

# Algoritma & Pemrograman

## FUNGSI

**Pengampu : Agus Priyanto, M.Kom**



**INSTITUT TEKNOLOGI TELKOM**  
**Smart, Trustworthy, And Teamwork**



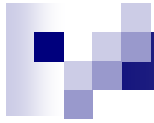
# FUNGSI

- Modul program yang mengembalikan/ memberikan (return) sebuah nilai yang bertipe sederhana.
  - tipe data sederhana : integer, real, boolean, dan string
- Dalam matematika :
  - $f(x,y) = 3x - y + xy$
  - f adalah sebuah fungsi dengan parameter x dan y.
- Nilai yang diberikan fungsi tergantung nilai parameter masukannya.



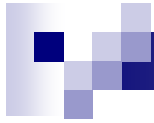
## FUNGSI (*cont.*)

- Fungsi diakses dengan memanggil namanya (sama seperti prosedur).
- Fungsi dapat mengandung parameter formal berjenis parameter masukan.



# Mendefinisikan Fungsi

- Struktur fungsi sama dengan struktur algoritma, yaitu : bagian judul, bagian deklarasi, dan badan fungsi.
- Setiap fungsi memiliki nama unik serta daftar parameter formalnya (jika ada).



- Tipe hasil (pada *header*) menspesifikasikan tipe data dari nilai yang diberikan fungsi.



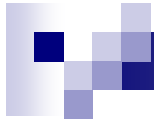
# Struktur Fungsi

```
type_hasil Nama_Fungsi(daftar parameter input formal)
{
    /*Spesifikasi fungsi, berisi penjelasan tentang apa yang
    dilakukan dan yang dikembalikan oleh fungsi ini*/

    /*semua nama yang digunakan dalam fungsi dan hanya
    berlaku lokal di dalam fungsi didefinisikan di sini*/

    /*badan fungsi, berisi kumpulan instruksi*/

    return hasil;    //pengembalian nilai yang dihasilkan fungsi
}
```



# Pemanggilan Fungsi

- Fungsi diakses dengan memanggil namanya dari program pemanggil (program utama atau modul program lain), diikuti dengan daftar parameter aktual (jika ada).



# Pemanggilan Fungsi (*cont.*)

Karena fungsi menghasilkan nilai, maka :

- nilai tersebut dapat ditampung dalam sebuah peubah yang **bertipe sama** dengan tipe fungsi :

Peubah = **NAMA\_FUNGSI(daftar parameter aktual);**

- nilai yang diberikan oleh fungsi dapat langsung dimanipulasi

```
printf("[format]",NAMA_FUNGSI(daftar parameter))
```

```
if (2 * NAMA_FUNGSI(daftar parameter) < 0)
```

```
{
```

```
}
```





## Pemanggilan Fungsi (*cont.*)

- Parameter aktual dapat berupa tetapan, nama tetapan, atau nama peubah asalkan sudah terdefinisi tipe dan harganya.
- Dalam program pemanggil, *prototype* fungsi harus dideklarasikan supaya program pemanggil mengenal nama fungsi tersebut serta cara mengaksesnya.



# Contoh Fungsi & Pemanggilannya 1

```
int pangkat( int y )  
{  
  
    return y* y;  
}
```

```
int main()  
{  
  
    for ( int x = 1; x <= 10; x++ )  
        cout << pangkat( x ) << " ";  
    cout << endl;  
    return 0;  
}
```



# Contoh Fungsi & Pemanggilannya 2

```
float luas_persegi(float p, float t)
{
    /* mengembalikan nilai hasil
       perhitungan luas persegi */

    float hasil;

    hasil = p*t;

    return hasil;
}
```

```
int main()
{

    float luas, panjang, tinggi;

    cout << "Masukan Panjang = ";
    cin >> panjang;
    cout << "Masukan Tinggi = ";
    cin >> tinggi;

    luas =
luas_persegi(panjang, tinggi);
    cout << "Jadi Luasnya = " << luas;

}
```



# Contoh Fungsi & Pemanggilannya 3

```
float PANGKATKAN(float a, int n)
{
    /* mengembalikan harga
       perpangkatan  $a^n$ , n bilangan
       bulat positif */

    float hasil;
    int i;

    hasil = 1;
    for( i=1; i<=n; i++)
        hasil = hasil * a;

    return hasil;
}
```

```
float PANGKATKAN(float a, int n);

Int main()
{
    /* program utama untuk menghitung
       volume bola dengan rumus
        $V = \frac{4}{3} \pi r^3$  */

    float V, r;

    cout<<"Jari-jari : ";
    cin>>r;
    V = 1.33 * 3.14 * PANGKATKAN(r,3);
    cout<<"Volume Bola : "<< V;

}
```



# PERHATIKAN!

- Cara pendefinisian fungsi
- Penggunaan parameter masukan, keluaran, dan masukan/keluaran
- *Parsing parameter*
- Tipe data parameter aktual dan formal
- Nama lokal dan nama global
- Cara pemanggilan fungsi (bandingkan dengan prosedur)



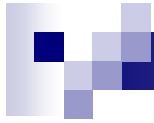
# Prosedur atau Fungsi ?

- Sebuah prosedur dapat dituliskan sebagai fungsi, demikian juga sebaliknya.
- Sebaiknya buat modul program dalam bentuk :
  - **Fungsi**
    - jika modul program menghasilkan **sebuah** nilai.
  - **Prosedur**
    - jika modul menghasilkan **efek netto** dari satu/sekumpulan aksi.
- Modul Program dengan output lebih dari satu tidak elegan dituliskan sebagai fungsi.



# LATIHAN 1

- Buat program untuk menghitung luas permukaan tabung dan volume dengan ketentuan sebagai berikut:
  - $\Phi=3.14$  dijadikan konstanta
  - Jari-jari dan tinggi tabung dimasukkan user di program utama
  - *Belum ada modul*



- ❑ Kembangkan program di atas dengan membuat fungsi untuk bagian program yang menghitung:
  - Luas permukaan
  - Volume
- ❑ Dan buat program utamanya yang memanggil ke-2 fungsi tsb.





# Listing Program

```
float luas_permukaan(float r,float  
    t)  
{  
    float hasil;  
    hasil = 2 * 3.14 * r * ( r + t);  
    return hasil;  
}
```

```
float volume(float r,float t)  
{  
    float hasil;  
    hasil = 3.14 * r * 2 * t;  
    return hasil;  
}
```

```
int main()  
{  
    float luas, vol, jari2, tinggi;  
    cout << "Jari-Jari =";  
    cin>>jari2;  
    cout << "Tinggi =";  
    cin>>tinggi;  
    luas = luas_permukaan(jari2,tinggi);  
    vol = volume (jari2,tinggi);  
    cout << "Luas Permukaan Tabung  
=<<luas<<  
    cout << "Volume Tabung ="<<vol;  
    return 0;  
}
```



# LATIHAN 2

- Nilai dari seorang mahasiswa akan dikonversi dari bentuk angka ke bentuk huruf dengan aturan:

nilai  $\geq 80$  : 'A'

$60 \leq \text{nilai} < 80$  : 'B'

$40 \leq \text{nilai} < 60$  : 'C'

$20 \leq \text{nilai} < 40$  : 'D'

nilai  $< 20$  : 'E'

Buatlah fungsi yang menerima masukan nilai angka dari program utama dan menghasilkan nilai huruf dari mahasiswa tsb.

- Buat program utama yang menerima masukan NIM, nama dan nilai mahasiswa, dan menampilkan keluaran berupa nilai huruf mahasiswa.