06-04-2021

#include <iostream>

using namespace std;

class IncreDecre

{

int a,b; //data member

public :

void accept() //member function

{

cout<<"enter two numbers :"<<endl;

cin>>a>>b;

}

void operator --() //operator function to overload

{

a--;

b--;

}

void operator ++() //operator function

{

a++;

b++;

}

void display()

{

cout<<"\n A is :"<<a;

cout<<"\n B is :"<<b<<endl;;

}

};

int main()

{

IncreDecre id; //decalaration of object

id.accept(); //calling

--id; //calling operator function

cout<<"\n After decrementing :"<<endl;

id.display ();

id.accept();

++id; //calling operator function

cout<<"\n After Incrementing :"<<endl;

id.display();

return 0;

}

//Overloading unary operator with user defined return type//

#include <iostream>

using namespace std;

class count

{

int value;

public:

count() // by-default constructor

{

value=10;

}

count operator --()

{

count temp;//declaration of an object

temp.value= ++value;

return temp;

};

void display()

{

cout<<"Count is :"<<value<<endl;

}

};

int main()

{

count obj1,result;

result=--obj1;

result.display();

return 0;

}

07-04-21

/\* operator overloading in binary operator

a+b //operands(a,b) operator (+,>,==)

a>b

a==b\*/

#include <iostream>

using namespace std;

class A

{

int x; //data member

public :

A() // default constructor

{}

A(int i) //paramerterized constructor

{

x=i;

}

void operator +(A);//delcaration of binary function//(binary object will be pass as an arguement)

};

void A :: operator +(A a)

{

int m = x+a.x;

cout<<"Addition of a and b :"<<m<<endl;

}

int main()

{

A a(7); //declaration of an object

A b(5);

a+b; //calling

return 0;

}

9-04-2021

/\*\*

Friend operator overloding function/ Binary operator overloading using friend function \*/

#include <iostream>

using namespace std;

class Test2;

class Test1

{

int a; //data member

public:

void get\_a()

{

cout<<"Enter the no. of a "<<endl;

cin>>a;

}

friend void operator >(Test1,Test2);

};

class Test2

{

int b;

public:

void get\_b()

{

cout<<"Enter the value of b:"<<endl;

cin>>b;

}

friend void operator >(Test1, Test2);

};

void operator >( Test1 t1,Test2 t2)

{

t1.a>t2.b?cout<<"A is greater no:":cout<<"B is greater no";

}

int main()

{

Test1 t1;

Test2 t2;

t1.get\_a();

t2.get\_b();

t1>t2;

return 0;

}

#include <iostream>

using namespace std;

class mst2;

class mst1

{

int a; //data member

public:

void get\_a()

{

cout<<"Enter the no of a"<<endl;

cin>>a;

}

friend void operator >(mst1,mst2);

};

class mst2

{

int b;

public:

void get\_b()

{

cout<<"Enter the value of b :"<<endl;

cin>>b;

}

friend void operator >(mst1,mst2);

};

void operator >(mst1 obj1,mst2 obj2)//explicit argument

{

obj1.a>obj2.b ?cout<<"A is the greater number":cout<<"B is greater number:"<< endl;;

//conditional operator

}

int main()

{

mst1 obj1;

mst2 obj2;

obj1.get\_a();

obj2.get\_b();

obj1>obj2;

return 0;

}

//binary operator overloading using friend function (oeprand of a same class)

#include <iostream>

using namespace std;

class A

{

private: // data member

int a;

public:

void set\_a(); //declaration of the member function

void get\_a();

friend A operator \*(A,A);//binary operator \* overloaded friend, which takes two object as arguements

};

// definition of set\_a() function

void A :: set\_a() //set a fun belongs to A

{

a=5;

}

//defination of get\_b() function

void A::get\_a()

{

cout<<a<<"\n";

}

//defination of overloaded binary operator \* friend function

A operator\* (A t1,A t2)

{

A obj1;

obj1.a =t1.a\* t2.a;

return obj1;

}

int main()

{

A t1 , t2;

t1.set\_a();

t2.set\_a();

cout<<"the value of a in the first object :"<<endl;

t1.get\_a();

cout<<"the value of a in sencond object : "<<endl;

t2.get\_a();

A obj2 = t1\* t2;

cout<<"the value of a after the calling operator overloading function \* is :";

obj2.get\_a();

}

13-04-15

#include<iostream>

using namespace std;

class A

{

public:

int x,y;

};

int main()

{

//Pointer to member variable x of class A

int A::\* p1 = &A :: x;

//Pointer to member variable y of class A

int A::\* p2 = &A :: y;

//Creating an object of class A

A ob;

ob.\*p1 = 100; // ob.x=100

ob.\*p2 = 10;

cout<<"The value of x is : " << ob.\*p1 << "\n";

cout<<"The value ot y is : " << ob.\*p2 << "\n";

}