PRAKTIKUM 14

STRUKTUR 2

A. Tujuan

1. Struktur dan Fungsi
2. Melewatkan Elemen Struktur ke dalam Fungsi
3. Melewatkan Struktur ke dalam Fungsi
4. Mengerti tentang penggunaan Struktur pada Pointer *(pointer to struct)*

B. DASAR TEORI

Struktur dan Fungsi

Melewatkan sebuah struktur untuk menjadi parameter 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 nilai *(pass by value)* maupun secara acuan *(pass by reference)*.

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

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

void cetak_tanggal(int mm, int dd, int yy)
{
 static 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",
    nama_bulan[mm],dd,yy);
   }
```

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).

```
main()
{
   struct koordinat {
       int x;
       int y;
   } posisi;
   ...
   tukar_xy(&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 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, lebih baik kalau parameter fungsi diubah menjadi bentuk struktur, sehingga parameter fungsi tidak lagi sebanyak tiga buah, melainkan hanya satu. Selengkapnya, perhatikan program di bawah ini.

```
struct date
{
   int month;
   int day;
   int year;
};

void cetak_tanggal(struct date);
main()
{
   struct date today;
   ...
   cetak_tanggal(today);
   ...
}

void cetak_tanggal(struct date now)
{
   static 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);
}
```

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

Masalah pointer ke struktur dapat diterapkan dalam program sebelumnya. Argumen dari fungsi **tukar_xy()** dapat disederhanakan menjadi satu argumen saja, yakni sebagai berikut :

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

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

Pada definisi fungsi di atas,

```
struct koordinat *pos_xy
```

menyatakan bahwa **pos_xy** adalah pointer yang menunjuk ke obyek bertipe struktur **koordinat**. Adapun penulisan :

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

menyatakan : elemen 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 *).

```
#include <stdio.h>

struct koordinat
     { int x;
       int 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;
}
```

Contoh eksekusi :

```
Masukkan koordinat posisi (x, y) : 34, 21
x, y semula   = 34, 21
x, y sekarang = 21, 34
```

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;
}
```

C. TUGAS PENDAHULUAN

1. Buatlah program sebagai berikut dengan menggunakan array dari struktur:

   Input:

   Judul buku

   Jumlah

Harga_satuan

Lakukan pengisian secara langsung untuk, field field yang ada sehingga didapatkan tampilan seperti tampilan dibawah ini ;

Tampilan :

```
Judul buku : matematika, jumlah:5, harga satuan 2000
Judul buku : elektromagnetika, jumlah : 4, harga satuan
4000
```

D. PERCOBAAN

1. Buatlah program sebagai berikut dengan menggunakan array of struct :

   Input:

   Nama Mhs

   Nilai Tugas

   Nilai UTS

   Nilai UAS

   Pengisian nilai dilakukan secara langsung

   Proses:

   Nilai Akhir= 20% tugas + 40% UTS + 40%UAS

   Nilai huruf A(81-100), AB(71-80), B(65-70), BC(61-64), C(56-60), D(40-55), E(0-39)

   Nilai dinyatakan lulus jika minimal C

   Output :

```
Nama : andika
Nilai tugas : 60
Nilai UTS : 80
Nilai UAS : 80
Nilai akhir : 76
Nilai Huruf : AB

Nama : cita
Nilai tugas : 80
Nilai UTS : 100
Nilai UAS : 80
```

```
Nilai akhir : 88
Nilai Huruf : A
```

2. Ubahlah program no. 1 kedalam bentuk fungsi!

3. Dengan menggunakan fungsi buatlah fasilitas untuk menampilkan data mahasiswa yang mempunyai nilai UAS tertinggi!

4. Dengan menggunakan fungsi buatlah fasilitas untuk menampilkan data mahasiswa yang mempunyai nilai Akhir tertinggi (dengan rumus 20% tugas + 40%UTS + 40 %UAS)

E. LAPORAN RESMI

Kerjakan soal tugas pendahuluan dalam bentuk fungsi!