

Mata Kuliah	:	Dasar Pemrograman
Bobot Sks	:	2
Dosen Pengembang	:	Riad Sahara, S.SI, M.T Syahid Abdullah, S.Si, M.Kom
Tutor	:	Syahid Abdullah, S.Si, M.Kom
Capaian Pembelajaran Mata Kuliah	:	1. Mahasiswa memahami variabel dalam fungsi 2. Mahasiswa memahami fungsi inline 3. Mahasiswa memahami fungsi overloading
Kompetensi Akhir di Setiap Tahap (Sub- Cpmk)		1. Mahasiswa mampu memahami Konsep Fungsi
Minggu Perkuliahan Online Ke-		14

JUDUL TOPIK – Fungsi

Fungsi

Jenis Variabel

Jenis Variabel pada C++ ini sangat berguna didalam penulisan suatu fungsi agar penggunaan didalam penggunaan suatu variabel tidak salah. Terdapat beberapa jenis variabel yaitu:

1. Variabel Lokal.
2. Variabel Eksternal atau Global.
3. Variabel Statis.

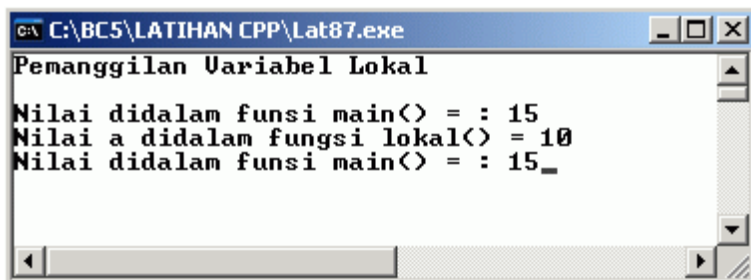
1. Variabel Lokal

Variabel Lokal adalah variabel yang dideklarasikan didalam fungsi dan hanya dikenal oleh fungsi yang bersangkutan. Variabel lokal biasa disebut dengan Variabel Otomatis.

```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
lokal();
main()
{
    int a = 15;
    clrscr();
    cout<<"Pemanggilan Variabel Lokal"<<endl;
    cout<<"\nNilai didalam fungsi main() = : "<<a;
    lokal();
    cout<<"\nNilai didalam fungsi main() = : "<<a;
    getch();
}
lokal()
{
    int a = 10;
    cout<<"\nNilai a didalam fungsi lokal() = "<<a;
}
```

Hal ini terlihat bahwa variabel a yang berada diluar fungsi lokal tidak dikenal oleh fungsi lokal.

Output yang akan dihasilkan, dari program contoh diatas adalah :



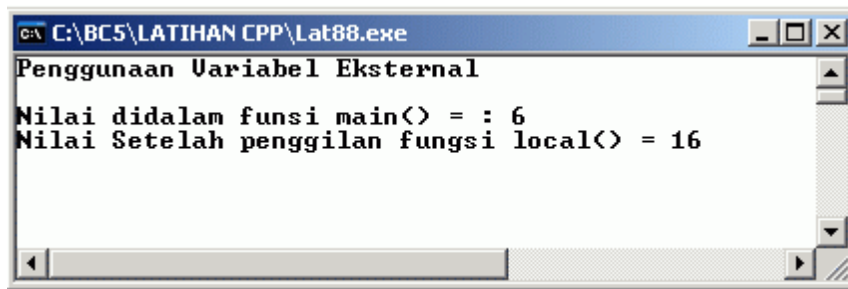
2. Variabel Eksternal

Variabel Eksternal adalah variabel yang dideklarasikan diluar fungsi yang bersifat global yang artinya dapat digunakan bersama-sama tanpa harus dideklarasikan berulang-ulang. Untuk pendeklarasian variabel eksternal ini, diluar dari fungsi main(), yang selama ini pendeklarasian variabel selalu didalam fungsi main().

```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
int a = 6; //--> deklarasi variabel eksternal
void lokal();
void main()
{
    clrscr();
    cout<<"Penggunaan Variabel Eksternal"<<endl;
    cout<<"\nNilai didalam fungsi main() = : "<<a;
    lokal(); //--> pemanggilan fungsi lokal
    cout<<"\nNilai Setelah penggilan fungsi lokal() = ";
    cout<<a;
    getch();
}
void lokal()
{
    a+=10;
}
```

Hal ini terlihat bahwa variabel a yang dideklarasikan diluar fungsi main(), dapat digunakan didalam fungsi main() dan fungsi lokal().

Output yang akan dihasilkan, dari program contoh-8 diatas adalah :



3. Variabel Statis

Variabel Statis dapat berupa variabel local atau variabel eksternal. Sifat variabel statis ini mempunyai sifat antar lain.

- Jika variabel statis bersifat local, maka variabel hanya dikenal oleh fungsi tempat variabel dideklarasikan.
- Jika variabel statis bersifat eksternal, maka variabel dapat dipergunakan oleh semua fungsi yang terletak pada file yang sama ditempat variabel statis dideklarasikan.
- Jika tidak ada inisialisasi oleh pemrograman secara otomatis akan diberikan nilai awal nol.

Suatu variabel statis diperoleh dengan menambahkan kata-kunci static didepan penentu tipe data variabel.

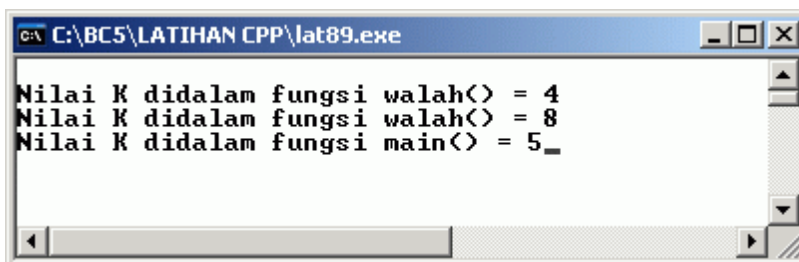
```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
walah(); //--> prototipe fungsi walah
main()
{
    int k = 5;
    clrscr();
    walah();
    walah();
    cout<<"\nNilai K didalam fungsi main() = "<<k;
    getch();
}
```

```
}  
walah()  
{  
    static int k; // --> deklarasi variabel statis  
    k += 4;  
    cout<<"\nNilai K didalam fungsi() = "<<k;  
}
```

Hal ini terlihat bahwa :

- Pada pada prototipe fungsi `walah()` tidak tedapat nilai awal, maka secara otomatis variabel `k = 0`.
- Pada pemanggilan fungsi `walah()` pertama, tercetak nilai variabel `k = 4`, didapat dari `k=0+4`.
- Pada pemanggilan fungsi `walah()` kedua, tercetak nilai variabel `k = 8`, didapat dari `k=4+4`, karena nilai `k` yang terbaru adalah 4.
- Pada pencetakan `k` didalam fungsi `main()`, adalah 5, karena variabel `k`, didalam fungsi `main()` bersifat lokal.

Output yang akan dihasilkan, dari program contoh diatas adalah :



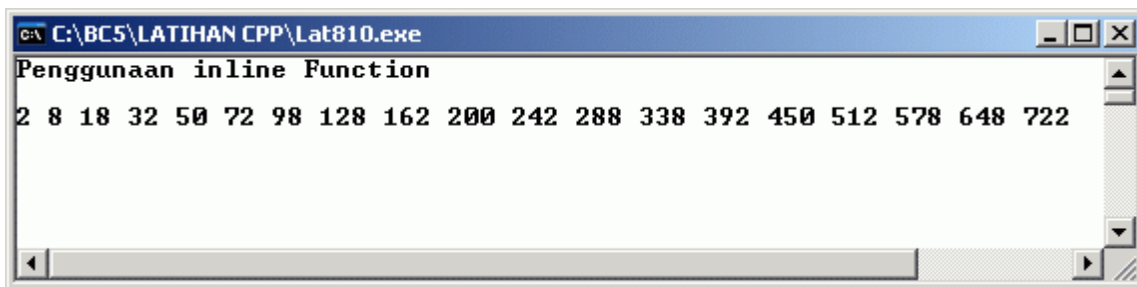
Fungsi inline

Fungsi inline (inline function) digunakan untuk mengurangi lambatnya eksekusi program dan mempercepat eksekusi program terutama pada program yang sering menggunakan atau memanggil fungsi yang berlebih. terutama program-program yang menggunakan pernyataan perulangan proses seperti `for`, `while` dan `do – while`. Inline function dideklarasikan dengan menambahkan kata kunci `inline` didepan tipe data.

```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
inline int kali(int i, int j)
{
    return(i * j);
}

main()
{
    int k;
    clrscr();
    for(k = 1; k < 20; k++)
        cout<<kali(k, k*2)<<" ";
    getch();
}
```

Output yang akan dihasilkan, dari program contoh diatas adalah :

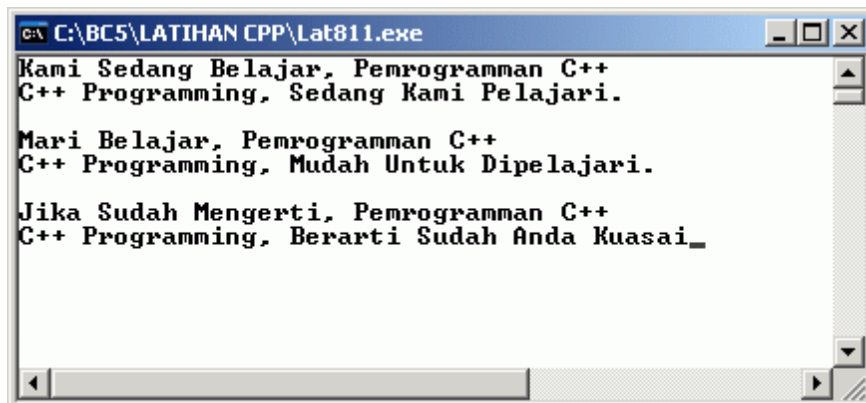


```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
inline static void cplusplus()
{
```

```
cout << "Pemrogramman C++\n";
cout << "C++ Programming, ";
}

int main()
{
    {
        cout << "Kami Sedang Belajar, ";
        cplusplus();
        cout << "Sedang Kami Pelajari.\n\n";
    }
    {
        cout << "Mari Belajar, ";
        cplusplus();
        cout << "Mudah Untuk Dipelajari.\n\n";
    }
    {
        cout << "Jika Sudah Mengerti, ";
        cplusplus();
        cout << "Berarti Sudah Anda Kuasai";
    }
    getch();
}
```

Output yang akan dihasilkan, dari program contoh diatas adalah:



Function Overloading

Function Overloading adalah mendefinisikan beberapa fungsi, sehingga memiliki nama yang sama tetapi dengan parameter yang berbeda. Dapat diartikan bahwa fungsi yang overload berarti menyediakan versi lain dari fungsi tersebut. Salah satu kelebihan dari C++ adalah Overloading. Sebagai contoh membentuk fungsi yang sama dengan tipe yang berbedabeda dan dibuatkan pula nama fungsi yang berbeda-beda pula.

```
#include<conio.h>
#include<stdio.h>
#include<iostream.h>
int hitung(int b);
long hitung(long c);
float hitung(float d);
void main()
{
    clrscr();
    cout<< "Hasilnya Fungsi overload -1 : ";
    cout<<hitung(4)<<endl;
    cout<< "Hasilnya Fungsi overload -2 : ";
    cout<<hitung(2)<<endl;
```



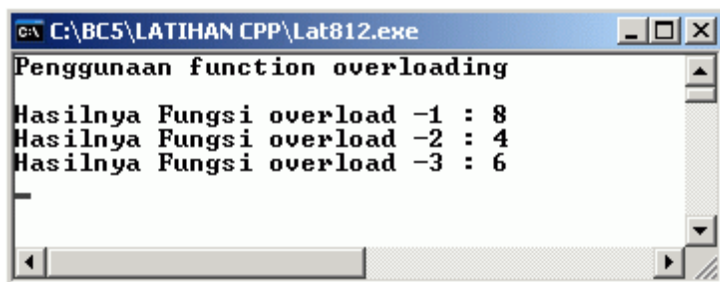
```
cout<< "Hasilnya Fungsi overload -3 : ";  
cout<<hitung(3)<<endl;  
getch();  
}
```

```
int hitung(int b)  
{  
    return(b*b);  
}
```

```
long hitung(long c)  
{  
    return(c*c);  
}
```

```
double hitung(double d)  
{  
    return(d*d);  
}
```

Output yang akan dihasilkan, dari program contoh diatas adalah :



```
C:\BC5\LATIHAN CPP\Lat812.exe  
Penggunaan function overloading  
Hasilnya Fungsi overload -1 : 8  
Hasilnya Fungsi overload -2 : 4  
Hasilnya Fungsi overload -3 : 6
```

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.