Pertemuan 14

Struktur, Struktur dan Fungsi, Struktur & Pointer

Mata Kuliah : Algoritma & Pemrograman

Dosen : Tessy Badriyah, SKom., MT., PhD.

Pembahasan

- Pembahasan tentang tipe data Struktur
- Penggunaan Struktur dalam Fungsi
- Hubungan antara Struktur dan Pointer

Pokok Bahasan

- Definisi dan deklarasi struktur
- Deklarasi variable bertipe struktur
- Struktur di dalam struktur
- Mengakses elemen struktur
- Pembentukan array dari struktur
- Struktur dengan Fungsi
- Struktur dengan Pointer

Definisi dan deklarasi struktur

- Struktur adalah pengelompokan variable-variable yang memiliki tipe data yang tidak harus sama tapi semuanya memiliki satu fungsi/tujuan tertentu
- Contoh struktur : untuk menyimpan informasi tahun kelahiran
  - Struktur terdiri dari tanggal, bulan dan tahun
  - Deklarasi struktur

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

Deklarasi variable bertipe struktur

- Misal terdapat deklarasi struktur berikut :

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

- Kemudian kita deklarasikan variable borndate yang memiliki tipe data seperti struktur diatas maka :

  struct tgl_lahir borndate;

Struktur berisi struktur

- Struktur tanggal lahir (tgl_lahir) :

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

- Struktur student

```
struct student {
char name[30];
struct tgl_lahir borndate;
};
```

Mengakses elemen struktur

- Elemen dari suatu variabel struktur dapat diakses dengan menyebutkan nama variabel struktur diikuti dengan operator titik (‘.’) dan nama dari elemen strukturnya.
- Cara penulisan (sintak) :

  variabel_struktur.nama_field

Mengakses elemen struktur

- Misal kita punya definisi berikut :

```
struct tgl_lahir {
int tanggal;
int bulan;
int tahun;
};
struct student {
char name[30];
struct tgl_lahir borndate;
};
struct student mhs;
```

- Maka untuk mengisi field name pada variable mhs :

```
strcpy(mhs.name, “Tessy Badriyah”);
```

Contoh program struktur (1)

```
/* Mengisi field dr variabel struktur kemudian menampilkannya */
#include <stdio.h>
#include <string.h>
#include <conio.h>
struct tgl_lahir { /* definisi global dari tipe date */
int tanggal;
int bulan;
int tahun;
};
struct student{ /* definisi global dari tipe student */
char name[30];
struct tgl_lahir borndate;
};
/* deklarasi global dari variabel mhs*/
struct student mhs;
```

Contoh program struktur (2)

```
main()
{
  /* memberikan nilai kepada field dari struktur mhs */
  strcpy(mhs.name, "TESSY BADRIYAH");
  mhs.borndate.tanggal = 14;
  mhs.borndate.bulan = 9;
  mhs.borndate.tahun = 1970;
  /* menampilkan isi semua field dari struktur mhs */
  printf("Name : %s\n", mhs.name);
  printf("Tanggal Lahir : %d-%d-%d\n",mhs.borndate.tanggal,
  mhs.borndate.bulan, mhs.borndate.tahun);
  getch();
}
```

Inisialisasi variable struktur

```
int bulan;
int tahun;
};
struct student{ /* definisi global dari tipe student */
char name[30];
struct tgl_lahir borndate;
};
main()
{
  /* inisialisasi variabel struktur */
  static struct student mhs={"TESSY BADRIYAH",14,9,1970};
  /* menampilkan isi semua field dari struktur mhs */
  printf("Name : %s\n", mhs.name);
  printf("Tanggal Lahir : %d-%d-%d\n",mhs.borndate.tanggal,
  mhs.borndate.bulan, mhs.borndate.tahun);
  getch();
}
```

Pembentukan array dari struktur

- Elemen-elemen dari suatu array juga dapat berbentuk sebuah struktur. Misalnya
- array yang dipakai untuk menyimpan sejumlah data siswa *(struct student).*

- #define MAKS 20

```
/* Mengisi field dr variabel struktur kemudian menampilkannya */
#include <stdio.h>
#include <string.h>
#include <conio.h>
struct tgl_lahir { /* definisi global dari tipe date */
int tanggal;
int bulan;
int tahun;
};
struct student{ /* definisi global dari tipe student */
char name[30];
struct tgl_lahir borndate;
};
/* deklarasi global dari variabel mhs*/
struct student mhs[30];
```

Struktur dengan 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)*.

Contoh program struktur dengan fungsi (1)

```
/* File program : cetak1.c
Melewatkan elemen struktur sbg parameter fungsi scr nilai */
#include <stdio.h>
void cetak_tanggal(int, int, int);
main()
{
struct date { /* definisi lokal dari tipe date */
int month;
int day;
int 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);
}
```

Contoh program struktur dengan fungsi (2)

```
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\n",
nama_bulan[mm],dd,yy);
}
```

Cara lain (1)

```
/* File program : cetak2.c
Melewatkan struktur sebagai parameter fungsi */
#include <stdio.h>
struct date { /* definisi global dari tipe date */
int month;
int day;
int 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);
}
```

Cara lain(2)

- 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);
- }

Struktur dengan Pointer

- 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.

Struktur dengan pointer (1)

```
/* File program : posisi2.c
Fungsi parameternya berupa pointer yg menunjuk ke struktur */
#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);
}
```

Struktur dengan pointer (2)

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

Yang sudah dipelajari

- Pembahasan tentang tipe data Struktur
- Penggunaan Struktur dalam Fungsi
- Hubungan antara Struktur dan Pointer

Referensi

- Robertson, Lesley Anne. (1992). *Students' guide to program design*. Oxford : Newnes
- Santner, Williams, and Notz (2003), Design and Analysis of Computer Experiments, Springer.
- Deitel & Deitel, *C How to Program*, Prentice Hall 1994 (2nd edition)
- Brookshear, J.G., *Computer Science: An Overview*, Benjamin-Cummings 2000 (6th edition)
- Kernighan & Ritchie, *The C Programming Language*, Prentice Hall