

# Konsep Pemrograman

## 7. Fungsi 1

Umi Sa'adah

Entin Martiana Kusumaningtyas

Tri Hadiah Muliawati

2021



Politeknik Elektronika Negeri Surabaya  
Departemen Teknik Informatika dan Komputer

# Overview

- Pendahuluan
- Tujuan Fungsi
- Dasar Fungsi
- Jenis Fungsi :
  - memiliki *return value*
    - Integer
    - Selain integer
  - Tidak memiliki *return value*
- Prototype/Deklarasi Fungsi



# Pendahuluan

- Fungsi adalah :
  - suatu bagian dari program
  - yang dirancang untuk melaksanakan **tugas tertentu**
  - letaknya dipisahkan dari program yang menggunakannya.
- Macam fungsi:
  - **standard** : sudah disediakan oleh compiler, tinggal dipakai dengan menyebutkan headernya (kamusnya) pada preprocessor include, misalnya fungsi :  
`printf()` → `stdio.h`;  
`exit()` → `stdlib.h`
  - **user defined** : **didefinisikan oleh user**,  
disesuaikan dengan kebutuhan user ybs



# Tujuan Fungsi

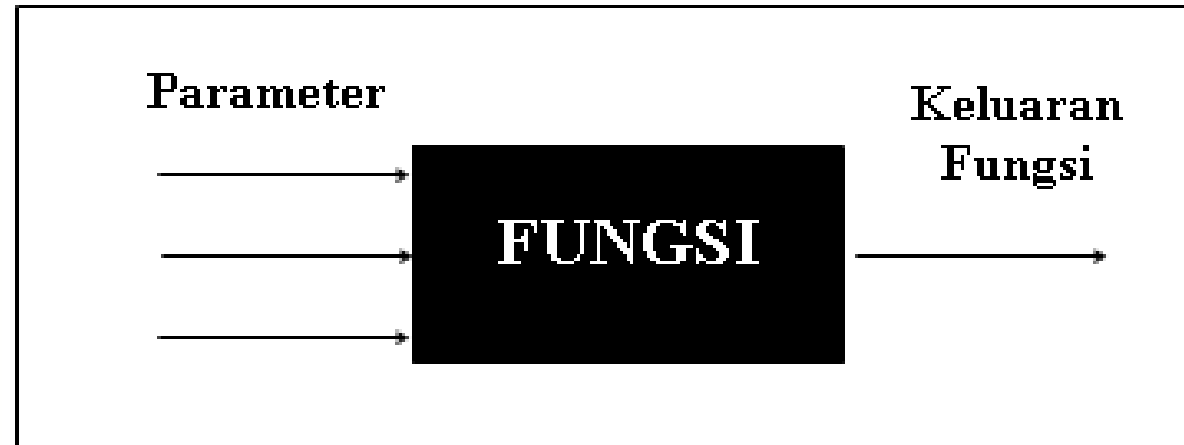
- Fungsi banyak digunakan dengan tujuan :
  - **Membuat program menjadi terstruktur**, sehingga mudah dipahami dan mudah dikembangkan. Dengan memisahkan langkah-langkah detail ke satu atau lebih fungsi-fungsi, maka fungsi utama (*main()*) menjadi lebih pendek, jelas dan mudah dimengerti.
  - **Mengurangi pengulangan (duplikasi) kode**, langkah-langkah program yang sama dan dipakai berulang-ulang di program dapat dituliskan sekali saja secara terpisah dalam bentuk fungsi-fungsi. Selanjutnya bagian program yang membutuhkan langkah-langkah ini cukup memanggil fungsi-fungsi tersebut.

# Dasar Fungsi

- Fungsi standar C yang mengemban tugas khusus contohnya adalah
  - *printf()* , yaitu untuk menampilkan informasi atau data ke layar.
  - *scanf()* , yaitu untuk membaca kode tombol yang diinputkan.
- Pada umumnya fungsi memerlukan nilai masukan atau parameter yang disebut sebagai argumen yang akan diolah oleh fungsi → parameter = **bahan baku**
- Hasil akhir fungsi berupa sebuah nilai (disebut sebagai *return value* atau nilai keluaran fungsi) → return value = **oleh-oleh**
- Oleh karena itu fungsi sering digambarkan sebagai "kotak gelap" seperti ditunjukkan pada gambarberikut ini.



# Dasar Fungsi



Gambar 5.1 Fungsi sebagai sebuah kotak gelap

- Parameter bisa diartikan sebagai “bahan baku” yang akan diproses dalam fungsi dan dikirim dari tempat fungsi tsb dipanggil
- Keluaran fungsi (*return value*) bisa diartikan sebagai “oleh-oleh” yang akan dibawa ketika proses kembali ke tempat asal fungsi tsb dipanggil

# Dasar Fungsi

- Bentuk umum dari definisi sebuah fungsi adalah sbb :

```
tipe-keluaran-fungsi  nama-fungsi  (deklarasi argumen)
{
    tubuh fungsi;
}
```

## Keterangan :

- **tipe-keluaran-fungsi**, dapat berupa salah satu tipe data C, misalnya *char* atau *int* . Kalau penentu tipe tidak disebutkan maka dianggap bertipe *int* (secara *default*).
- **tubuh fungsi** berisi deklarasi variabel (kalau ada) dan statemen-statemen yang akan melakukan tugas yang akan diberikan kepada fungsi yang bersangkutan. Tubuh fungsi ini ditulis di dalam tanda kurung kurawal buka dan kurung kurawal tutup.

# Jenis Fungsi

Berdasarkan keberadaan return value-nya, maka fungsi dibagi menjadi 2 jenis, yaitu :

## 1. punya return value (RV), ciri-cirinya :

- ada nama tipe data di depan nama fungsi, kecuali jika RVnya integer, boleh tidak ditulis karena merupakan tipe default
- ada statemen return di dalam body fungsi

## 2. tidak punya return value, ciri-cirinya :

- ada tipe void di depan nama fungsi
- tidak ada statemen return di dalam body fungsi



# Fungsi dengan Return Value integer

- Fungsi yang memiliki RV integer, maka di depan nama fungsi boleh dituliskan tipe `int` atau tanpa tipe sama sekali.

```
int minimum(int x, int y)
{
    if (x < y)
        return(x);
    else
        return(y);
}
```

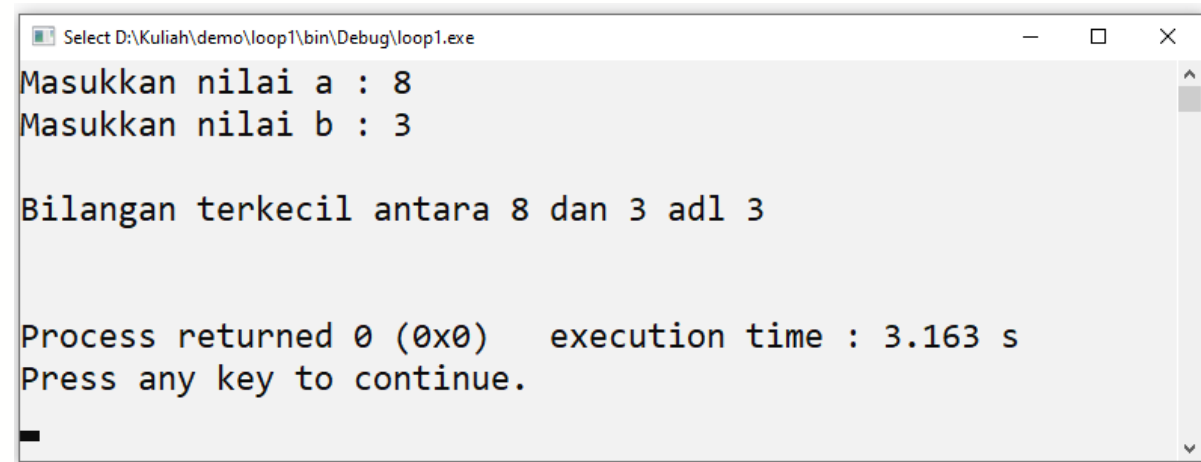
```
minimum(int x, int y)
{
    if (x < y)
        return(x);
    else
        return(y);
}
```

# Fungsi dengan Return Value integer

```
#include <stdio.h>
int minimum (int, int);
main() {
    int a, b, kecil;

    printf("Masukkan nilai a : ");
    scanf("%d", &a);
    printf("Masukkan nilai b : ");
    scanf("%d", &b);
    kecil = minimum(a, b);
    printf("\nBilangan terkecil antara %d dan %d adl %d\n\n", a, b, kecil);
}

minimum(int x, int y) {
    if (x < y)
        return(x);
    else
        return(y);
}
```



```
Select D:\Kuliah\demo\loop1\bin\Debug\loop1.exe
Masukkan nilai a : 8
Masukkan nilai b : 3

Bilangan terkecil antara 8 dan 3 adl 3

Process returned 0 (0x0)   execution time : 3.163 s
Press any key to continue.
```



# Fungsi dengan Return Value bukan integer

- Untuk fungsi yang mempunyai RV bertipe bukan integer, maka fungsi HARUS didefinisikan dengan diawali tipe RV-nya (ditulis di depan nama fungsi).
- Contoh fungsi `jumlah()` memiliki RV bertipe float sbb :

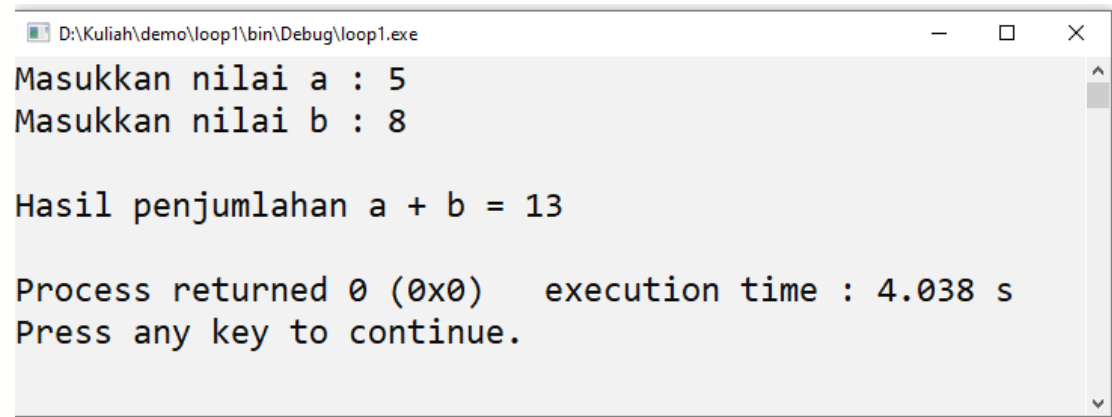
```
float jumlah(float x, float y)
{
    return(x + y);
}
```

# Fungsi dengan Return Value bukan integer

```
#include <stdio.h>
float jumlah(float, float);
main() {
    float a, b, c;

    printf("Masukkan nilai a : ");
    scanf("%f", &a);
    printf("Masukkan nilai b : ");
    scanf("%f", &b);
    c = jumlah(a, b);
    printf("\nHasil penjumlahan a + b = %g\n", c);
    return 0;
}

float jumlah(float x, float y) {
    return(x + y);
}
```



```
D:\Kuliah\demo\loop1\bin\Debug\loop1.exe
Masukkan nilai a : 5
Masukkan nilai b : 8

Hasil penjumlahan a + b = 13

Process returned 0 (0x0)   execution time : 4.038 s
Press any key to continue.
```

# Fungsi Tanpa Return Value

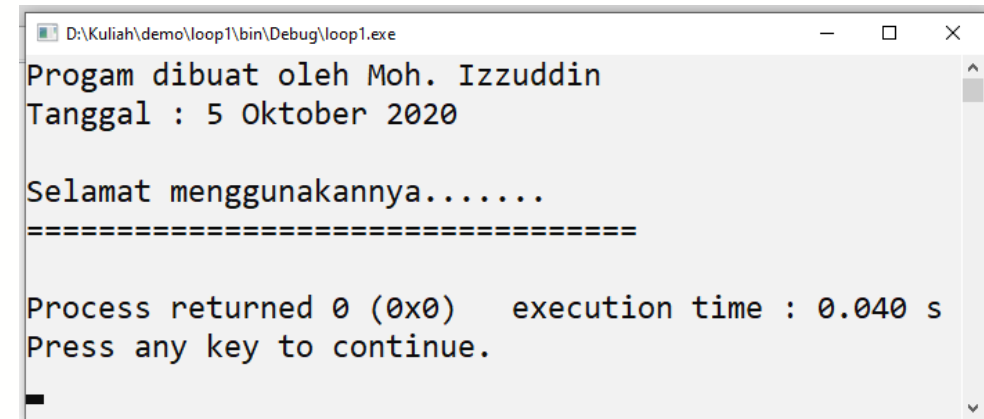
- Khusus untuk fungsi yang dirancang tanpa memberikan RV (melainkan hanya menjalankan suatu tugas khusus) biasa didefinisikan dengan diawali kata kunci *void* (di depan nama fungsi).
- Contoh fungsi `info_program()` yang tugasnya menampilkan informasi tentang sebuah program

```
void info_program()
{
    puts("=====");
    puts("Progam dibuat oleh Moh. Izzuddin ");
    puts("Tanggal : 5 Maret 2007          ");
    puts("                                ");
    puts("Selamat menggunakannya.....    ");
    puts("=====");
}
```

# Fungsi Tanpa Return Value

```
#include <stdio.h>
void info_program();
main() {
    info_program();
}

void info_program() {
    puts("=====");
    puts("Progam dibuat oleh Moh. Izzuddin ");
    puts("Tanggal : 5 Oktober 2020 ");
    puts(" ");
    puts("Selamat menggunakannya..... ");
    puts("=====");
    return 0;
}
```



```
D:\Kuliah\demo\loop1\bin\Debug\loop1.exe
Progam dibuat oleh Moh. Izzuddin
Tanggal : 5 Oktober 2020

Selamat menggunakannya.....
=====
Process returned 0 (0x0)   execution time : 0.040 s
Press any key to continue.
```

# Prototype/Deklarasi Fungsi

- Selain nama, prototipe fungsi digunakan untuk menjelaskan kepada kompiler mengenai :
  - tipe keluaran fungsi
  - jumlah parameter
  - tipe dari masing-masing parameter
- Bagi kompiler, informasi tsb akan dipakai untuk memeriksa keabsahan (validitas) parameter dalam pemanggilan fungsi.
- Salah satu keuntungannya adalah, kompiler akan melakukan konversi seandainya antara tipe parameter dalam fungsi dan parameter saat pemanggilan fungsi tidak sama, atau akan menunjukan kesalahan bila jumlah parameter dalam definisi dan saat pemanggilan berbeda.



# Prototype/Deklarasi Fungsi

Contoh prototipe fungsi;

```
float jumlah (float x, float y);
```

atau

```
float jumlah (float, float);
```

Penjelasannya adalah sbb :

Diagram explaining the components of the function prototype `float jumlah (float, float);`:

- `float`: Tipe keluaran fungsi (Function output type)
- `jumlah`: Nama fungsi (Function name)
- `(float, float)`: Tipe parameter pertama (First parameter type) and Tipe parameter kedua (Second parameter type)
- `;`: Diakhiri dengan titik koma (Ends with a semicolon)



# Prototype/Deklarasi Fungsi

Untuk fungsi yang tidak memiliki argumen (contoh program `void.c`), maka deklarasinya adalah

```
void info_program(void);
```



menyatakan bahwa `info_program()`  
tidak memiliki parameter

## Catatan :

- Untuk fungsi-fungsi pustaka, prototipe dari fungsi-fungsi berada di file-file judulnya (*header file*). Misalnya fungsi pustaka `printf()` dan `scanf()` prototipenya berada pada file dengan nama `stdio.h`
- Untuk fungsi pustaka pencantuman pada prototipe fungsi dapat dilakukan dengan menggunakan *preprocessor directive* `#include`.



# Latihan

1. a. Buatlah sebuah fungsi yang berfungsi untuk menampilkan sebuah string (di layar) = “Pilihan Menu” (misalkan nama fungsinya = **menu**). Fungsi tersebut tidak memiliki nilai kembalian (*return value*) dan juga tidak menerima parameter masukan apapun.  
b. Tulislah prototipe fungsi untuk fungsi tersebut.  
c. Buat function main untuk memanggil function `menu()` secara berulang-ulang, dengan jumlah perulangan yang merupakan input dari user.
  
2. a. Buatlah sebuah fungsi untuk menghitung jumlah triangular n (misal nama fungsinya = **triangular**). Fungsi tersebut memiliki sebuah parameter berupa bilangan int (n) yang akan dicari triangularnya serta tidak memiliki nilai kembalian (*return value*)  
b. Tulislah prototipe fungsi untuk fungsi tersebut.  
c. Buat function main untuk memanggil function `triangular()` tersebut dengan nilai n yang merupakan input dari user.



# Latihan

3. a. Buatlah sebuah fungsi untuk menghitung nilai bilangan kuadrat (misal nama fungsinya = **kuadrat**). Fungsi tersebut memiliki sebuah parameter bertipe float, yaitu bilangan yang akan dikuadratkan serta memiliki sebuah *return value* bertipe float, yaitu hasil kuadratnya  
b. Tulislah prototipe fungsi untuk fungsi tersebut.  
c. Buat fungsi `main()` untuk memanggil function `kuadrat()` tersebut dengan bilangan x yang akan dicari kuadratnya merupakan input dari user.
4. a. Definisikan sebuah fungsi `ganjil()` yang memiliki sebuah parameter bilangan bulat dan mengembalikan nilai 1 jika parameter yang diberikan adalah bilangan ganjil dan mengembalikan nilai 0 jika parameter tsb bukan bilangan ganjil  
b. Tulislah prototipe fungsi untuk fungsi tersebut.  
c. Buat fungsi `main()` untuk memanggil function `ganjil()` yang menerima input sebuah bilangan bulat yang akan ditentukan ganjil/genapnya. Tampilkan pesannya (ganjil/genap) dalam `main()`.
5. Buatlah program untuk menghitung faktorial dengan menggunakan 2 fungsi (`main()` dan `faktorial()`). Fungsi `faktorial()` memberikan return value bertipe `long int` yang akan dicetak ke layar dalam fungsi `main()`.



# Latihan

6. a. Definisikan sebuah fungsi `radian()` yang berfungsi untuk mengkonversi besaran sudut dari derajat ke radian dengan rumus  $rad = drjt / 180.0f * PI$ . Fungsi tersebut memiliki sebuah parameter yaitu derajat yang akan dikonversi, dan memiliki sebuah *return value* berupa hasil konversi dalam radian.
  - b. Tulislah prototipe fungsi untuk fungsi tersebut.
  - c. Buat `main()` untuk memanggil `radian()`, setelah sebelumnya meminta masukan nilai derajat yang akan dikonversi.
  - d. Definisikan `PI` sebagai sebuah konstanta yang bernilai `3.14159f`
7. a. Definisikan sebuah fungsi `float konversi(suhu, asal, tuju)` untuk mengkonversikan suhu dari Celsius (C) ke Fahrenheit (F), C ke Reamur (R), F ke C, F ke R, R ke C, dan R ke F. Di mana `suhu` adalah suhu sumber, `asal` adalah satuan awal suhu yang akan dikonversi dan `tuju` adalah satuan hasil konversi
  - b. Tulislah prototipe fungsi untuk fungsi tersebut.
  - c. Buat `main()` untuk fungsi `konversi()`, setelah sebelumnya meminta masukan nilai suhu, satuan asal dan satuan tujuannya.

Contoh tampilan:

```
Masukkan suhu sumber = 100
Masukkan satuan asal = C
Masukkan satuan tujuan = R
Hasil konversi suhu 100 C = 80 R
```



# Latihan

8. Dengan fungsi, buat program menghitung pangkat  $n$  dari sebuah bilangan. Input adalah bilangan itu sendiri ( $m$ ) dan pangkatnya ( $n$ ), sedangkan sebagai output adalah pangkat  $n$  dari bilangan  $m$  ( $m^n$ ).
9. Dengan menggunakan fungsi, buatlah program menghitung nilai akhir perkuliahan pada suatu matakuliah, dengan ketentuan sebagai berikut:

Nilai Absensi \* 10 %

Nilai Tugas \* 20 %

Nilai U.T.S \* 30 %

Nilai U.A.S \* 40 %

*Tampilan yang diinginkan:*

## **Program Hitung Nilai Akhir Mata Kuliah**

Masukkan Nilai Absensi : .....<di-input>

Masukkan Nilai Tugas : .....<di-input>

Masukkan Nilai U.T.S : .....<di-input>

Masukkan Nilai U.A.S : .....<di-input>

Nilai akhir yang diperoleh sebesar = <output>

# Referensi

1. Brian W. Kernighan, Dennis M. Ritchie (2012): The C Programming Language : Ansi C Version 2 Edition, PHI Learning
2. Byron Gottfried (2010) : Programming with C, Tata McGraw - Hill Education
3. [Kochan Stephen](#) (2004) : Programming in C, 3rd Edition, Sams
4. K. N. King (2008) : C Programming: A Modern Approach, 2nd Edition, W. W. Norton & Company
5. Abdul Kadir (2012) : Algoritma & Pemrograman Menggunakan C & C++, Andi Publisher, Yogyakarta
6. <http://www.gdsw.at/languages/c/programming-bbrowne/>
7. <https://www.petanikode.com/tutorial/c/>
8. <http://www.cprogramming.com/tutorial/c-tutorial.html>



**bridge to the future**

<http://www.eepis-its.edu>