[](https://1.bp.blogspot.com/-jfHFQ1ZKZAk/VrXwBeAa5dI/AAAAAAAADGo/LxFz2PPfN8I/s1600/overloading.jpg)

Setelah kemarin kita pernah membahas mengenai indetifier yang berarti bahwa sebuah variabel / function yang kita buat tidak boleh memiliki tanda atau nama yang sama di dalam satu ruang lingkup.  Nama variabel / function diperbolehkan untuk menggunakan tanda baca A-Z / a-z, (\_) dan 0 sampai 9. Tapi tidak untuk penamaan yang sama dalam variabel satu ruang lingkup. Dan kali ini kita akan membahas mengenai function dan parameter yang mempunyai nama yang sama tapi dengan tipe data yang berbeda dan itu masih sah untuk dilakukan.

#include <iostream>

using namespace std;

int penghitung (int a, int b)

{

   return (a\*b);

}

float penghitung (float a, float b)

{

   return (a/b);

}

int main ()

{

   int x=5,y=2;

   float n=5.0,m=2.0;

   cout << penghitung (x,y) << '\n';

   cout << penghitung (n,m) << '\n';

   return 0;

}

Di dalam mendirikan function, dua function yang berbeda dapat memiliki ama yang sama jika parameter mereka berbeda, baik mereka memiliki nomer yang berbeda dari parameter maupun karena setap parameter mereka dari tipe data yang berbeda atau function itu memiliki tipe data yang berbeda.

Dalam contoh diatas saya membuat dua function yang memiliki nama yang sama yaitu “penghitung” dan parameternyapun memiliki nama variabel yang sanam yaitu a dan b. Yang berbeda hanylah tipe data yang digunakan yaitu int dan float.Dan untuk bagian output.

  cout << penghitung (x,y) << '\n';

  cout << penghitung (n,m) << '\n';

mereka memanggil nama yang sama tetapi mengapa compiler masih bisa berjalan tanpa ada bentrokan, yup. Ada code Sebelumnya sebelum code output diatas itu, telah didirikan variabel x,y sebagai int dan n,m sebagai float. Dan compiler tahu mana yang akan dia panggil dalam setiap kasus dengan memeriksa jenis variabel sebagai agrumen ketika fungs dipanggil. Compiler akan mencari function “penghitung” yang memiliki tipe data yang sama dengan variabel x,y dan n,m. Misalnya x,y merupakan variabel pertipe int dan compiler akan mencarikan function bertipe data int meskipun ada banyak nama yang sama.

Agar kita bsa melihat perbedaantersebut, saya membuat contoh program diatas dengan operasi yang berbeda, int sebagai perkalian dan float sebagai pembagi.dengan ini anda akan mudah bisa melihat perbedaan tersebut. Tetapi sebenarnya ini merupakan yang baik, tetapi dalam penjelasan ini untuk menunjukan cara ini bahwa ini memungkinkan /bisa digunakan. Dan semua ini dinamakan sebagai overloaded Function.

Function Overloading/Overloaded 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. S Sebagai contoh membentuk fungsi yang sama dengna tipe yang berbeda-beda dan dibuatkan pula nama fungsi yang berbeda-beda pula. intinya adalah dua function dengan nama yang sama tetapi memiliki karakteristik yang berbeda dan ini mungkin untuk digunakan tanpa ada bentrok/kesalahan dalam menterjemahkan.

Contoh Lain Penggunaan Function Overloading

#include<conio.h>

#include<stdio.h>

#include<iostream.h>

using namescape std;

int hitung(int b);

long hitung(long c);

float hitung(float d);

int 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);

}

Baca juga :

* [Pengertian dan Struktur Function C++](http://belajarcpplus.blogspot.com/2016/01/pengertian-dan-struktur-function-cplusplus.html)
* [Pengertian Prototipe dan Parameter Function C++](http://belajarcpplus.blogspot.com/2016/01/pengertian-prototipe-dan-parameter-function-cplusplus.html)
* [Cara dan Contoh Pemanggilan Dengan Nilai (Call by Value) C++](http://belajarcpplus.blogspot.com/2016/01/cara-dan-contoh-pemanggilan-dengan-nilai-call-by-value-cplusplus.html)
* [Cara dan Contoh Pemanggilan Dengan Referensi (Call by Reference) C++](http://belajarcpplus.blogspot.com/2016/01/cara-dan-contoh-pemanggilan-dengan-referensi-call-by-reference-cplusplus.html)
* [Pengertian dan Contoh Pernyataan Return() C++](http://belajarcpplus.blogspot.com/2016/01/pengertian-dan-contoh-pernyataan-return-cplusplus.html)
* [Pengertian dan Jenis-jenis Variabel C++](http://belajarcpplus.blogspot.com/2016/01/pengertian-dan-jenis-jenis-variabel-cplusplus.html)
* [Pengertian dan Contoh Function Inline C++](http://belajarcpplus.blogspot.com/2016/01/pengertian-dan-contoh-function-inline-cplusplus.html)
* [Pengertian dan Contoh Structure C++](http://belajarcpplus.blogspot.co.id/2016/01/pengertian-dan-contoh-structure-cplusplus.html)
* [Cara dan Contoh Structure dengan Array pada Function C++](http://belajarcpplus.blogspot.co.id/2016/01/cara-dan-contoh-structure-dengan-array-pada-function-cplusplus.html)

Cukup sekian dari belajarcpp.com semoga artikel ini bisa bermanfaat untuk kita semua. Mohon maaf atas kekuranganya dan Terima Kasih atas dukungan dan kunjungan anda ke BelajarCPP. Have a nice day.

Last update : 25 Maret 2016