Learning Computer Programming

A resource for Learning Computer Programming. I post articles, tips, tricks, techni algorithms etc. related to C++, PHP and other programming language. Many of th things that I discuss here applies to most languages.

WEDNESDAY, AUGUST 15, 2007

Overloading the Assignment Operator (=)

Labels: Class, Methods, Operator Overloading

We know that if we want objects of a class to be operated by common operators then we need to *overload them*. But there is one operator whose operation is automatically crested by C++ for every class we define, it is the assignment operator '='.

Actually we have been using similar statements like the one below previously

```
ob1=ob2;
```

where ob1 and ob2 are objects of a class.

This is because even if we don't overload the '=' operator, the above statement is valid.

As I said C++ automatically creates a default assignment operator. The default operator created, does a member-by-member copy, but if we want to do something specific we may overload it.

The simple program below illustrates how it can be done. Here we are defining two similar classes, one with the default assignment operator (created automatically) and the other with the overloaded one. Notice how we could control the way assignments are done in that case.

```
// Program to illustrate the
// overloading of assignment
// operator '='
#include <iostream.h>

// class not overloading the
// assignment operator
class myclass
{
  int a;
```

RECENT POSTS

RECEN COMM

CATEGORIES

- Algorithms
- Apache
- Atom
- Blogger
- Blogging
- CAPTCHA
- Complete Script
- CSS
- Example Programs
- Functions
- Game
- HTML
- IDE
- Image
- JavaScript
- Memory Allocation
- Merging
- Methods
- Miscellaneous
- OOP
- Operator Overloading
- Personal
- Personel
- PHP
- Problems
- Programming Skills
- Questions
- Searching
- Securing
- Software
- Strings
- Tricks
- Useful Scripts
- Virtual Functions
- WAP
- Web
- Web Scraping
- XML


```
  int b;

public:
  myclass(int, int);
  void show();
};

myclass::myclass(int x,int y)
{
  a=x;
  b=y;
}

void myclass::show()
{
  cout<<a<<endl<<b<<endl;
}

// class having overloaded
// assignment operator
class myclass2
{
  int a;
  int b;

public:
  myclass2(int, int);
  void show();

  myclass2 operator=(myclass2);
};

myclass2 myclass2::operator=(myclass2 ob)
{
  // -- do something specific --
  // this is just to illustrate
  // that when overloading '='
  // we can define our own way
  // of assignment
  b=ob.b;

  return *this;
};

myclass2::myclass2(int x,int y)
{
  a=x;
  b=y;
}

void myclass2::show()
{
  cout<<a<<endl<<b<<endl;
}

// main
void main()
{
  myclass ob(10,11);
  myclass ob2(20,21);

  myclass2 ob3(100,110);
  myclass2 ob4(200,210);
```

```
    // does a member-by-member copy
    // '=' operator is not overloaded
    ob=ob2;
    ob.show();


    // does specific assignment as
    // defined in the overloaded
    // operator definition
    ob3=ob4;
    ob3.show();
}
```

Related Articles:

- Overloading Post-Fix Forms of ++ and -- Operators
- Overloading Increment/Decrement Operators
- Introduction to Operator Overloading in C++ II
- Introduction to Operator Overloading in C++
- Introduction to Function Overloading in C++

Posted by Arvind Gupta at 7:14 PM

Reactions: funny (0) interesting (0) cool (0)

10 comments:

Saughmraat May 12, 2008 10:42 PM

Hi Arvind Gupta!

First of all.. I'd like to appreciate you for sharing your knowledge in your blog.

In the assignment operator overloading, please mention a warning about the pitfalls of the same, when the member variables are pointers to some data.

I don't know if you did so some where else in your blog..

But, it'll be nice to see a little more info on deep copy and shadow copy.

any how...

this blog is wonderful.

keep rocking.

I always admire the people who are ready to share their knowledge.

Reply

Arvind Gupta May 13, 2008 03:06 AM

Thanks for appreciating!

I'll try to do what you've stated maybe in some of the future posts.

Thanks once again!

Reply

Syed Azhar May 10, 2009 07:48 PM

I am trying to overload = in a Binary Tree class.

The binary tree class has a pointer as member function. So I cannot rely on default overload.

But even if I am overloading manually I am not getting the expected results.

I think it is because of the pointers as mentioned by Saughmraat.

It would be great if one could find the answer to such problem here is this blog as this blog looks good and google is giving this as first link !!!

Thanks anyways.

Reply

ANURAG Nov 10, 2009 08:44 PM

Why can't we overload the = (Assignment Operator) Through friend function???

please send me the answer of this question to my email id.
nema.anu@gmail.com

Regards

Reply

Anonymous Jul 6, 2010 03:42 AM

I would like to overload = operator as follows:

```
class Number
{
private:
int n;
public:
Number(int i) : n(i) {}
int GetN() {return n;}
.............
}

int main()
{
int x;
Number m(5);

x = m;
}
```

is it possible;

Reply

Arvind Gupta Jul 7, 2010 09:08 PM

@ Anonymous,

Yeah, that is possible but it's not quite overloading. When you have a constructor that takes one parameter the compiler does that for you.

Reply

Anonymous Aug 17, 2010 12:19 PM

why dont u just give the output along with input ?
i think for beginners like us it will be easy to understand

Reply

Anonymous Aug 30, 2011 06:42 AM

Overloading the Assignment Operator (+)?
Overloading the Assignment Operator (-)?
Overloading the Assignment Operator (/)?

Can you kindly show me how to solve them?

e.g

Rational operator+(const Rational& r);
Should create a new dynamic Rational object, sets its values to current object + the object r and
return it.
new.n = n * r.d + d * r.n
new.d = d * r.d

A rational number is a number that can be expressed as a fraction or ratio (rational). The numerator n
and the denominator d of the fraction are both integers. Assume the convention that rational number
is typed and displayed as n/d where n is the value of the numerator value and d is the value of the
denominator.

Reply

shashank Sep 17, 2011 11:31 AM

Thnx a lot for such explanation.
I am a tayro in C++.
Can you tell me why we use return *this;???

Reply

Anonymous Sep 29, 2011 08:39 AM

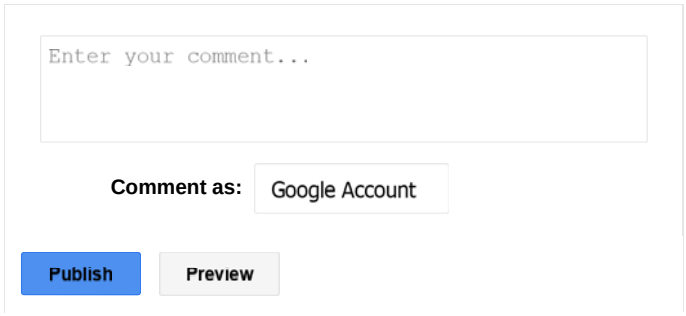
is it mandatory to overload assignment operator by making it as a member function.can v overload it through friend function also?
reply sooooooooon

Reply

Add comment

You are free to comment anything, although you can comment as 'Anonymous'
it is strongly recommended that you supply your name. Thank You.

Please don't use abusive language.



Newer Post Home Older Post

Subscribe to: Post Comments (Atom)