MODUL – 6

Pokok bahasan :

Fungsi

- **Penggunaan fungsi menggunakan void**
- **Penggunaan fungsi tanpa argument**
- **Fungsi rekursif**
- **Fungsi prototype**

Fungsi.

Fungsi terbagi 2, ada yang namanya **build in function** dan fungsi yang kita deklarasikan sendiri. Build in function adalah fungsi-fungsi yang sudah ada dalam **library C** itu sendiri seperti beberapa fungsi string yang ada di string.h **(strlen, strcpy, strcmp, dll).** Aturan penggunaan build in function untuk string akan kita bahas di tutorial mendatang.

Fungsi merupakan suatu bagian dari program c++ yang di maksudkan untuk mengerjakan suatu tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Fungsi merupakan elemen utama dalam c++ karena bahasa c++ sendiri terbentuk dari kumpulan fungsi-fungsi. Dalam setiap program, setidaknya terdapat satu fungsi utama yang merupakan fungsi awal pemanggilan program yaitu fungsi **main()**.

Fungsi terbagi 2, ada yang namanya **build in function** dan fungsi yang kita deklarasikan sendiri. Build in function adalah fungsi-fungsi yang sudah ada dalam **library C** itu sendiri seperti beberapa fungsi string yang ada di string.h **(strlen, strcpy, strcmp, dll).** Aturan penggunaan build in function untuk string akan kita bahas di tutorial mendatang.

Struktur fungsi :

```
tipe_data nama_fungsi(parameter){
        //statement
```

Tipe Data

Sama halnya dengan variabel, setiap fungsi juga ada tipe datanya. Tipe data sendiri ada yang namanya int, float, double, char dan ada juga void. Void artinya tidak bertipe data sehingga tidak memerlukan nilai return. Itulah sebabnya kenapa jika kita menggunakan int main(){} maka kita wajib melakukan return 0; di paling bawah fungsi.

Nama Fungsi

Nama fungsi merupakan nama alias yang akan kita gunakan saat pemanggilan. Seperti contoh diatas tadi, jika nama fungsinya adalah belajarfungsi maka saat pemanggilan kita akan menulis seperti ini : belajarfungsi();

Parameter

Parameter merupakan nilai yang akan dijadikan acuan saat menjalankan program. Bisa dikatakan juga parameter adalah pelengkap saat menjalankan fungsi tertentu. Parameter disini dapat dijabarkan menjadi 2 jenis fungsi yakni :

Pass by Value artinya kita akan mempassing value sebagai parameter dalam fungsi yang kita buat, contohnya adalah sebagai berikut :

```
#include <stdio.h>

void cetakUmur(int umur){
    printf("Halo, Umur kamu adalah %d tahun", umur);
}

int main(){
    cetakUmur(20);
```

```
    getchar();
    return 0;
}
```

- **Pass by Address**
  Pass by Address berarti kita akan menggunakan pointer untuk mengakses atau bisa juga mengubah value dari sebuah alamat. Contohnya adalah sebagai berikut :

```
#include <stdio.h>

void cetakUmur(int *umur){
    printf("Halo, Umur kamu adalah %d tahun", *umur);
}

int main(){
    int angka=20;
    cetakUmur(&angka);
    getchar();
    return 0;
}
```

1. Prototipe fungsi

Sebuah fungsi tidak dapat dipanggil kecuali sudah dideklarasikan. Deklarasi fungsi disebut juga sebagai prototipe fungsi. Prototipe ini berupa :

- tipe nilai balik fungsi
- nama fungsi
- jumlah dan tipe argumen
- akhiri dengan ;

Beberapa contoh prototipe fungsi :

1. int fungsi();
2. int fungsi (int angka)
3. float fungsi(float);
4. void garis();

dan masih banyak lagi.

Penggunaan Fungsi menggunakan Void

- Terkadang terdapat fungsi yang tanpa memerlukan adanya pengembalian nilai
- Contoh, sebuah fungsi yang hanya bertugas mencetak kalimat ke layar monitor dan tanpa memerlukan adanya pertukaran parameter
- Kondisi diatas sehingga dipergunakan kata kunci yaitu void
-
-
- **Penggunaan Fungsi menggunakan Void**}

takterhingga.cpp | goto.cpp | while.cpp | dov

```
1   #include<iostream.h>
2   #include<conio.h>
3   void Ucapan(void)
4   {
5    cout<<"Selamat Belajar C++";
6   }
7   int main()
8   {
9    Ucapan();
10   return 0;
11  }
```

Fungsi tanpa Argumen

- **Ada beberapa cara penggunaan fungsi tanpa argumen, diantaranya :**

1. **Pemanggilan dengan nilai**
2. **Pemanggila dengan acuan**

Fungsi tanpa Argumen

takterhingga.cpp | goto.cpp | while.cpp | dowhile.cp

```
1   #include<iostream.h>
2   #include<conio.h>
3
4   void kali (int& a, int& b, int& c)
5   {
6    a *= 2;
7    b *= 2;
8    c *= 2;
9   }
10
11      int main()
12  {
13   int x=1, y=3, z=7;
14   kali(x,y,z);
15   cout<< "x= "<<x<<endl;
16   cout<< "y= "<<y<<endl;
17   cout<< "z= "<<z<<endl;
18   return 0;
19  }
```

Nilai Default dalam Argumen

- Ketika mendeklarasikan fungsi, maka tiap-tiap parameter yang dideklarasikan dapat diberikan sebuah nilai default.
- Nilai default ini akan dipergunakan bila dalam pemanggilan fungsi, tidak diberikan nilai kepada parameter.

Nilai Default dalam Argumen

- Ketika mendeklarasikan fungsi, maka tiap-tiap parameter yang dideklarasikan dapat diberikan sebuah nilai default.
- Nilai default ini akan dipergunakan bila dalam pemanggilan fungsi, tidak diberikan nilai kepada parameter.
- Ketika mendeklarasikan fungsi, maka tiap-tiap parameter yang dideklarasikan dapat diberikan sebuah nilai default.

Nilai default ini akan dipergunakan bila dalam pemanggilan fungsi, tidak diberikan nilai kepada parameter

Nilai Default dalam Argumen

takterhingga.cpp | goto.cpp | while.cpp | dowhile.cpp

```
1   #include<iostream.h>
2   #include<conio.h>
3
4   long factorial (long a)
5   {
6    if (a>1)
7    return (a* factorial (a-1));
8    else
9    return (1);
10  }
11  int main()
12  {
13   long l;
14   cout<<"tuliskan bilangan : ";
15   cin>>l;
16   cout<<"!"<<l<<" = "<<factorial(l);
17   return 0;
18  }
```

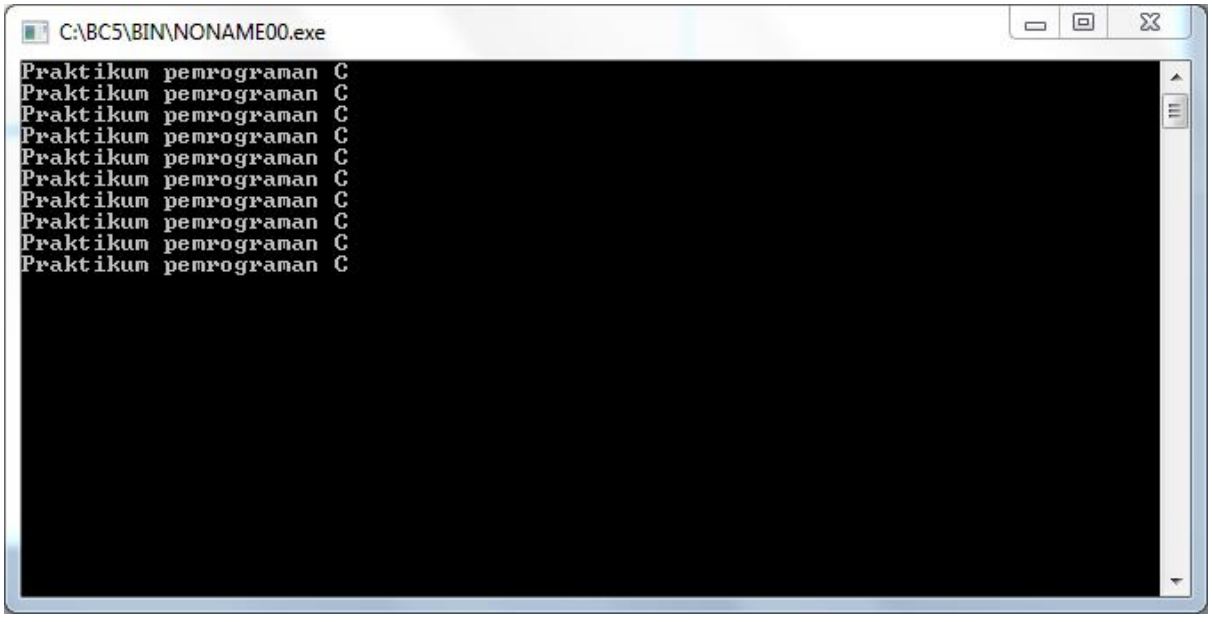
Fungsi Prototype

- Sampai saat ini, setiap dideklarasikan sebuah fungsi baru di letakan di atas fungsi main().
- Berbeda dengan fungsi prototype yang dideklarasikan di bawah fungsi main()
- Bagi compiler, informasi dalam prototype akan dipakai untuk memerikas validitas parameter dalam pemanggilan fungsi

Script :

```
/*simple program example using the function * /

#include “stdio.h”
#include “conio.h”
#include “iostream.h”

void Tulis10kali()

{
for (int = 100 0; C < 10; C++)
{
printf (“Practical programming in C n”);
}
getch();
}

int main ()
{
Tulis10kali();
return 0;
}
```

Output :



Membuat Program Mencari Faktorial Dengan Bahasa C++

Seperti yang kita ketahui, rumus untuk mencari faktorial adalah "n! = (n-1)*(n-2) ...” contoh "faktorial dari 5 adalah 5*4*3*2*1 atau hasilnya adalah 120". dimana dalam kasus ini, angka bersifat bulat. Dari rumus tersebut, dapat kita rumuskan algoritmanya yaitu jika angka yang diinputkan adalah *n* maka hasilnya adalah penjumlahan dari *hasil = (n-1)*(n-2)...* perhatikan listing program berikut ini :

```
#include <iostream>
using namespace std;
int fakt(int a);

int main()
{int z,x;
cout << "Masukkan angka = "; cin >> z;
cout << "Deretnya = ";
for (x=z+1; x>1; x--)
{cout << x-1 << "x";}
cout << "\nHasil = " << fakt(z)<< endl;
}
int fakt(int a)
{
if (a<=1)
 a=1;
else
a=a*(fakt(a-1));
return (a);
}
```

ontoh Program

Contoh ke-1 Iterasi sederhana

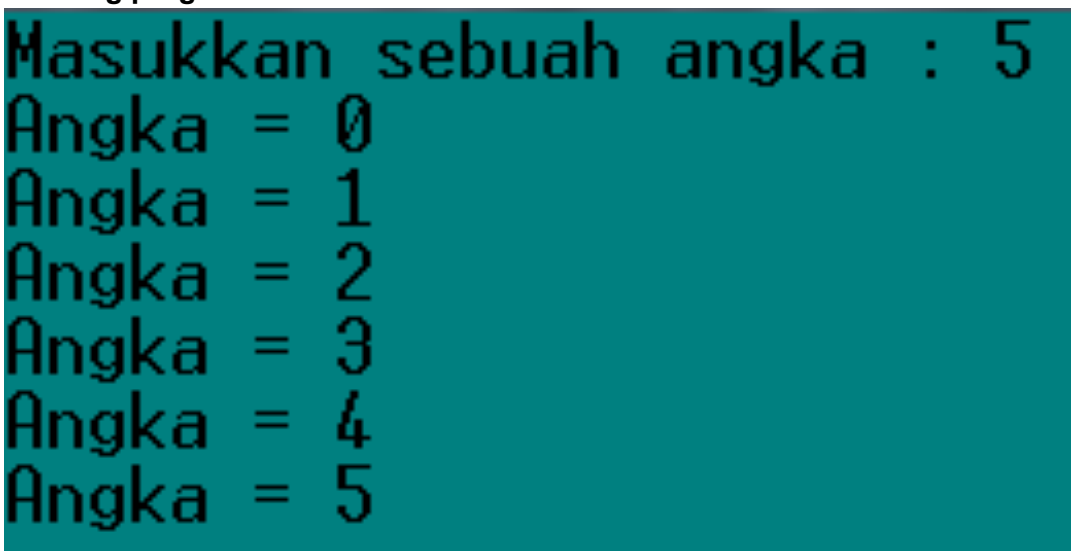
Diketahui algoritma program untuk menampilkan bilangan dari 0 hingga k, dengan k sembarang.
1. Mulai
2. Masukan sebuah anga (k)
3. Untuk indeks (n) = 1 sampai dengan k, tampilkan k
4. Selesai

Source code :

```
#include <iostream>

using namespace std;

main (){
    int n,k;
    cout<<"Masukkan sebuah angka ";cin>>k;
    for (n=0; n<=k; n++)
        cout<<"angka = "<<n<<endl;
}
```

- DEMO

Running program :



Contoh ke-2 Menghitung Faktorial dengan menggunakan iterasi

Dibawah ini terdapat program untuk menghitung Faktorial dengan menggunakan iterasi. Bagaimana algoritmanya?

Source code :

```
#include <iostream>

using namespace std;

long factorial (long a);

int main (){
    int n;
    long hasil;
    cout<<"Menghitung N Faktorial (N!) \n";
    cout<<"Masukan N : ";cin>>n;
    hasil = factorial(n);
    cout<<n<<" = "<<hasil<<endl;

    return 0;
}

long factorial (long a)
{
    long i, hasil1=1;
    for (i=1; i<=a; i++)
        hasil1 = hasil1*i;
    return (hasil1);
}
```

Contoh ke-3 Menampikan deret fibonaci dengan iterasi

Berikut disajikan algoritma untuk menampikan deret fibonaci dengan iterasi.

Algoritma Fibonaci

1. Mulai

2. fib0 = 0, fib1 = 1
3. Selama fib0 <= batas akhir(N) kerjakan langkah 4 sampai dengan 7
4. fib = fib0+fib1
5. fib0 = fib1
6. fib1 = fib
7. Tampilkan fib1
8. Selesai

Algoritma program utama
1. Mulai
2. Masukan batas akhir nilai (N)
3. Panggil fungsi Fibonaci (N)
4. Tampilan deret Fibonaci

Source code :

```
#include <iostream>

using namespace std;

void Fibonaci (int N);

main (){
    int  N;
    long hasil;

    cout<<"Masukan batas akhir dari bilangan fibonaci : ";cin>>N;
    Fibonaci(N);
    cout<<endl;

}

void Fibonaci (int N)
{
    long fib0=0, fib1=1, fib;
    cout<<"Nilai Fibonaci "<<fib0<<" ";
    while (fib0<= N/2)
    {
        fib=fib0+fib1;
        fib0=fib1;
        fib1=fib;
        cout<<fib1<<" ";

    }
}
```

- DEMO

Running program :

```
Masukan batas akhir dari bilangan fibonaci : 100
Nilai Fibonaci 0 1 2 3 5 8 13 21 34 55 89
```

Contoh ke-3 Menampikan deret fibonaci dengan iterasi

Berikut disajikan algoritma untuk menampikan deret fibonaci dengan iterasi.
Algoritma Fibonaci
1. Mulai
2. fib0 = 0, fib1 = 1
3. Selama fib0 <= batas akhir(N) kerjakan langkah 4 sampai dengan 7
4. fib = fib0+fib1
5. fib0 = fib1
6. fib1 = fib
7. Tampilkan fib1
8. Selesai

Algoritma program utama
1. Mulai
2. Masukan batas akhir nilai (N)
3. Panggil fungsi Fibonaci (N)
4. Tampilan deret Fibonaci

Source code :

```
#include <iostream>

using namespace std;

void Fibonaci (int N);

main (){
    int  N;
    long hasil;

    cout<<"Masukan batas akhir dari bilangan fibonaci : ";cin>>N;
    Fibonaci(N);
    cout<<endl;

}

void Fibonaci (int N)
{
    long fib0=0, fib1=1, fib;
    cout<<"Nilai Fibonaci "<<fib0<<" ";
    while (fib0<= N/2)
    {
        fib=fib0+fib1;
        fib0=fib1;
        fib1=fib;
        cout<<fib1<<" ";

    }
}
```

Contoh ke-3 Menampikan deret fibonaci dengan iterasi

Berikut disajikan algoritma untuk menampikan deret fibonaci dengan iterasi.
Algoritma Fibonaci
1. Mulai
2. fib0 = 0, fib1 = 1
3. Selama fib0 <= batas akhir(N) kerjakan langkah 4 sampai dengan 7
4. fib = fib0+fib1

5. fib0 = fib1
6. fib1 = fib
7. Tampilkan fib1
8. Selesai

Algoritma program utama
1. Mulai
2. Masukan batas akhir nilai (N)
3. Panggil fungsi Fibonaci (N)
4. Tampilan deret Fibonaci

Source code :

```
#include <iostream>

using namespace std;

void Fibonaci (int N);

main (){
    int  N;
    long hasil;

    cout<<"Masukan batas akhir dari bilangan fibonaci : ";cin>>N;
    Fibonaci(N);
    cout<<endl;

}

void Fibonaci (int N)
{
    long fib0=0, fib1=1, fib;
    cout<<"Nilai Fibonaci "<<fib0<<" ";
    while (fib0<= N/2)
    {
        fib=fib0+fib1;
        fib0=fib1;
        fib1=fib;
        cout<<fib1<<" ";

    }
}
```

Contoh Penerapan Fungsi Rekursif Pada C++

By Setiawan Dimas | Maret 1, 2020

0 Comment

Rekursif adalah suatu proses dari sebuah fungsi yang dapat memanggil dirinya sendiri secara berulang-ulang. Berbeda dengan fungsi atau prosedur yang mana keduanya hanya bisa dilakukan pemanggilan dari fungsi atau prosedur lain, sementara rekursif dapat memanggil fungsinya sendiri. Jadi **fungsi rekursif c++** ini akan berjalan dengan melakukan proses sampai sebuah kondisi yang ditetapkan pada fungsi tersebut terpenuhi.
Fungsi rekursif adalah salah satu teknik pemrograman yang cukup penting, dimana dalam beberapa kasus menggunakan fungsi rekursif akan jauh lebih mudah. Selain itu proses yang

berjalan akan jauh lebih cepat dan efisien, hanya saja akan membutuhkan *space* memori yang cukup banyak karena proses iterasi dari bagian fungsi tersebut akan dipanggil secara terus menerus sehingga memerlukan ruang penyimpanan yang cukup besar jika dibandingkan dengan proses lainnya.
Baca juga : Contoh Program Struct pada C++

Bahasa pemrograman C++ mendukung penggunaan rekursif. Penerapan fungsi ini juga cukup banyak, yang paling sering misalnya untuk mencari nilai pangkat dan menghitung nilai faktorial. Kali ini saya akan membagikan kepada teman-teman bagaimana contoh penerapan fungsi rekrursif pada C++ melalui 2 contoh sederhana berikut:

Menghitung Nilai Faktorial Dengan Rekursif

```
#include <iostream>

using namespace std;


long int faktorial (int A);


int main(){


          int r,hasil;


          cout<<"MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF"<<endl;

          cout<<endl;

          cout<<"Masukan Nilai = ";

          cin>>r;


          hasil=faktorial(r);

          cout<<"Faktorial "<<r<<"!= "<<hasil<<endl;

}


long int faktorial (int A){

          if (A==1)

                    return(A);

                    else

                    return (A*faktorial(A-1));
```

```
}
```