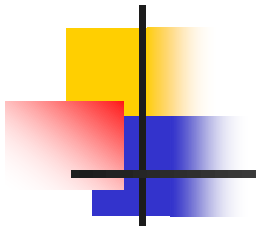
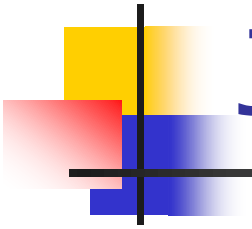
Fungsi Lanjutan

Tujuan



- Mengerti variabel dalam Fungsi
  a. Variabel lokal
  b. Variabel eksternal
  b. Variabel statis
  c. Variabel register
- Memahami dalam menciptakan sejumlah fungsi.



Jenis-jenis Variable

- Jenis-jenis variable pada C
  - Variabel lokal
  - Variabel eksternal
  - Variabel statis
  - Variabel register

Variabel Lokal

- Hanya dikenal didalam fungsi tempat variabel tersebut dideklarasikan. Setelah fungsi selesai dijalankan, variabel lokal tersebut otomatis akan hilang.
- Tidak ada inisialisasi secara otomatis (saat variabel diciptakan, nilainya tak menentu).

Contoh Variabel Lokal

```
#include <stdio.h>
void fung_1(void);
main()
{
   int i = 20;
   fung_1();
   printf("nilai i di dalam main()    = %d\n", i);
}
void fung_1(void)
{
   int i = 11;
   printf("nilai i di dalam fung_1() = %d\n", i);
}
```

Variabel Eksternal

- Dapat diakses oleh semua fungsi
- Kalau tak diberi nilai, secara otomatis diinisialisasi dengan nilai sama dengan nol.

Contoh#1 Variabel Eksternal

```
#include <stdio.h>
int i = 273;          /* variabel eksternal */
void tambah(void);
main()
{
   printf("Nilai awal i = %d\n", i);
   i += 7;
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
}
void tambah(void)
{
   i++;
}
```

Contoh#2 Variabel Eksternal

```
#include <stdio.h>
int i = 273;          /* variabel eksternal */
void tambah(void);
main()
{
   extern int i;      /* variabel eksternal */
   printf("Nilai awal i = %d\n", i);
   i += 7;
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
}
void tambah(void)
{
   extern int i;      /* variabel eksternal */
   i++;
}
```

Contoh#3 Variabel Eksternal

```
#include <stdio.h>
int i = 273;          /* variabel eksternal */
void tambah(void);
main()
{
   extern int i;      /* variabel eksternal */
   printf("Nilai awal i = %d\n", i);
   i += 7;
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
   tambah();
   printf("Nilai i kini = %d\n", i);
}
void tambah(void)
{
   int i;      /* variabel lokal */
   i++;
}
```

Variabel Static

- **Kalau variabel statis bersifat internal, maka variabel hanya dikenal oleh fungsi tempat variabel dideklarasikan**
- **Kalau variabel statis bersifat eksternal, maka variabel dapat dipergunakan oleh semua fungsi yang terletak pada file yang sama, tempat variabel statis dideklarasikan**
- **Berbeda dengan variabel lokal, variabel statis tidak akan hilang sekeluarnya dari fungsi (nilai pada variabel akan tetap diingat).**
- **Inisialisasi akan dilakukan hanya sekali, yaitu saat fungsi dipanggil yang pertama kali. Kalau tak ada inisialisasi oleh pemrogram secara otomatis akan diberi nilai awal nol**

Contoh Variabel Static

```
#include <stdio.h>
void fung_y(void);
main()
{
   int y = 20;
   fung_y();
   fung_y();
   printf("Nilai y dalam main()    = %d\n", y);
}
void fung_y(void)
{
   static int y;
   y++;
   printf("Nilai y dalam fung_y() = %d\n", y);
}
```

Variabel Register

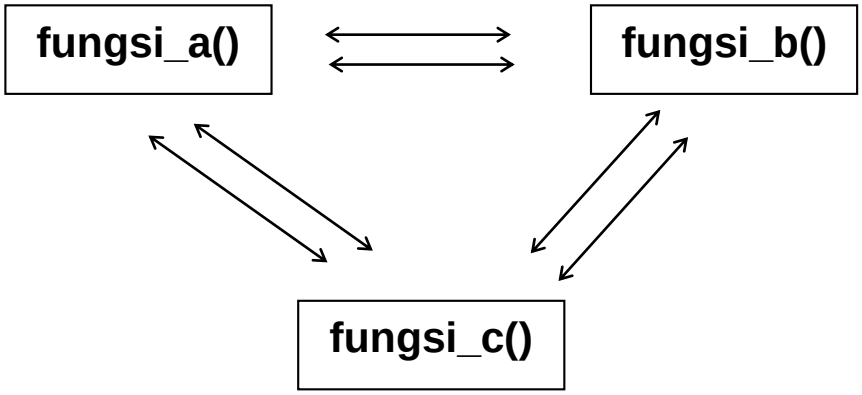
- Variabel register adalah variabel yang nilainya disimpan dalam register dan bukan dalam memori RAM, sehingga mempercepat proses.
- Variabel yang seperti ini hanya bisa diterapkan pada variabel yang lokal atau parameter formal, yang bertipe *char* atau *int*.
- Variabel register biasa diterapkan pada variabel yang digunakan sebagai pengendali *loop*.

Contoh#1 Variabel Eksternal

```
#include <stdio.h>
main()
{
  register int i;/* variabel register */
  int jumlah = 0;
  for(i = 1; i <= 100; i++)
      jumlah = jumlah + i;
  printf("1 + 2 + 3 + ... + 100 = %d\n",
  jumlah);
}
```

Menciptakan sejumlah Fungsi

- Pada C, semua fungsi bersifat sederajat.
- Suatu fungsi tidak dapat didefinisikan di dalam fungsi yang lain. Akan tetapi suatu fungsi diperbolehkan memanggil fungsi yang lain, dan tidak tergantung kepada peletakan definisi fungsi pada program.



Contoh# Fungsi dalam fungsi

```
#include <stdio.h>
void fungsi_1(void);
void fungsi_2(void);
main()
{
   fungsi_1();
}
void fungsi_1()
{
   puts("fungsi 1 dijalankan");
   fungsi_2();
}
void fungsi_2()
{
   puts("fungsi 2 dijalankan");
}
```