



Dasar Pemrograman

Program Studi Informatika

Sesi 13 – Fungsi (Part 1)

Syahid Abdullah, S.Si, M.Kom



Outline Perkuliahan

- Struktur Fungsi
- Prototipe Fungsi
- Parameter Fungsi
- Pernyataan return()
- Pengiriman Data ke Fungsi



Fungsi

- Fungsi (Function) merupakan blok dari kode yang dirancang untuk melaksanakan tugas khusus
- Kegunaan dari fungsi adalah:
 - Mengurangi pengulangan penulisan program yang berulang atau sama.
 - Program menjadi lebih terstruktur, sehingga mudah dipahami dan dapat lebih dikembangkan



Fungsi yang umum dikenal

- **main()** -> fungsi yang bersifat mutlak, menandakan program dimulai
- **printf()** -> fungsi untuk mencetak output ke layar
- **countf()** -> fungsi untuk menghitung jumlah
- Dan fungsi lainnya



Struktur Fungsi

```
nama_fungsi(argumen)
```

```
{
```

```
... pernyataan / perintah;
```

```
... pernyataan / perintah;
```

```
... pernyataan / perintah;
```

```
}
```

Nama fungsi, boleh dituliskan secara bebas dengan ketentuan, tidak menggunakan spasi dan nama-nama fungsi yang mempunyai arti sendiri.

Argumen boleh diisi dengan suatu data atau dibiarkan kosong

Pernyataan / perintah, diletakan diantara tanda kurung '{ }



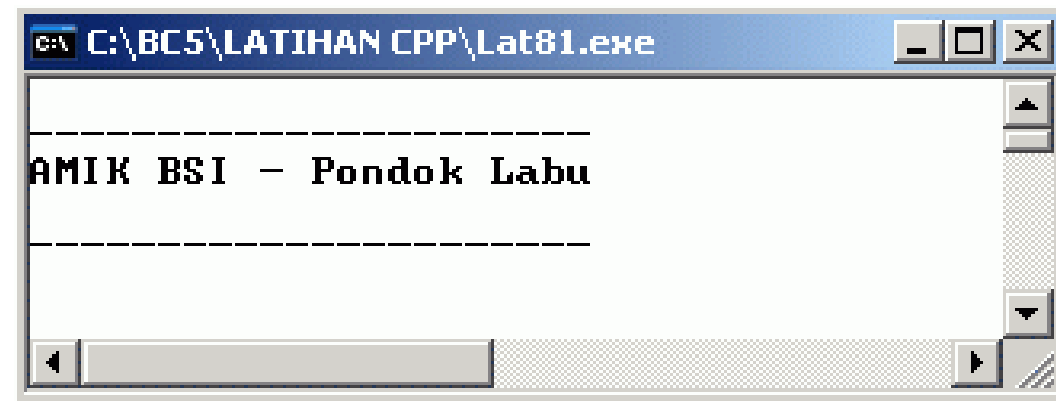
Contoh Fungsi Sederhana dengan C++

```
/* pembuatan fungsi garis() */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>

garis()
{
    printf("\n-----\n");
}

/* program utama */
main()
{
    clrscr();
    garis(); //memanggil fungsi garis
    cout<<"AMIK BSI - Pondok Labu"<<endl;;
    garis(); //memanggil fungsi garis
    getch();
}
```

Output yang dihasilkan:





Prototipe Fungsi

Prototipe fungsi digunakan untuk mendeklarasikan ke kompiler mengenai:

- Tipe data keluaran dari fungsi.
- Jumlah parameter yang digunakan
- Tipe data dari masing-masing parameter yang digunakan.



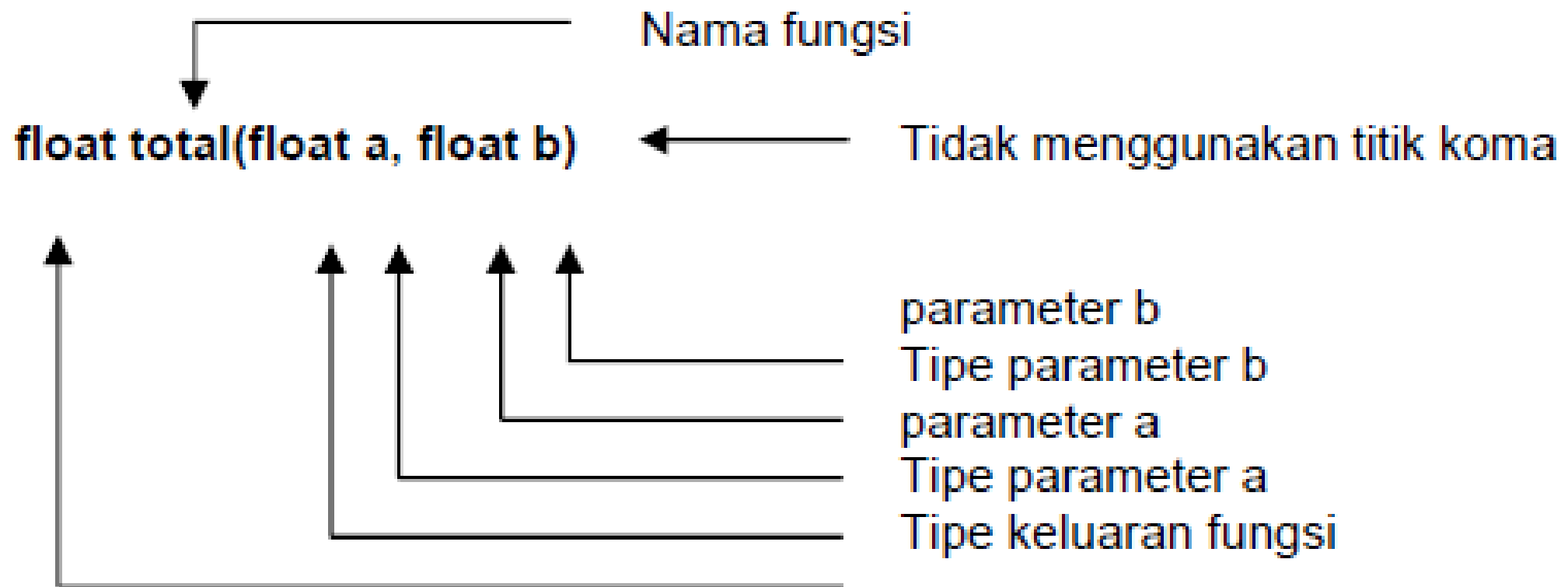
Prototipe Fungsi

Keuntungan didalam pemakai prototipe yaitu :

- Kompiler akan melakukan konversi antara tipe parameter dalam definisi dan parameter fungsi.
- Jika jumlah parameter yang digunakan dalam definisi fungsi dan pada saat pemanggilan fungsi berbeda atau tidak sama, maka akan menunjukkan kesalahan



Contoh Prototipe Fungsi





Parameter Fungsi

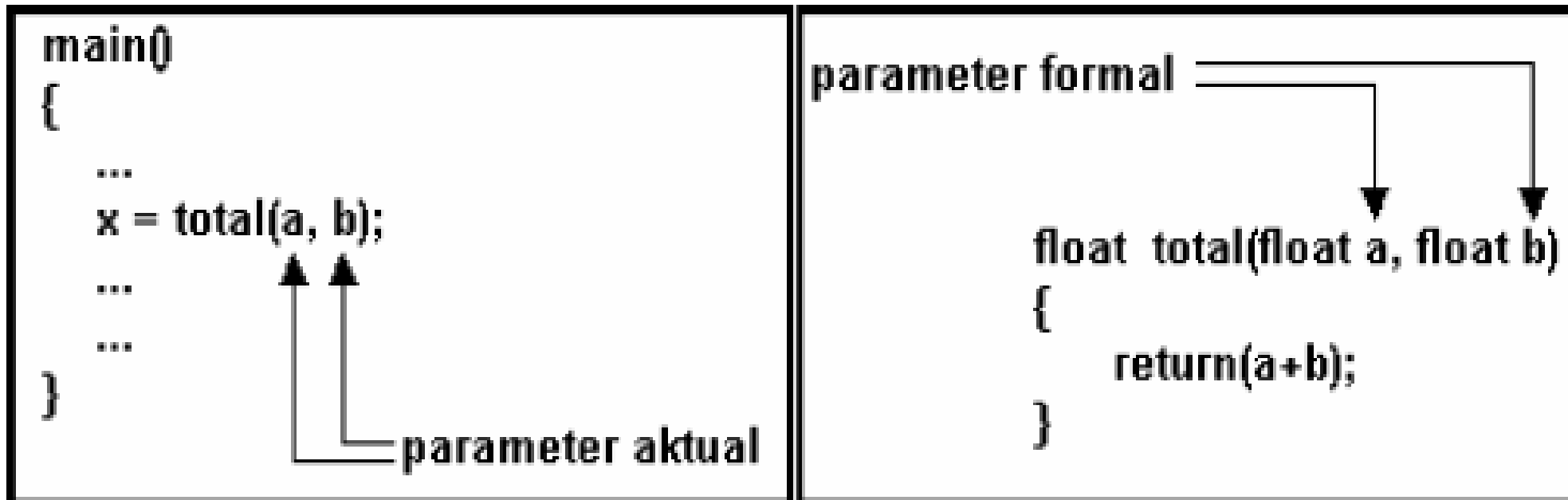
Terdapat dua macam parameter fungsi, yaitu :

- Parameter formal adalah variabel yang terdapat pada daftar parameter yang berada didalam definisi fungsi.
- Parameter Aktual adalah variabel yang digunakan pada pemanggilan suatu fungsi.



Parameter Fungsi

- Bentuk penulisan Parameter Formal dan Parameter Aktual





Parameter Fungsi

- Dua cara melewatkan parameter ke dalam fungsi:
 1. Pemanggilan dengan nilai (Call by Value)
 2. Pemanggilan dengan Referensi (Call by Reference)



Parameter Fungsi

Pemanggilan dengan nilai (Call by Value)

- Pada pemanggilan dengan nilai yaitu nilai dari parameter aktual akan dimasukkan keparameter formal. Dengan cara ini nilai parameter aktual tidak bisa berubah, walaupun nilai dari parameter formal berubah.



Contoh Call By Value

```
/* Penggunaan Call By Value */
/* Program Tambah Nilai */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
tambah(int m, int n);
main()
{
    int a, b;
    a = 5;
    b = 9;
    clrscr();
    cout<<"Nilai Sebelum Fungsi Digunakan ";
    cout<<"\na = "<<a<<" b = "<<b;
    tambah(a,b);
```

```
    cout<<"\nNilai Setelah Fungsi Digunakan";
    cout<<"\na = "<<a<<" b = "<<b;
    getch();
}
tambah(int m, int n)
{
    m+=5;
    n+=7;
    cout<<"\n\nNilai di dalam Fungsi Tambah()";
    cout<<"\nm = "<<m<<" n = "<<n;
    cout<<endl;
}
```

Output yang dihasilkan:

```
C:\BC5\LATIHAN CPP\Lat82.exe
Nilai Sebelum Fungsi Digunakan
a = 5 b = 9
Nilai di dalam Fungsi Tambah()
m = 10 n = 16
Nilai Setelah Fungsi Digunakan
a = 5 b = 9_
```



Parameter Fungsi

Pemanggilan dengan referensi (Call by Reference)

- Pemanggilan dengan reference merupakan pemanggilan alamat suatu variabel didalam fungsi. Cara ini dapat dipakai untuk mengubah isi suatu variabel yang diluar dari fungsi dengan melaksanakan pengubahan nilai dari suatu variabel dilakukan didalam fungsi



Contoh Call By Reference

```
/* Penggunaan Call By Reference */
/* Program Tambah Nilai */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
tambah(int *c, int *d);
main()
{
    int a, b;
    a = 4;
    b = 6;
    clrscr();
    cout<<"Nilai    Sebelum    Pemanggilan
    Fungsi";
```

```
    cout<<"\na = "<<a<<" b = "<<b;
    tambah(&a,&b);
    cout<<endl;
    cout<<"\nNilai Setelah Pemanggilan Fungsi";
    cout<<"\na = "<<a<<" b = "<<b;
    getch();
}

tambah(int *c, int *d)
{
    *c+=7;
    *d+=5;
    cout<<endl;
    cout<<"\nNilai di Akhir Fungsi Tambah()";
    cout<<"\nc = "<<*c<<" d = "<<*d;
}
```

Output yang dihasilkan:

```
C:\BC5\LATIHAN CPP\Lat83.exe
Nilai Sebelum Pemanggilan Fungsi
a = 3 b = 7
Nilai di Akhir Fungsi Tambah()
c = 10 d = 12
Nilai Setelah Pemanggilan Fungsi
a = 10 b = 12
```




Pernyataan return()

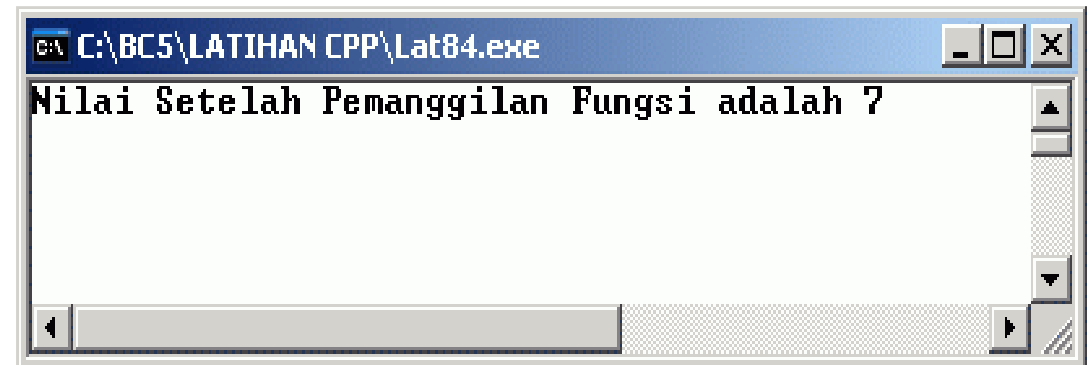
- Digunakan untuk mengirimkan nilai atau nilai dari suatu fungsi kepada fungsi yang lain yang memanggilnya. Pernyataan return() diikuti oleh argumen yang berupa nilai yang akan dikirimkan.



Contoh pernyataan return()

```
/* Penggunaan Fungsi return() */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
tambah(int *c); //prototype fungsi tambah
main()
{
    int a, b = 5;
    clrscr();
    a = tambah(&b);
    cout<<"Nilai Setelah Pemanggilan Fungsi adalah "<<a;
    getch();
}
tambah(int *c) //fungsi tambah
{
    return(*c+=2);
}
```

Output yang dihasilkan:





Pengiriman Data ke Fungsi

- Mengirimkan suatu nilai data konstanta ke suatu fungsi yang lain dapat dilakukan dengan cara yang mudah

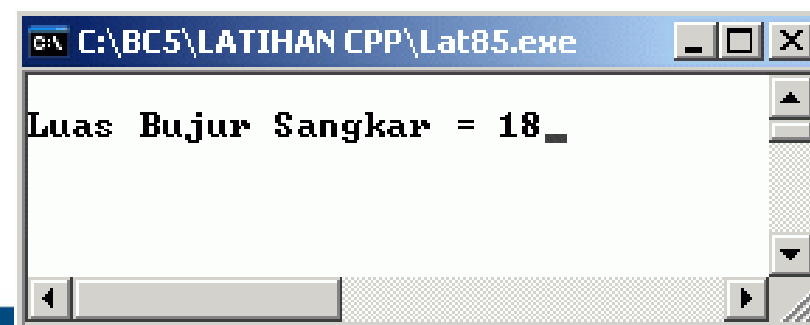


Contoh pengiriman data ke fungsi

```
/* Pengiriman data Konstanta */
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
luas(float sisi);
main()
{
float luas_bs;
clrscr();
luas_bs = luas(4.25);
cout<<"\nLuas Bujur Sangkar = "<<luas_bs;
getch();
}
luas(float sisi)
{
return(sisi*sisi);
}
```

Pernyataan `luas_bs=luas(4.25)`, akan dikirimkan nilai kepada fungsi `luas()`, untuk diolah lebih lanjut, yang nilai tersebut akan ditampung pada variabel `sisi`. Selanjutnya didalam fungsi `return` terjadi perkalian `sisi` dengan `sisi`, setelah itu hasil perkalian tersebut dikirim balik ke variabel `luas_bs` yang memanggil fungsi

Output yang dihasilkan:





Pengiriman Data Variabel ke Fungsi

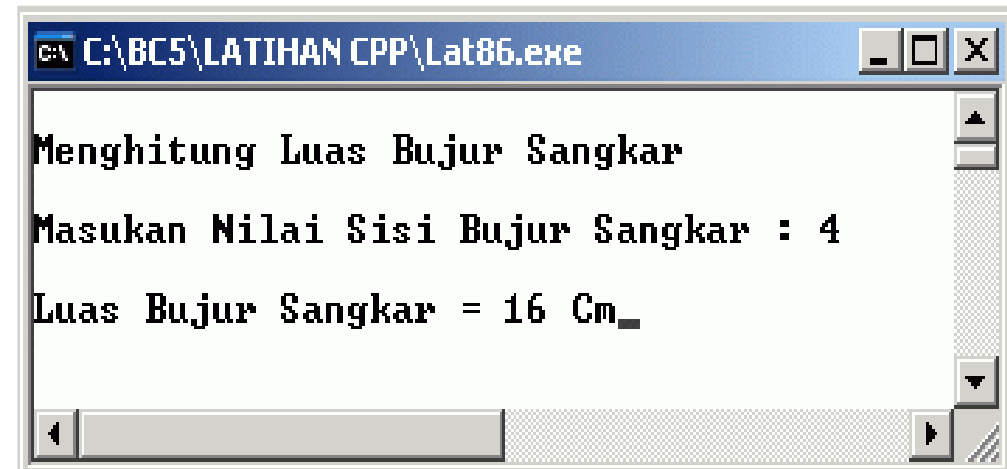
- Bentuk pengiriman data Variabel, sama seperti halnya pengiriman suatu nilai data konstanta ke suatu fungsi, hanya saja nilai yang dikirimkan tersebut senantiasa dapat berubah-ubah.



Contoh pengiriman data variabel ke fungsi

```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
luas(float sisi);
main()
{
float luas_bs, sisi_bs;
clrscr();
cout<<"\nMenghitung Luas Bujur Sangkar"<<endl;
cout<<"\nMasukan Nilai Sisi Bujur Sangkar : ";
cin>>sisi_bs;
luas_bs = luas(sisi_bs);
cout<<"\nLuas Bujur Sangkar = "<<luas_bs<<" Cm";
getch();
}
luas(float sisi)
{
return(sisi*sisi);
}
```

Output yang dihasilkan:





Daftar Pustaka

Frieyadie, *Pemrograman C++ dengan Borland C++ 5.02 (Edisi Revisi)*. DIKTAT KULIAH PEMROGRAMAN KOMPUTER BINA SARANA INFORMATIKA, 2007.

Goodrich, Michael, Roberto Tamassia, and David Mount. *Data structures and algorithms in C++*. John Wiley & Sons, 2011.

Mehlhorn, Kurt, and Peter Sanders. *Algorithms and data structures: The basic toolbox*. Springer, 2010.



Terima Kasih