Finally: ELEVEN

Function Parameter by Value and by Reference

REFRESH!

- Function yang telah kita pelajari adalah kumpulan instruksi yang digunakan untuk melaksanakan suatu maksud tertentu dan diberi nama yang unik.
- Function ada yang mengembalikan nilai (*non-void*) dan ada yang tidak mengembalikan nilai (**void**)

  Contoh:

```
int Kali(int a,int b){
     return a*b;
}
```

  Atau seperti ini:

```
void ShowError(){
     printf("Error on line %d: %s",Error.line,Error.message);
}
```

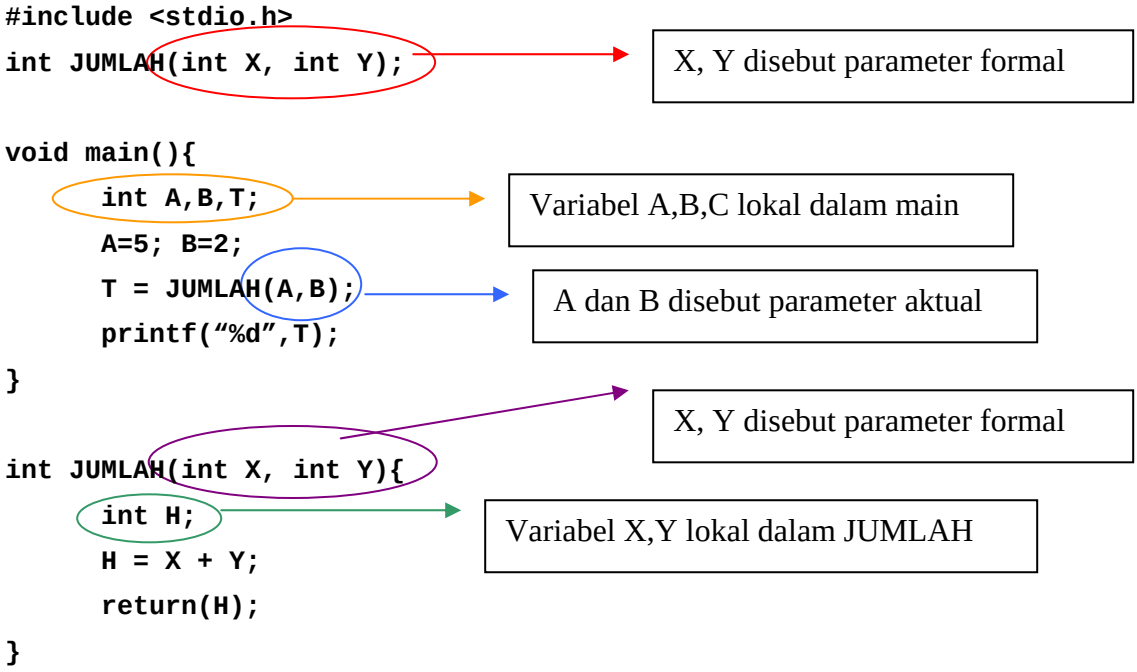
- Variabel-variabel yang ada di dalam function bersifat **lokal** untuk fungsi tersebut saja, sedangkan nilai kembalian fungsi akan dikenal pada fungsi yang memanggil fungsi tersebut.

```
#include <stdio.h>
#include <conio.h>

int d=3,e=1;

void coba_lokal(int a,int b){
      int c = 0;
      int d = 10;
      int e;
      e = (a+b) * (c+d);
      printf("lokal a = %d\n",a);
      printf("lokal b = %d\n",b);
      printf("lokal c = %d\n",c);
      printf("lokal d = %d\n",d);
      printf("lokal e = %d\n",e);
}

void main(){
      int a=2;
      int b;
```

```
        b = 4;
        int c=0;

        printf("main a = %d\n",a);
        printf("main b = %d\n",b);
        coba_lokal(a,b);
        printf("main c = %d\n",c);
        printf("global d = %d\n",d);
        printf("global e = %d\n",e);
        getch();
    }
```

- Function boleh memiliki parameter ataupun tidak memiliki parameter satupun
- Parameter dalam function terdiri dari 2 jenis:
    - **Parameter Formal**: yaitu parameter yang tertulis dalam definisi fungsi.
    - **Parameter Aktual**: yaitu parameter yang merupakan inputan langsung pada saat penggunakan fungsi tersebut. Dapat berupa variabel maupun langsung berupa value.

Contoh:



Pengiriman Parameter ke suatu Function secara nilai (By Value)

- Yang dikirimkan ke fungsi adalah **nilainya**, *bukan* **alamat memori** letak dari datanya
- Fungsi yang menerima kiriman nilai ini akan menyimpannya di **alamat terpisah** dari nilai aslinya yang digunakan oleh program yang memanggil fungsi tersebut
- Karena itulah pengubahan nilai di dalam fungsi **tidak akan berpengaruh** pada nilai asli di program yang memanggil fungsi walaupun keduanya menggunakan nama variabel yang sama

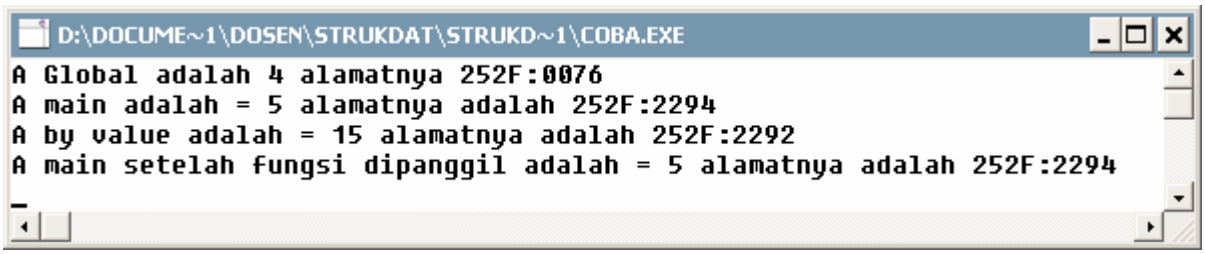
Contoh

```
#include <stdio.h>
```

```
#include <conio.h>

int a=4;

void getAGlobal(){
     printf("A Global adalah %d alamatnya %p\n",a,&a);
}

void fungsi_by_value(int a){
     a = a * 3;
     printf("A by value adalah = %d alamatnya adalah %p\n",a,&a);
}

void main(){
     int a = 5;
     getAGlobal();
     printf("A main adalah = %d alamatnya adalah %p\n",a,&a);
     fungsi_by_value(a);
     printf("A main setelah fungsi dipanggil adalah = %d
alamatnya adalah %p\n",a,&a);
     getch();
}
```

Pengiriman satu arah

Hasil:



Di dalam Memori:

a di *global* nilai **4** alamat **252F:0076**	a di *main* nilai **5** alamat **252F:2294**	a di *fungsi_by_value* nilai **15** alamat **252F:2292**	a di *main after function* nilai **5** alamat **252F:2294**

- Pengiriman by value adalah pengiriman **searah**, dari program pemanggil fungsi ke fungsi yang dipanggilnya
- Pengiriman by value dapat dilakukan untuk suatu **statement,** *tidak hanya* untuk suatu variabel, value, array atau konstanta saja.

```
...
void fungsi_by_value(int a){
     a = a * 3;
     printf("A by value adalah = %d alamatnya adalah %p\n",a,&a);
```

```
}

void main(){
     int a = 5;
     getAGlobal();
     printf("A main adalah = %d alamatnya adalah %p\n",a);

     fungsi_by_value(5*a+1);
     getch();
}
...
```

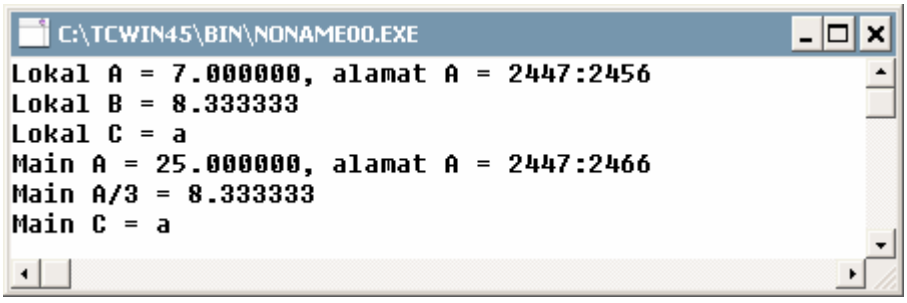
Statement

- Secara **default**, C menggunakan **parameter by value** untuk pembuatan fungsi-fungsinya.

Contoh lain:

```
#include <stdio.h>
#include <conio.h>

void Secara_Nilai(float a,float b,char c){
     float *Alamat_A;
     Alamat_A = &a;
     a = 7;
     printf("Lokal A = %f, alamat A = %p\n",a,Alamat_A);
     printf("Lokal B = %f\n",b);
     printf("Lokal C = %c\n",c);
}

void main(){
     float a=25,*Alamat_A;
     char c = 'a';
     Alamat_A = &a;
     Secara_Nilai(a,a/3,c);
     printf("Main A = %f, alamat A = %p\n",a,Alamat_A);
     printf("Main A/3 = %f\n",(a/3));
     printf("Main C = %c\n",c);
     getch();
}
```

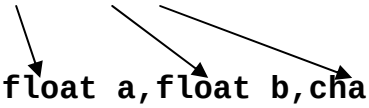
Hasil:



Dapat disimpulkan sebagai berikut:

1. Parameter aktual yang dikirimkan adalah datanya, yaitu **`Secara_Nilai(a,a/3,c)`**
2. Alamat nilai a pada `main` dan a pada fungsi `Secara_Nilai` berbeda, yaitu **2447:2456** dan **2447:2466**
3. Perubahan nilai a dalam fungsi `Secara_Nilai` menjadi **7** tidak mengubah nilai a pada main yaitu tetap **25**
4. Pengirimannya satu arah

   `Secara_Nilai(a,a/3,c)`

   `Secara_Nilai(float a,float b,char c)`

5. Pengiriman parameter dapat berupa ungkapan (statement) yaitu **a/3**

Pengiriman Parameter secara Acuan (by Reference)

1. Yang dikirimkan adalah **alamat memori** letak dari nilai datanya, *bukan* **nilai datanya**
2. Fungsi yang menerima parameter ini akan menggunakan **alamat yang sama** dengan alamat nilai datanya
3. Karena itulah pengubahan nilai di fungsi **akan mengubah** juga nilai asli di program pemanggil fungsi tersebut
4. Pengiriman parameter by reference adalah pengiriman **dua arah**, yaitu dari program pemanggil fungsi ke fungsi dan sebaliknya dari fungsi ke program pemanggilnya
5. Pengiriman parameter by reference **tidak dapat** digunakan untuk suatu ungkapan, hanya bisa untuk variabel, konstanta atau elemen array saja

Contoh1:

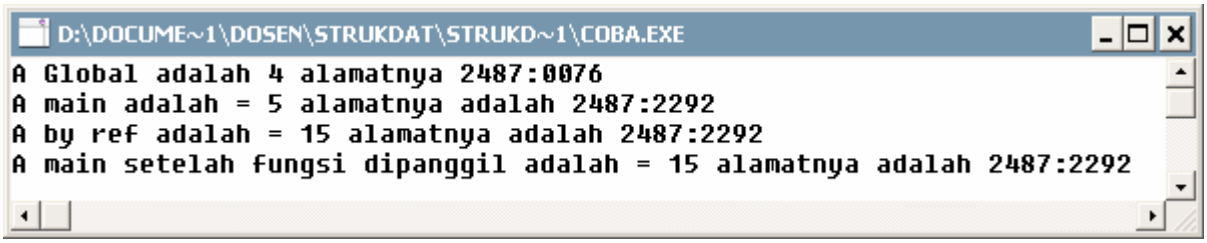
```
#include <stdio.h>
#include <conio.h>

int a=4;

void getAGlobal(){
     printf("A Global adalah %d alamatnya %p\n",a,&a);
}

void fungsi_by_ref(int *a){
     *a = *a * 3;
     printf("A by ref adalah = %d alamatnya adalah %p\n",*a,a);
}

void main(){
     int a = 5;
     getAGlobal();
     printf("A main adalah = %d alamatnya adalah %p\n",a,&a);
     fungsi_by_ref(&a);
```

Menggunakan asteris/bintang (*)

Menggunakan asteris/bintang (*)

```
     printf("A main setelah fungsi dipanggil adalah = %d alamatnya
adalah %p\n",a,&a);
     getch();
}
```

Hasil1:



Di dalam Memori:

a di *global* nilai **4** alamat **2487:0076**	a di *main* nilai **5** alamat **2487:2292**	a di *fungsi_by_value* nilai **15** alamat **2487:2292**	a after *fungsi_by_ref* nilai **15** alamat **2487:2292**

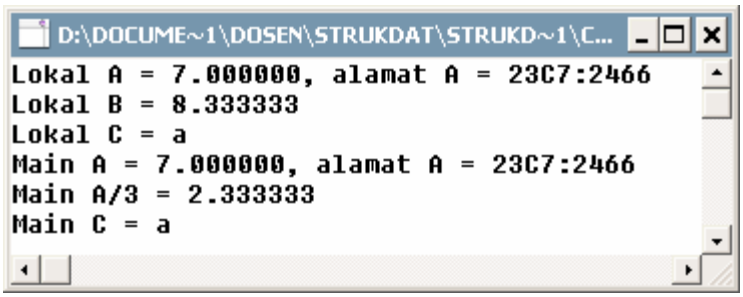
Contoh lain:

Menggunakan asteris/bintang (*)

Tanpa asteris/bintang (*)

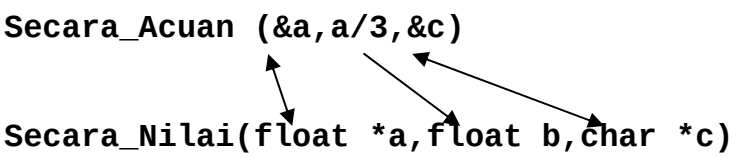
```
#include <stdio.h>
#include <conio.h>

void Secara_Acuan(float *a,float b,char *c){
     float *Alamat_A;
     Alamat_A = a;
     *a = 7;
     printf("Lokal A = %f, alamat A = %p\n",*a,Alamat_A);
     printf("Lokal B = %f\n",b);
     printf("Lokal C = %c\n",*c);
}

void main(){
     float a=25,*Alamat_A;
     char c = 'a';
     Alamat_A = &a;
     Secara_Acuan(&a,a/3,&c);
     printf("Main A = %f, alamat A = %p\n",a,Alamat_A);
     printf("Main A/3 = %f\n",(a/3));
     printf("Main C = %c\n",c);
     getch();
}
```

Hasil:



Kesimpulan:

1. Operator pada pengiriman parameter by reference adalah menggunakan **operator '&'** yang berarti mengacu pada alamat memori dari variabel tersebut.
2. Parameter aktual yang dikirimkan adalah alamat memori, yaitu **`Secara_Acuan(&a,a/3,&c)`**
3. Alamat nilai a pada main dan a pada fungsi `Secara_Acuan` sama, yaitu **23C7:2466**
4. Perubahan nilai a dalam fungsi `Secara_Acuan` menjadi **7** juga mengubah nilai a pada main yang semula **25** menjadi **7**
5. Pengirimannya dua arah



6. Pengiriman parameter tidak dapat berupa ungkapan (statement), sehingga untuk statement **a/3** harus berupa parameter by value

Pengiriman Parameter berupa Array berdimensi satu

- Pengiriman ini adalah pengiriman **by reference**, karena yang dikirimkan sebenarnya adalah **alamat dari elemen pertama** array yang digunakan sebagai parameter, bukan seluruh nilai elemen-elemennya.
- Alamat elemen pertama array dapat ditulis dengan menyebutkan **nama array tanpa menuliskan indeksnya.**

Contoh1:

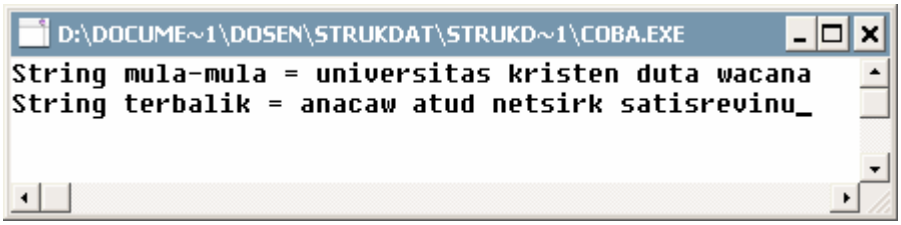
```
#include <stdio.h>
#include <conio.h>

void Cetak_Mundur(char S[]){
     int n;
     //menghitung panjang string dimasukkan ke n
     for(n=0;S[n]!='\0';n++);
     for(int i=n-1;i>=0;i--) printf("%c",S[i]);
}
```

Tanpa parameter

```
void main(){
     char S[] = "universitas kristen duta wacana";
     printf("String mula-mula = %s\n",S);
     printf("String terbalik = ");
     Cetak_Mundur(S);
     getch();
}
```

Parameter aktual

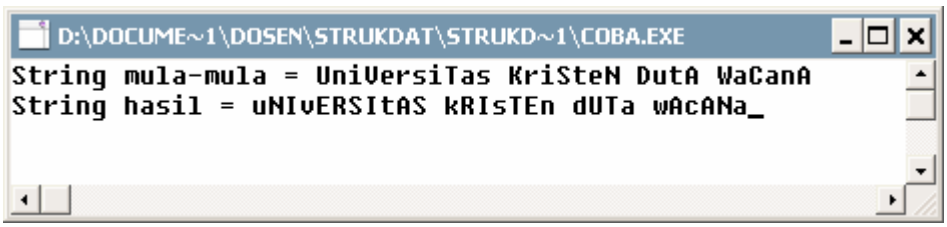
Hasil:



Contoh2:

```
#include <stdio.h>
#include <conio.h>

void Balik_Huruf(char S[]){
     int n;
     //menghitung panjang string dimasukkan ke n
     for(n=0;S[n]!='\0';n++);
     for(int i=0;i<n;i++){
          if(S[i]>='a' && S[i]<='z') S[i] = S[i] - 'a' + 'A';
     //diubah menjadi huruf besar
          else if(S[i]>='A' && S[i]<='Z') S[i] = S[i] - 'A' + 'a';
     //diubah menjadi huruf kecil
     }
}

void main(){
     char S[] = "UniVersiTas KriSteN DutA WaCanA";
     printf("String mula-mula = %s\n",S);
     printf("String hasil = ");
     Balik_Huruf(S);
     printf("%s",S);
     getch();
}
```

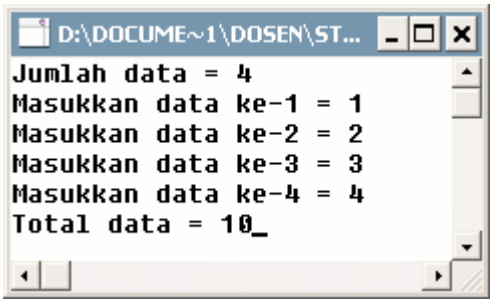
Hasil:



Contoh3:

```
#include <stdio.h>
#include <conio.h>

void Masukkan_Data(int data[],int jml){
     for(int i=0;i<jml;i++){
          printf("Masukkan data ke-%d = ",i+1);scanf("%d",&data[i]);
     }
}

int Hitung_Total(int data[],int jml){
     int t = 0;
     for(int i=0;i<jml;i++){
          t += data[i];
     }
     return t;
}

void main(){
     int n;
     int total;
     int data[10];
     printf("Jumlah data = ");scanf("%d",&n);
     Masukkan_Data(data,n);
     total = Hitung_Total(data,n);
     printf("Total data = %d",total);
     getch();
}
```

Hasil:



Pengiriman Parameter berupa Array dua dimensi

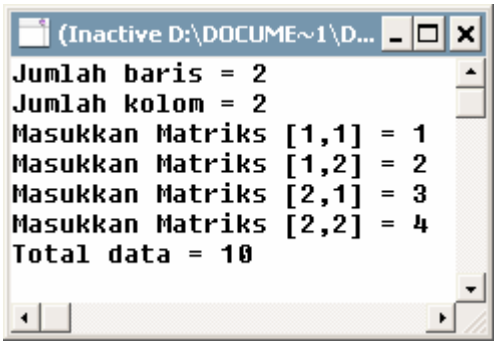
- Pengiriman parameter berupa array 2 dimensi **hampir sama** dengan pengiriman parameter array satu dimensi
- Perbedaannya: **menyebutkan baris dan kolom** array dimensi dua tersebut
- Mendeklarasikan MAX_ROWS dan MAX_COLS yang digunakan untuk pengiriman parameter array dua dimensi

- Pada saat pengiriman parameter formal array dua dimensi, kita harus menyebutkan banyaknya dimensi array untuk kolom, sehingga ukuran kolom dapat diketahui. Hal ini berkaitan dengan pemesanan variabel array di memori

Contoh:

```
#include <stdio.h>
#include <conio.h>
#define MAX_COLS 10
#define MAX_ROWS 10

void Masukkan_Data(int data[][MAX_COLS],int m,int n){
     for(int i=0;i<m;i++){
          for(int j=0;j<n;j++){
               printf("Masukkan Matriks [%d,%d] =
",i+1,j+1);scanf("%d",&data[i][j]);
          }
     }
}

int Hitung_Total(int data[][MAX_COLS],int m,int n){
     int t = 0;
     for(int i=0;i<m;i++){
          for(int j=0;j<n;j++){
               t += data[i][j];
          }
     }
     return t;
}

void main(){
     int m,n;
     int total;
     int data[MAX_ROWS][MAX_COLS];
     printf("Jumlah baris = ");scanf("%d",&m);
     printf("Jumlah kolom = ");scanf("%d",&n);
     Masukkan_Data(data,m,n);
     total = Hitung_Total(data,m,n);
     printf("Total data = %d",total);
     getch();
}
```

Hasil:



☺ THE END ☺
Good Luck for The Final Test

Thanks for being my great students!
I'm very sorry if I had mistakes to all of you!