# FUNGSI

Suatu program merupakan kumpulan dari pernyataan - pernyataan. Kadangkala pernyataan - pernyataan tersebut diperlukan berulang - ulang pada beberapa tempat di program yang sama. Pengulangan pernyataan - pernyataan tersebut membuat program menjadi tidak efisien. dengan pembuatan suatu fungsi, maka pengulangan pernyataan - pernyataan tsb tidak perlu terjadi, karena kita dapat memanggil kembali fungsi tsb.
Suatu fungsi dapat dibuat di dalam program yang sama ( fungsi internal) maupun di file lain ( fungsi eksternal ).
Dari dalam program yang sama, digunakan tanda '=' untuk memberikan nilai ke suatu variabel. Kalimat utk memberikan nilai ke sebuah variabel disebut 'kalimat penilaian' atau assignment.
dari luar program, pemberian nilai pada suatu variabel dilakukan dengan menggunakan fungsi masukan (input function).
Fungsi dlm bahasa C adalah subprogram. Program yang ditulis dengan C tdd fungsi - fungsi. Fungsi minimal yang harus tdp dalam suatu program adl main() dan muncul hanya sekali.
Pengertian fungsi dalam Turbo C mirip dengan subroutine dalam bahasa BASIC atau fungsi dan procedur dalam PASCAL

## SIFAT - SIFAT DAN MANFAAT FUNGSI

1. Menghindari penulisan program yang berulang
2. Fungsi akan membagi program menjadi modul - modul yang lebih kecil (modular design) , sehingga jika terjadi kesalahan dalam program akan lebih mudah dilacak.
3. Setiap fungsi memiliki tingkatan yang sama dan berdiri sendiri. Dengan demikian fungsi - fungsi ini dapat dipanggil dari sembarang fungsi yang lain dengan mudah. Dalam Turbo C tidak diperkenankan sebuah fungsi ada dalam fungsi yang lain (nested function)
4. Fungsi dalam C dapat menghasilkan nilai, data ataupun hasil atau kegiatan lain yang bukan berupa data maupun nilai. Turbo C tidak membedakan fungsi yang menghasilkan nilai ( fungsi) dan fungsi yg menghasilkan data (prosedur).

## STRUKTUR SEBUAH FUNGSI SEDERHANA

```
BU :    tipe nama fungsi (parameter)
        {
                deklarasi variabel lokal
                pernyataan - pernyataan
                return (ungkapan)
        }
```

nama fungsi mempunyai ketentuan yang serupa dengan nama variabel.

parameter diletakkan diantara tanda (), di belakang nama fungsi.Parameter boleh diisi dengan data atau dibiarkan kosong. Bila diisi dengan data maka data tsb akan dikirimkan ke fungsi yang bersangkutan untuk diolah. Jika fungsi ybs tidak memerlukan data dari luar , maka parameter dibiarkan kosong.

pernyataan pada fungsi harus diletakkan diantara tanda {}

Contoh :

```
main()
{
  int x,y;
  x=contoh();
  printf("x= %d\n",x);
  y=contoh();
  printf("y= %d\n",y);
}

contoh()
{
   return(0);
}
```

OUTPUT :

```
   x=0
   y=0
```

**FUNGSI main()**

Dalam C fungsi main() mutlak harus ada, karena dari fungsi inilah program akan dimulai. Dalam sebuah program yang tdd beberapa fungsi, main() boleh diletakkan dimana saja, sebaiknya fungsi main() diletakkan paling atas.

Dalam mendefinisikan sebuah fungsi akan dijumpai :

1. Parameter fungsi (function identifier)
   - nama fungsi
2. parameter formal
   - daftar variabel yg dipakai utk berkomunikasi dg fungsi lain ( yi fungsi pemanggil)
3. Variabel lokal
   - daftar variabel yang hanya dipakai di dalam fungsi
4. Isi fungsi
   - tubuh fungsi

***Membuat dan Menggunakan Fungsi sederhana***

Membuat sebuah fungsi yg mencetak karakter asterik sebanyak 65 dlm satu baris

```
#define NAMA "UNIVERSITAS GUNADARMA"
#define ALAMAT  "Jl.Margonda Raya"
#define TEMPAT "Depok"
main()
{
        starbar();
        printf ("%s\n",NAMA);
        printf ("%s\n",ALAMAT);
        printf ("%s\n",TEMPAT);
        starbar();
}

/* fungsi starbar() */
#include <stdio.h>
#define BATAS 65
starbar()
{
        int hitung;
        for (hitung = 1; hitung <= BATAS ; hitung++)  putchar('*');
        putchar('\n');
}
```

Fungsi starbar() dipanggil dari fungsi main() tanpa menggunakan argumen. Karena fungsi starbar() tidak mempunyai parameter.

Kita dapat menjadikan starbar() dan main() berada dalam satu file atau terpisah. Jika berada dalam satu file memudahkan dalam proses compiling. Jika berada dalam file yang terpisah, fungsi starbar() dapat digunakan oleh program lain.

**FUNGSI SEDERHANA YANG TIDAK MENGHASILKAN NILAI**

Contoh :

```
main()
{
        clrscr();
        printf ("\ngambar sebuah kotak yang dibuat dengan sebuah fungsi\n");
        kotak();
        printf ('\ntekan sembarang tombol, akan terlihat sebuah kotak lagi\n");
        getch();
        kotak();
        printf ("\ntekan semabarang tombol untuk berhenti);
        getch();
}
```

```
kotak()
{
        int i,j;
        printf ("\n");
        for (i=1;i<=5;++i)
        {
                for (j=1;j<=10;++j)
                        printf("\n");
        }
}
```

Fungsi kotak() digunakan untuk menampilkan gambar kotak di layar. Argumen dibiarkan kosong karena fungsi ini tidak memerlukan data dari luar.

**FUNGSI YANG MENGHASILKAN NILAI**

Contoh :

```
main()
{
        float luas_bs;
        clrscr();
        printf ("Program menghitung bujursangkar");
        luas_bs = luas();
        printf ("\n\nLuas Bujursangkar = %10.2f",luas_bs);
}
luas()
{
        float sisi, luas1;
        printf ("\nsisi bujursangkar = ");
        scanf ("%f",&sisi);
        luas1 = sisi * sisi;
        return(luas);
}
```

Fungsi luas() diatas menhasilkan nilai yang disimpan dalam suatu variabel ttt luas_bs yang terletak pada fungsi main().

**PERNYATAAN return**

digunakan untuk mengirimkan hasil / nilai dari suatu fungsi ke fungsi lain yang memanggilnya. return diikuti oleh argumen yang berupa nilai yang akan dikirim. Pernyataan return selalu digunakan oleh fungsi yang menghasilkan nilai.

Contoh : return(luas1)
nilai dari luas1 akan dikirim kembali ke fungsi yang memanggilnya, yaitu fungsi main() yang ditampung dalam variabel luas_bs

Argumen dari return boleh juga berupa rumus :

```
luas1 = sisi*sisi;
return(luas1);
```

dapat disederhanakan lagi menjadi :

```
return(sisi*sisi);
```

pernyataan return ini selalu digunakan oleh fungsi yang menghasilkan nilai

**MENGIRIMKAN DATA KONSTANTA KE SUATU FUNGSI**

CONTOH :

```
main()
{
        float luas_bs;
        clrscr();
        printf ("Program menghitung bujursangkar");
        luas_bs = luas(10.0);
        printf ("\n\nLuas Bujursangkar = %10.2f",luas_bs);
}

luas(sisi)
float sisi;
{
        return(sisi*sisi);
}
```

Tipe variabel sisi perlu ditentukan dan deklarasinya harus ditulis dibawah nama fungsi:luas(sisi)

```
        float sisi;
```

Deklarasi variabel argumen harus diletakkan diluar fungsi, seperti :

```
luas(sisi)
float sisi;
{
        pernyataan;
        pernyataan;
}
```

Bila deklarasi suatu variabel dilakukan didalam fungsi , maka variabel tadi akan diperlakukan sebagai variabel lokal, sebaliknya bila deklarasi dilakukan diluar fungsi, namun masih di bawah nama fungsi, maka deklarasi ini akan dianggap sebagai deklarasi terhadap variabel argumen.

**MENGIRIMKAN DATA VARIABEL KE SUATU FUNGSI**

Contoh :

```
main()
{
        float luas_bs,sisi_bs;
        clrscr();
        printf ("Program menghitung bujursangkar");
        printf ("\nsisi bujursangkar = ");
        scanf ("%f",&sisi);
        luas_bs = luas(sisi_bs);
        printf ("\n\nLuas Bujursangkar = %10.2f",luas_bs);
}
luas(sisi)
float sisi;
{
        return(sisi*sisi);
}
```

Pernyataan luas_bs = luas(sisi_bs);
artinya mengirimkan nilai variabel sisi_bs ke fungsi luas() untuk dioleh dan hasilnya akan diterima oleh variabel luas_bs yg terdapat dalam fungsi main(). Pada fungsi luas(), nilai variabel sisi_bs ini akan ditampung dalam variabel sisi.

2 tahap pembuatan sebuah fungsi :
1. Membuat prototype fungsi , prototype ditulis di luar blok fungsi
2. Mendefinisikan fungsi itu sendiri, yaitu menuliskan kode programnya sehingga jika dipanggil fungsi akan bekerja seperti yang diinginkan.

**ARGUMEN TERDIRI LEBIH DARI SATU DATA**

Contoh :

```
main()
{
        float panjang_se,lebar_se,luas_se;
        clrscr();
        printf ("Program menghitung luas segiempat");
        printf ("\n\tPanjang  = ");
        scanf ("%f",&panjang_se);
        printf ("\n\tLebar    = ");
        scanf ("%f",&lebar_se);
        luas_se = luas(panjang_se,lebar_se);
        printf ("\n\nLuas Segiempat = %10.2f",luas_se);
}
```

```
luas(panjang,lebar)
float panjang,lebar;
{
        return(panjang*lebar);
}
```

**PROTOTYPE FUNGSI**

prototype fungsi sebenarnya merupakan deklarasi fungsi yang dilengkapi dengan nama beserta tipe variabel argumennya. maksud dari penggunaan prototype fungsi adalah untuk mencegah terjadinya kesalahan tipe data pada waktu mengirimkan data dari sebuah fungsi ke fungsi lainnya melalui argumen. Prototype diberikan pada waktu fungsi tsb dideklarasikan maupun pada waktu menuliskan nama fungsinya sendiri.
Prototype fungsi menjelaskan 3 hal :

- tipe keluaran fungsi
- jumlah parameter
- tipe dari masing - masing parameter

Deklarasi fungsi dengan prototype

```
float luas(float panjang, float lebar)
main()
{
        .....dst
```

| | |
|---|---|
| float | : tipe keluaran fungsi |
| luas | : nama fungsi |
| float panjang dan float lebar | : tipe parameter 1dan tipe parameter 2 |

Fungsi yang dilengkapi dengan prototype fungsi

```
float luas(float panjang, float lebar)
{
        .....dst
```

pada fungsi tsb tidak lagi diperlukan deklarasi terhadap variabel yg terdapat dalam argumen, karena deklarasi ini sudah tercakup dalam prototype.

## DEKLARASI FUNGSI

Hasil suatu fungsi akan dinyatakan dalam bentuk integer. Untuk memperoleh hasil yang benar , maka fungsi tsb harus dideklarasikan . Hal - hal yang perlu diperhatikan dalam mendeklarasikan suatu fungsi :

1. BU deklarasi suatu fungsi :
   tipe namafungsi();
   Contoh : fungsi luas() akan didkelarasikan sebagai float.
   float luas();

2. Beberapa fungsi yang bertipe sama dapat dideklarasikan bersama - sama.
   Contoh : fungsi luas() & kell() akan dideklarasikan sbg float
   float luas), kell();

3.Deklarasi sebaiknya dilakukan di awal program di dekat fungsi main()

4. Deklarasi fungsi sebaiknya dilakukan diluar , sehingga fungsi tadi akan dikenal oleh semua fungsi lainnya.

5. Tipe dari fungsi harus dicantumkan di depan nama fungsi pada fungsi yang bersangkutan.
   float luas(panjang,lebar)
   float panjang_lebar;
   {
   .....dst

## VARIABEL LOKAL

- Fungsi dimana variabel lokal tsb dideklarasikan.
- hanya ada bilamana fungsi yg mengandung variabel lokal tsb dipanggil.
- sifatnya dinamis / otomatis
- dideklarasikan di dalam suatu fungsi

Contoh :

```
void fungsi1()
main()
{
  int i = 20;
  fungsi1();
  printf("i dalam main() = % d",i);
}
```

```
void fungsi1()
{
  int i = 10;
  printf("i dalam fungsi1 = %d",i);
}
```

OUTPUT :

```
i dalam main() = 20
i dalam fungsi1() = 10
```

**VARIABEL GLOBAL**

- dideklarasikan diluar semua fungsi
- dikenali oleh semua fungsi
- selalu ada selama program dijalankan
- sifatnya statis

Contoh :

```
int i = 273;
void tambah()
main()
{
  printf("Nilai awal i = % d\n",i);
  i=i+7;
  printf("Nilai i kini = % d\n",i);
  tambah();
  printf("Nilai awal i = % d\n",i);
  tambah();
  printf("Nilai awal i = % d\n",i);
}

void tambah()
{
  i ++;
}
```

OUTPUT

```
Nilai awal i = 273
Nilai i kini   = 280
Nilai i kini   = 281
Nilai i kini   = 282
```

Fungsi - fungsi yang terletak di bawah atau setelah deklarasi variabel global akan mengenal variabel tsb, sedang fungsi - fungsi yang terletak diatasnya tidak.
Bila nama variabel lokal sama dengan nama variabel global maka dalam fungsi yang bersangkutan nama variabel tadi akan diartikan sebagai variabel lokal.