pos_xy.x` mempunyai makna yaitu : "yang ditunjuk oleh **`pos_xy.x`** " (sebab operator titik mempunyai prioritas yang lebih tinggi daripada operator `*`).



Melewatkan Struktur ke dalam Fungsi : *pass by reference*

- Bentuk semacam :
`(*pos_xy) . x`
dapat ditulis dengan bentuk lain menjadi
`pos_xy->x`
- Dalam C operator `->` (berupa tanda minus - diikuti dengan tanda lebih dari >) disebut sebagai **operator panah**.
- Dengan menggunakan operator panah, maka fungsi `tukar_xy()` dalam program `posisi2.c` dapat ditulis menjadi

```
void tukar_xy(struct koordinat *pos_xy) {  
    int z;  
  
    z = pos_xy->x;  
    pos_xy->x = pos_xy->y;  
    pos_xy->y = z;  
}
```