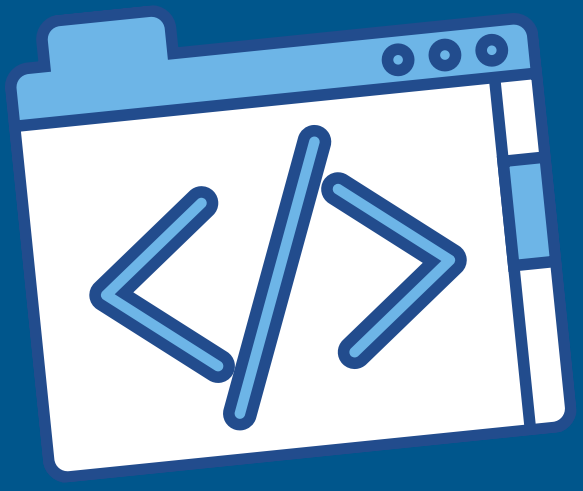
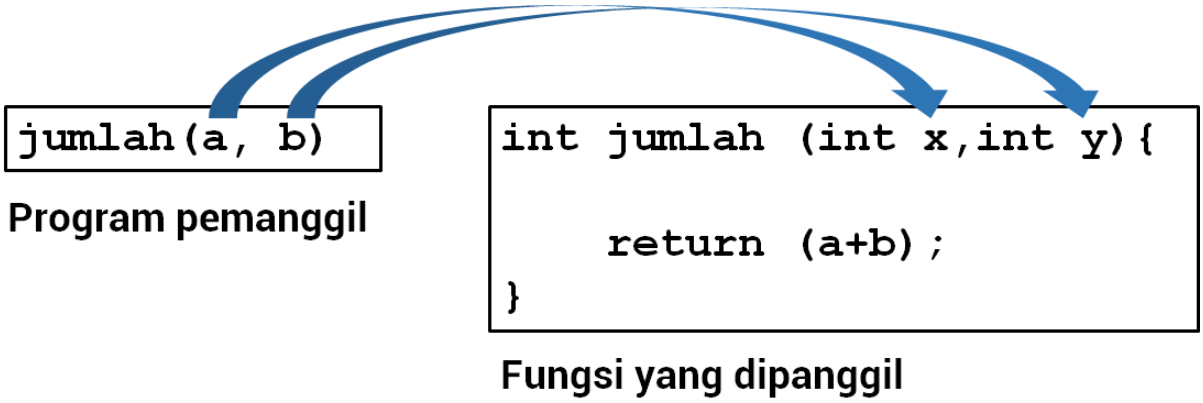
PENGIRIMAN PARAMETER



PENGIRIMAN PARAMETER

Sebuah fungsi dapat dipanggil atau digunakan oleh fungsi atau bagian program yang lain. Komunikasi yang terjadi antara satu fungsi yang memanggil dengan fungsi yang dipanggil dilakukan dengan cara saling bertukar data atau informasi.

Pertukaran atau pengirimanan data tersebut melalui antar muka atau penghubung yang disebut **parameter.** Parameter dapat berupa *identifier* (variabel atau konstanta) dapat pula berupa nilai.



Pada saat pemanggilan terjadi pertukaran data, parameter aktual pada program pemanggil mengirimkan data atau informasi pada parameter formal dalam fungsi. Pada contoh parameter aktual a dan b mengiriman nilai pada parameter formal x dan y.

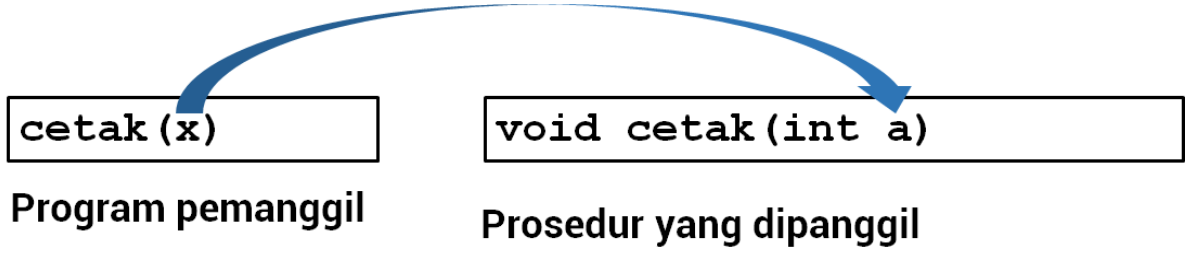
Terdapat dua buah cara untuk melewatkan parameter ke dalam sebuah fungsi maupun prosedur, yaitu pemanggilan dengan **nilai *(pass by value)*** dan berdasarkan a**lamatnya *(pass by reference)***.

CALL BY VALUE

Perhatikan contoh program berikut!

```
void cetak(int a){
   a = 0;
   cout << "Nilai pada prosedur
   cetak = "<< a << endl;
}

int main(){
   int x = 10;

   cout << "Nilai sebelum
   pemanggilan prosedur = " << x
   << endl;

   cetak(x);

   cout << "Nilai setelah
   pemanggilan prosedur = " << x
   << endl;
}
```

Berdasarkan contoh di atas, terdapat sebuah prosedur bernama cetak. Variabel a adalah parameter formal dalam prosedur, sedangkan variabel x yang bernilai 10 adalah **nilai** yang dikirimkan oleh program pemanggil, nilai variabel x diberikan kepada variabel a.



Program pemanggil **Prosedur yang dipanggil**

Ketika program tersebut dieksekusi maka nilai keluaran program adalah:

```
Hasil eksekusi:
Nilai sebelum pemanggilan prosedur
x = 10

Nilai pada prosedur cetak()
a = 0

Nilai setelah pemanggilan prosedur
x = 10
```

Nilai variabel x sebelum dan sesudah pemanggilan fungsi cetak **bernilai tetap** walaupun nilai a pada prosedur cetak berubah. Sebab nilai a hanyalah **salinan nilai** dari variabel x.

Sehingga, meskipun nilai variabel a pada prosedur cetak berubah, nilai variabel x pada program utama tidak berubah. Proses pengiriman parameter pada contoh tersebut merupakan proses pengiriman parameter dengan nilai ***(call by value).***

CALL BY VALUE

Berdasarkan contoh tersebut dapat disimpulkan bahwa pemanggilan dengan nilai merupakan proses **penyalinan nilai** dari parameter aktual ke parameter formal. Fungsi atau prosedur yang menerima kiriman nilai akan menyimpannya di **alamat terpisah** dari nilai asli yang digunakan oleh program yang memanggil fungsi.

Dengan cara ini, perubahan nilai di fungsi (parameter formal) **tidak akan merubah nilai asli** di bagian program yang memanggilnya.

CALL BY REFERENCE

Pada pemanggilan parameter secara acuan ***(call by reference)*** pengirimanan data dilakukan dengan cara **mengirimkan alamat** dari suatu variabel ke dalam fungsi yang dipanggil, jadi pada ***call by refererence*** yang dikirimkan adalah a**lamat dari data bukan nilai datanya.**

Fungsi yang menerima kiriman alamat akan menggunakan **alamat yang sama** untuk mendapatkan nilai datanya. Perubahan nilai pada fungsi akan merubah nilai asli di bagian program yang memanggil fungsi. Sebagai contoh, perhatikan program berikut:

```
void cetak (int *a) {
   a = 0;
   cout << "Nilai pada prosedur
   cetak = "<< a << endl;
}

int main(){
   int x = 10;

   cout << "Nilai sebelum
   pemanggilan prosedur = " << x
   << endl;

   cetak(&x);

   cout << "Nilai setelah
   pemanggilan prosedur = " << x
   << endl;
}
```

Pada saat pemanggilan penulisan parameter aktual dituliskan dengan opeartor **"&"** yang berarti **menunjukan alamat**. Sedangkan pada parameter formal ditandai dengan tanda arsterik "*" yang menyatakan bahwa variabel tersebut adalah suatu **variabel pointer** (variabel yang berisi alamat dari variabel lain) .

Pada contoh di atas yang menjadi parameter aktual adalah variabel x sedangkan yang menjadi parameter formal adalah variabel a. Saat fungsi dipanggil, alamat dari parameter aktual x dikirimkan kepada parameter formal a. Ketika program tersebut dieksekusi maka nilai keluaran program adalah:

```
Hasil eksekusi:
Nilai sebelum pemanggilan prosedur
x = 10

Nilai pada prosedur cetak()
a = 0

Nilai setelah pemanggilan prosedur
x = 0
```

Nilai parameter aktual setelah prosedur cetak dipanggil berubah menjadi 0. Hal itu terjadi, karena alamat memori dari parameter aktual yang dikirimkan ke dalam prosedur. Sehingga ketika nilai dari parameter formal dalam prosedur berubah maka nilai parameter aktual pada program pemanggil **berubah**.

PERBEDAAN CALL BY VALUE & CALL BY REFERENCE

CALL BY VALUE	CALL BY REFERENCE
Mengirimkan Nilai Pemanggilan secara nilai merupakan upaya untuk melewatkan nilai dari parmaeter formal ke dalam fungsi.	**Mengirimkan Alamat** Pemanggilan secara nilai merupakan upaya untuk melewatkan alamat dari parameter formal ke dalam fungsi.
Tidak menggunakan alamat yang sama Karena yang dikirimkan adalah nilai datanya. Maka parameter formal dan parmaeter aktual tidak menggunakan alamat yang sama.	**Menggunakan alamat yang sama** Karena yang dikirimkan adalah alamat data. Maka parameter formal dan parmaeter aktual menggunakan alamat yang sama.
Tidak Terjadi Perubahan Nilai Perubahan nilai pada fungsi tidak merubah nilai asli di bagian program yang memanggil fungsi.	**Terjadi Perubahan Nilai** Perubahan nilai pada fungsi akan merubah nilai asli di bagian program yang memanggil fungsi.