

Step-by-step walkthrough for example on
page 53 of the lecture notes:
C++ Class

```

class Complex /* File: complex.h */
{
    private:
        float real; float imag;
    public:
        Complex(float r, float i) { real = r; imag = i; }
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }

        Complex add1(const Complex& x) // Return by value
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
        Complex* add2(const Complex& x)
        // Return by value using pointer
        {
            real += x.real; imag += x.imag;
            return this;
        }
        Complex& add3(const Complex& x)
        // Return by reference
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
};

```

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	<input type="text"/>
imag	<input type="text"/>

```
class Complex /* File: complex.h */  
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return (*this);
```

```
}
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return this;
```

```
}
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

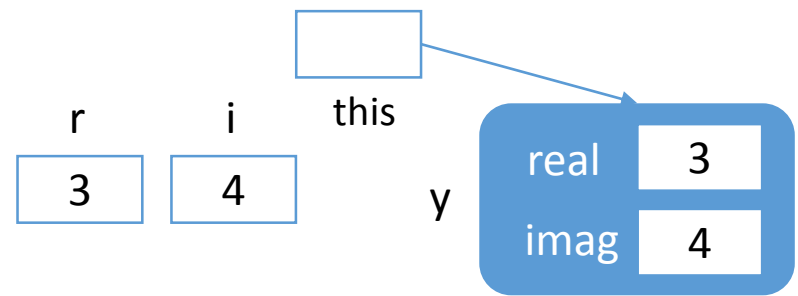
```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return (*this);
```

```
}
```

```
};
```



```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

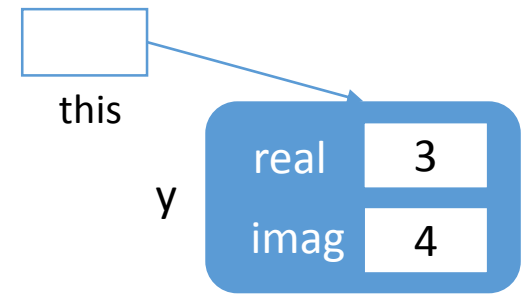
real	3
imag	4

```

class Complex /* File: complex.h */
{
    private:
        float real; float imag;
    public:
        Complex(float r, float i) { real = r; imag = i; }
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }

        Complex add1(const Complex& x) // Return by value
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
        Complex* add2(const Complex& x)
        // Return by value using pointer
        {
            real += x.real; imag += x.imag;
            return this;
        }
        Complex& add3(const Complex& x)
        // Return by reference
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
};

```



Output
(3 , 4)

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	3
imag	4

Output

(3 , 4)

Return by value

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```

y

real	3
imag	4

x

real	
imag	

Output

(3 , 4)

Return by value


```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

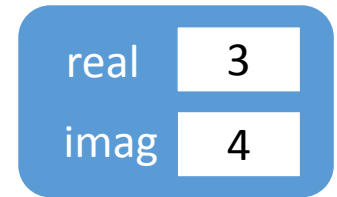
```
            return (*this);
```

```
        }
```

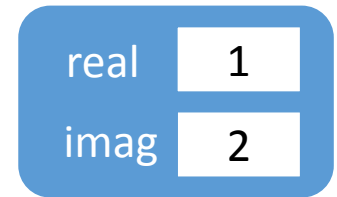
```
};
```



`y`



`x`



`this`

Output

(3 , 4)

Return by value

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```

y

real	3
imag	4

x

real	1
imag	2

Output

(3 , 4)

Return by value

```

class Complex /* File: complex.h */
{
    private:
        float real; float imag;
    public:
        Complex(float r, float i) { real = r; imag = i; }
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }

        Complex add1(const Complex& x) // Return by value
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
        Complex* add2(const Complex& x)
        // Return by value using pointer
        {
            real += x.real; imag += x.imag;
            return this;
        }
        Complex& add3(const Complex& x)
        // Return by reference
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
};

```

y

real	3
imag	4

x

real	1
imag	2



this

Output

(3 , 4)

Return by value
(1 , 2)

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```

y

real	3
imag	4

x

real	1
imag	2

Output

(3 , 4)

Return by value
(1 , 2)

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
Complex add1(const Complex& x) // Return by value
```

```
{
    real += x.real; imag += x.imag;
    return (*this);
}
```

```
Complex* add2(const Complex& x)
```

```
// Return by value using pointer
```

```
{
    real += x.real; imag += x.imag;
    return this;
}
```

```
Complex& add3(const Complex& x)
```

```
// Return by reference
```

```
{
    real += x.real; imag += x.imag;
    return (*this);
}
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

x



this

real	1
imag	2

Output

(3 , 4)

Return by value

(1 , 2)

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

x

this

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
    Complex add1(const Complex& x) // Return by value
```

```
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
```

```
    Complex* add2(const Complex& x)
```

```
    // Return by value using pointer
```

```
    {
        real += x.real; imag += x.imag;
        return this;
    }
```

```
    Complex& add3(const Complex& x)
```

```
    // Return by reference
```

```
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

```
    Complex add1(const Complex& x) // Return by value
```

```
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
```



this

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

temp

Generated by return (*this)

real	4
imag	6

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    temp x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value
(1 , 2)

temp

real	4
imag	6


```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
Complex add1(const Complex& x) // Return by value
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return (*this);
```

```
}
```

```
Complex* add2(const Complex& x)
```

```
// Return by value using pointer
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return this;
```

```
}
```

```
Complex& add3(const Complex& x)
```

```
// Return by reference
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return (*this);
```

```
}
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

this

temp

real	4
imag	6

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

this

temp

real	7
imag	10

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
    Complex add1(const Complex& x) // Return by value
```

```
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
```

```
    Complex* add2(const Complex& x)
    // Return by value using pointer
```

```
    {
        real += x.real; imag += x.imag;
        return this;
    }
```

```
    Complex& add3(const Complex& x)
```

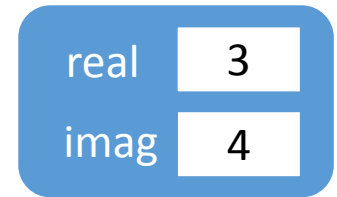
```
    // Return by reference
```

```
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
```

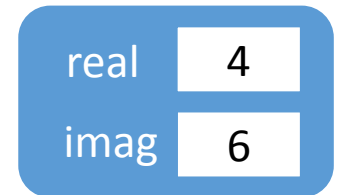
```
};
```

Nickname at add1

x
y

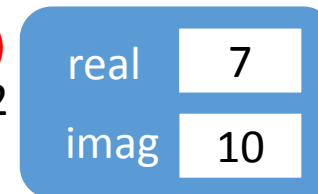


x



Generated by return (*this)

temp2



Output

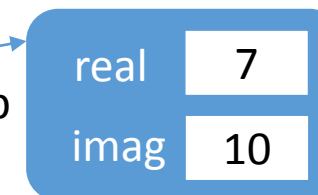
(3 , 4)

Return by value
(1 , 2)



this

temp



```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print(); temp2
x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value
(1 , 2)

temp2

real	7
imag	10

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
};
```

Nickname at add1

x
y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)



this

temp2

real	7
imag	10

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```

y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

```

class Complex /* File: complex.h */
{
    private:
        float real; float imag;
    public:
        Complex(float r, float i) { real = r; imag = i; }
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }

        Complex add1(const Complex& x) // Return by value
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
        Complex* add2(const Complex& x)
        // Return by value using pointer
        {
            real += x.real; imag += x.imag;
            return this;
        }
        Complex& add3(const Complex& x)
        // Return by reference
        {
            real += x.real; imag += x.imag;
            return (*this);
        }
};

```

y

real	3
imag	4

x

real	4
imag	6



this

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value


```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```

y

real	3
imag	4

x

real	4
imag	6

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return (*this);
```

```
    }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return this;
```

```
    }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

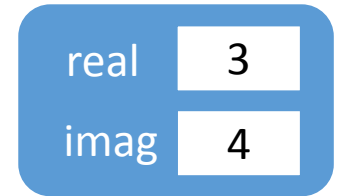
```
        return (*this);
```

```
    }
```

```
};
```

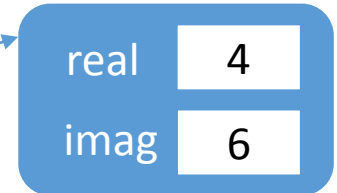
Nickname at add1

x
y



this

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return (*this);
```

```
    }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return this;
```

```
    }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

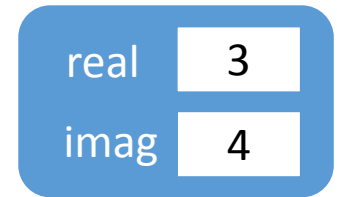
```
        return (*this);
```

```
    }
```

```
};
```

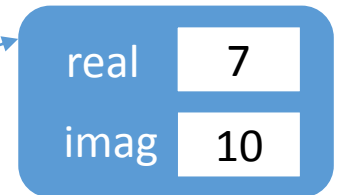
Nickname at add1

x
y



this

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return (*this);
```

```
    }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

```
        return this;
```

```
    }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
    {
```

```
        real += x.real; imag += x.imag;
```

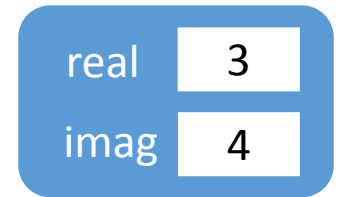
```
        return (*this);
```

```
    }
```

```
};
```

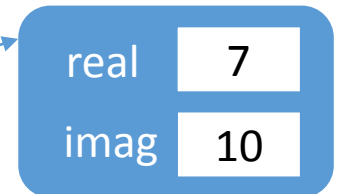
Nickname at add1

x
y



this

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

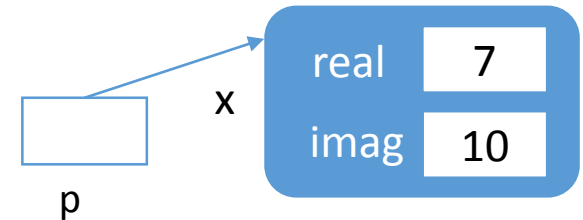
Return its pointer by value

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

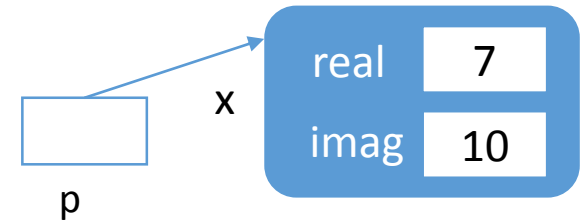
Return its pointer by value

```

#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"

int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}

```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

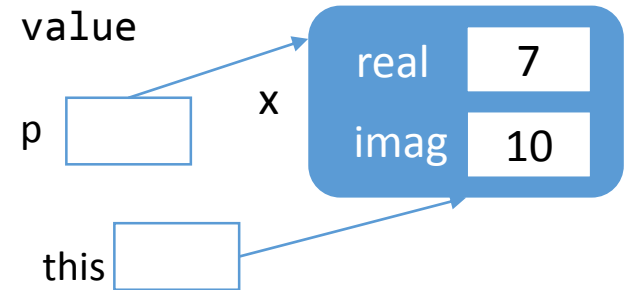
```
            return (*this);
```

```
        }
```

```
};
```

x
y

real	3
imag	4



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

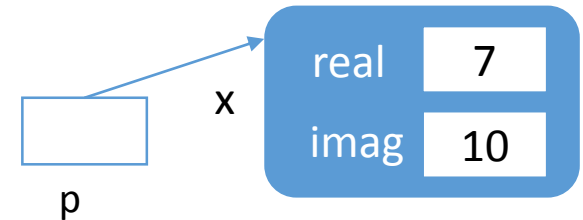
(4 , 6)

Return its pointer by value

(7 , 10)

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

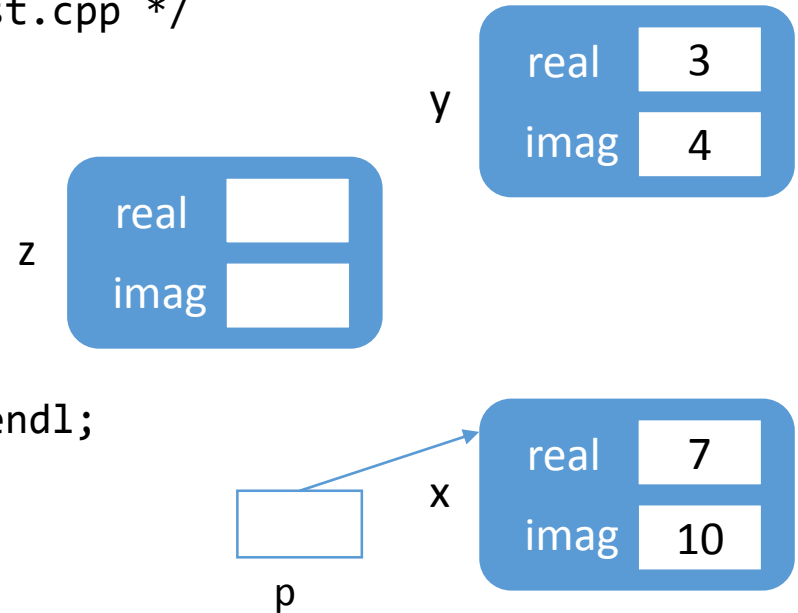
Return its pointer by value

(7 , 10)

Return by reference


```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

```

class Complex /* File: complex.h */
{
private:
    float real; float imag;
public:
    Complex(float r, float i) { real = r; imag = i; }
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }
};

```

Diagram illustrating the state of a `Complex` object `z` and its interaction with another object `x`.

Object `z` (labeled `r` and `i`):

real	1
imag	2

Object `x` (labeled `real` and `imag`):

real	3
imag	4

The diagram shows a box labeled `z` with `real` and `imag` fields, and a box labeled `x` with `real` and `imag` fields. An arrow points from a box labeled `this` to the `z` box.

```

Complex add1(const Complex& x) // Return by value
{
    real += x.real; imag += x.imag;
    return (*this);
}

```

```

Complex* add2(const Complex& x)
// Return by value using pointer
{
    real += x.real; imag += x.imag;
    return this;
}

```

```

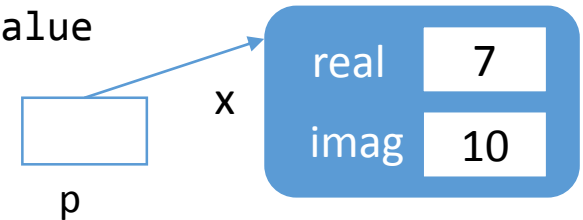
Complex& add3(const Complex& x)
// Return by reference
{
    real += x.real; imag += x.imag;
    return (*this);
}

```

```

};

```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

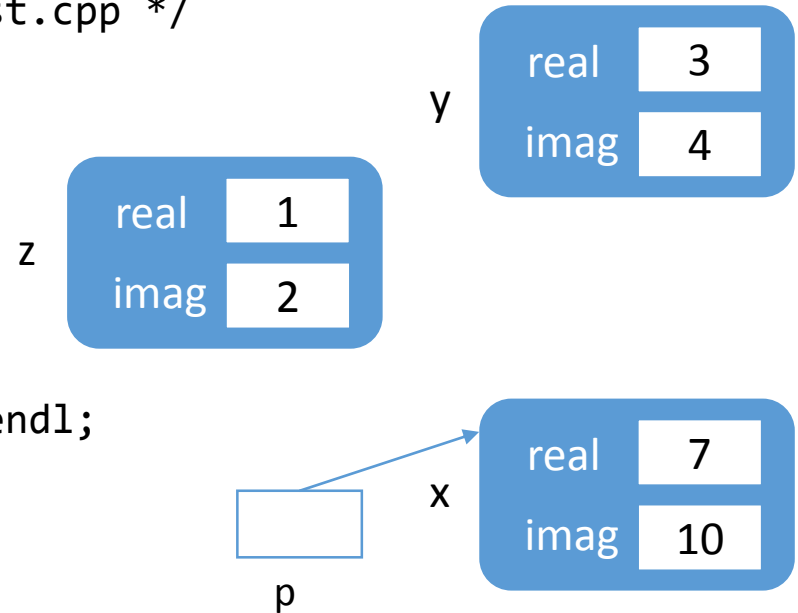
Return its pointer by value

(7 , 10)

Return by reference

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

```

class Complex /* File: complex.h */
{
private:
    float real; float imag;
public:
    Complex(float r, float i) { real = r; imag = i; }
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }

    Complex add1(const Complex& x) // Return by value
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
    Complex* add2(const Complex& x)
    // Return by value using pointer
    {
        real += x.real; imag += x.imag;
        return this;
    }
    Complex& add3(const Complex& x)
    // Return by reference
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
};

```

Diagram illustrating the state of the `Complex` class:

- `this` (pointing to the `Complex` object) has `real = 1` and `imag = 2`.
- `z` has `real = 1` and `imag = 2`.
- `y` has `real = 3` and `imag = 4`.
- `p` (pointing to the `Complex` object) has `real = 7` and `imag = 10`.
- `x` has `real = 7` and `imag = 10`.

Output:

```

( 3 , 4 )

Return by value
( 1 , 2 )
( 7 , 10 )
( 4 , 6 )

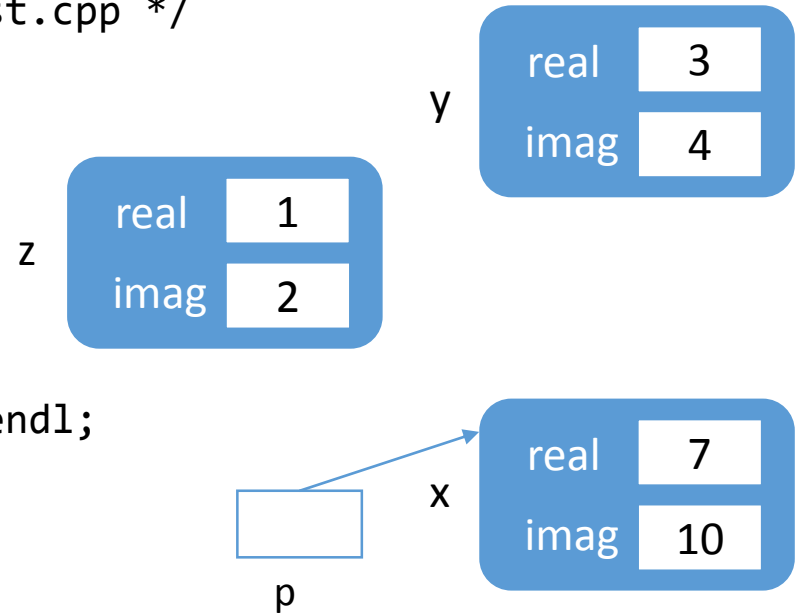
Return its pointer by value
( 7 , 10 )

Return by reference
( 1 , 2 )

```

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

```
class Complex /* File: complex.h */
{
```

```
private:
```

```
float real; float imag;
```

```
public:
```

```
Complex(float r, float i) { real = r; imag = i; }
```

```
void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
Complex add1(const Complex& x) // Return by value
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return (*this);
```

```
}
```

```
Complex* add2(const Complex& x)
```

```
// Return by value using pointer
```

```
{
```

```
    real += x.real; imag += x.imag;
```

```
    return this;
```

```
}
```

```
Complex& add3(const Complex& x)
```

```
// Return by reference
```

```
{
```

```
    real += x.real; imag += x.imag;
```

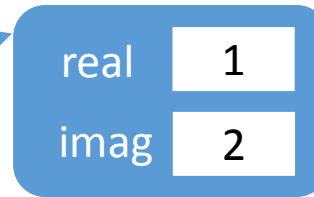
```
    return (*this);
```

```
}
```

```
};
```



z

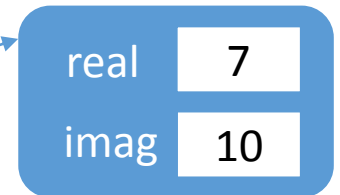


x
y



p

x



Output

```
( 3 , 4 )
```

```
Return by value
```

```
( 1 , 2 )
```

```
( 7 , 10 )
```

```
( 4 , 6 )
```

```
Return its pointer by value
```

```
( 7 , 10 )
```

```
Return by reference
```

```
( 1 , 2 )
```

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
};
```



z

real

4

imag

6

x
y

real

3

imag

4



p

x

real

7

imag

10

Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

(1 , 2)

Generated by return (*this)

```
class Complex /* File: complex.h */  
{
```

```
private:
```

```
    float real; float imag;
```

```
public:
```

```
    Complex(float r, float i) { real = r; imag = i; }
```

```
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
    Complex add1(const Complex& x) // Return by value
```

```
{  
    real += x.real; imag += x.imag;  
    return (*this);  
}
```

```
    Complex* add2(const Complex& x)
```

```
// Return by value using pointer
```

```
{  
    real += x.real; imag += x.imag;  
    return this;  
}
```

```
    Complex& add3(const Complex& x)
```

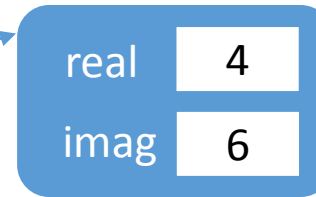
```
// Return by reference
```

```
{  
    real += x.real; imag += x.imag;  
    return (*this);  
}
```

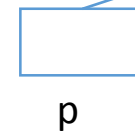
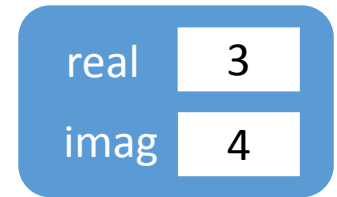
```
};
```



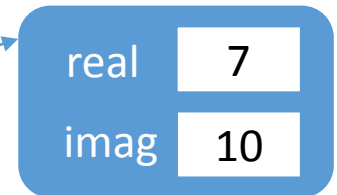
z



x
y



x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

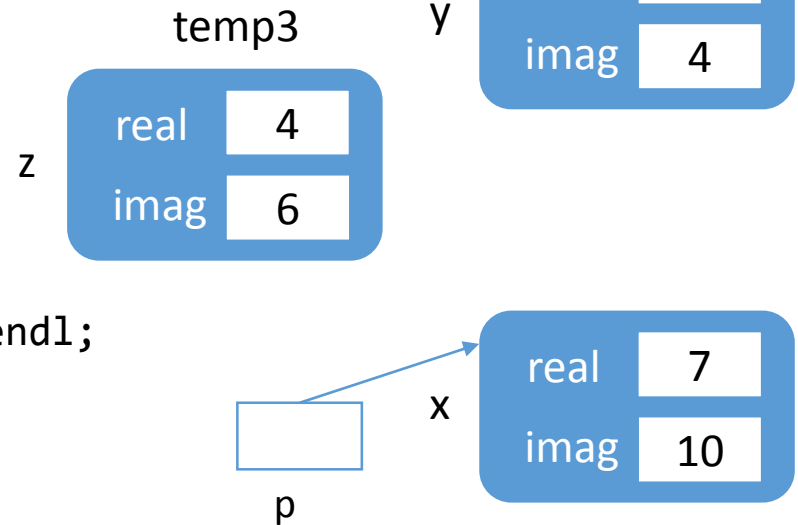
(7 , 10)

Return by reference

(1 , 2)


```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    temp3 z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

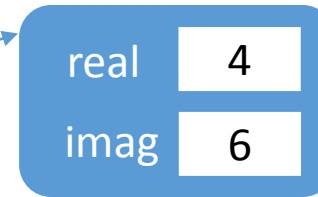
```
            return (*this);
```

```
        }
```

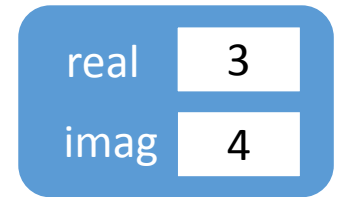
```
};
```



z

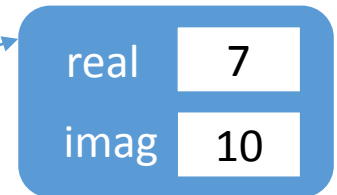


x
y



p

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

(1 , 2)

```

class Complex /* File: complex.h */
{
private:
    float real; float imag;
public:
    Complex(float r, float i) { real = r; imag = i; }
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }

    Complex add1(const Complex& x) // Return by value
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
    Complex* add2(const Complex& x)
    // Return by value using pointer
    {
        real += x.real; imag += x.imag;
        return this;
    }
    Complex& add3(const Complex& x)
    // Return by reference
    {
        real += x.real; imag += x.imag;
        return (*this);
    }
};

```

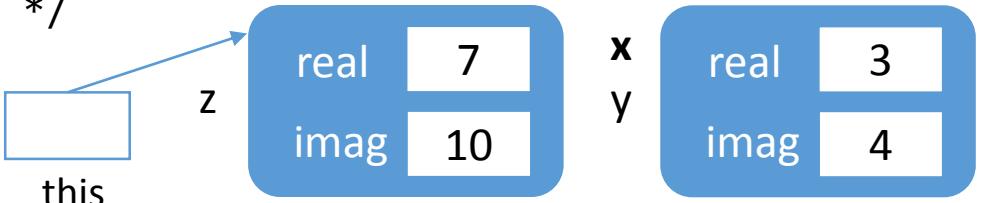


Diagram illustrating the state of variables `z` and `y`. Variable `z` points to a `Complex` object with `real = 7` and `imag = 10`. Variable `y` points to a `Complex` object with `real = 3` and `imag = 4`.

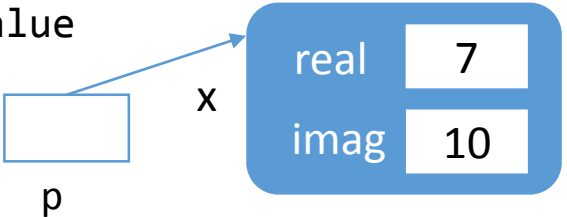


Diagram illustrating the state of variables `p` and `x`. Both `p` and `x` point to the same `Complex` object with `real = 7` and `imag = 10`.

Output

```

( 3 , 4 )

Return by value
( 1 , 2 )
( 7 , 10 )
( 4 , 6 )

Return its pointer by value
( 7 , 10 )

Return by reference
( 1 , 2 )

```

Generated by return (*this)

```
class Complex /* File: complex.h */  
{
```

```
private:
```

```
    float real; float imag;
```

```
public:
```

```
    Complex(float r, float i) { real = r; imag = i; }
```

```
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
    Complex add1(const Complex& x) // Return by value
```

```
{  
    real += x.real; imag += x.imag;  
    return (*this);  
}
```

```
    Complex* add2(const Complex& x)
```

```
// Return by value using pointer
```

```
{  
    real += x.real; imag += x.imag;  
    return this;  
}
```

```
    Complex& add3(const Complex& x)
```

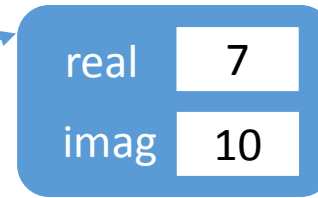
```
// Return by reference
```

```
{  
    real += x.real; imag += x.imag;  
    return (*this);  
}
```

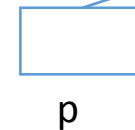
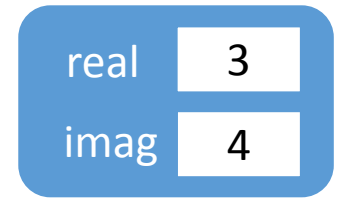
```
};
```



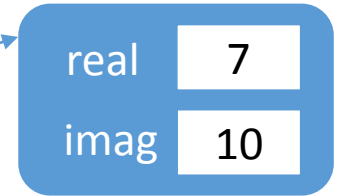
z



x
y



x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

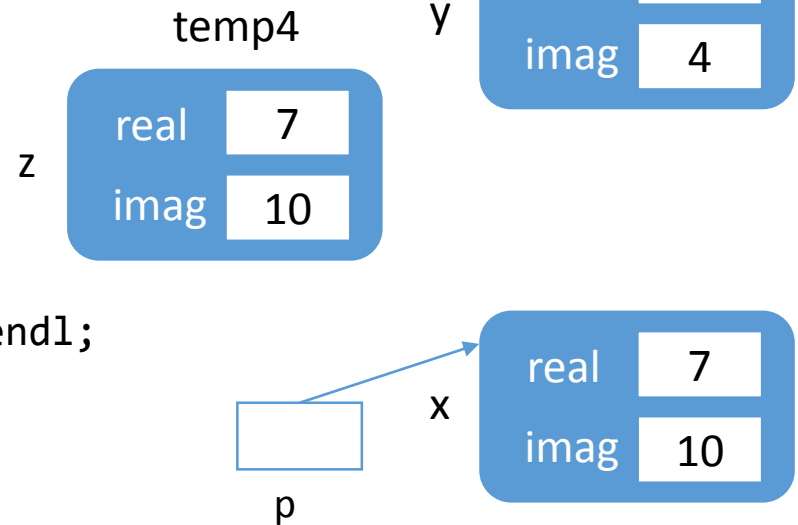
(7 , 10)

Return by reference

(1 , 2)

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    temp4 z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

(1 , 2)

```
class Complex /* File: complex.h */
{
```

```
    private:
```

```
        float real; float imag;
```

```
    public:
```

```
        Complex(float r, float i) { real = r; imag = i; }
```

```
        void print() { cout << "( " << real << " , " << imag << " )" << endl; }
```

```
        Complex add1(const Complex& x) // Return by value
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return (*this);
```

```
        }
```

```
        Complex* add2(const Complex& x)
```

```
        // Return by value using pointer
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

```
            return this;
```

```
        }
```

```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

```
        {
```

```
            real += x.real; imag += x.imag;
```

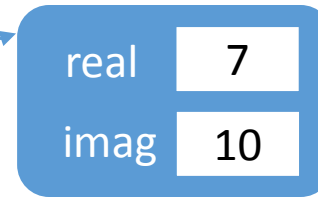
```
            return (*this);
```

```
        }
```

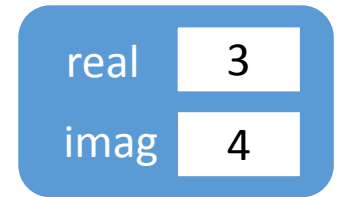
```
};
```



z

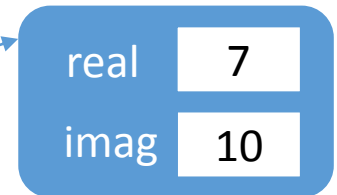


x
y



p

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

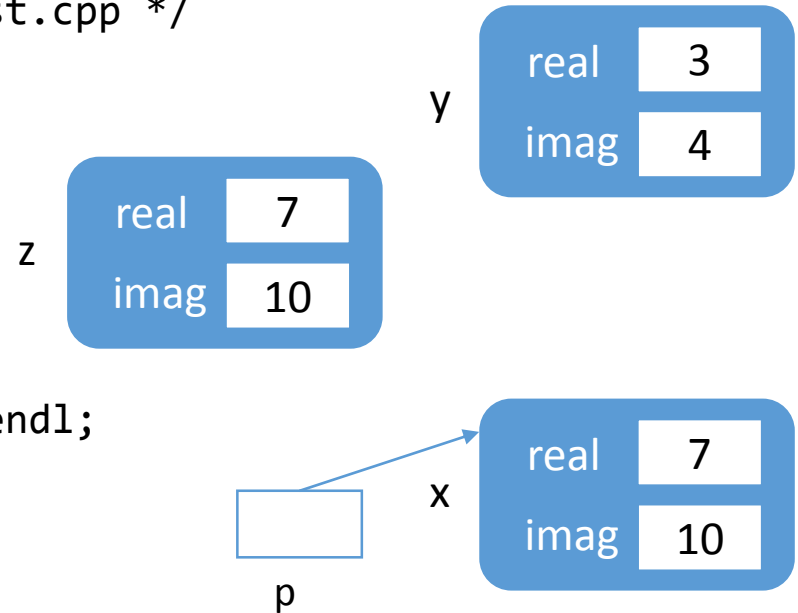
Return by reference

(1 , 2)

(7 , 10)

```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

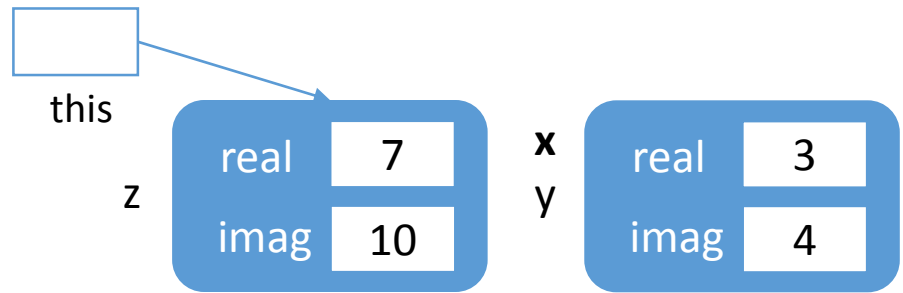
(1 , 2)

(7 , 10)

```

class Complex /* File: complex.h */
{
private:
    float real; float imag;
public:
    Complex(float r, float i) { real = r; imag = i; }
    void print() { cout << "( " << real << " , " << imag << " )" << endl; }

```



```

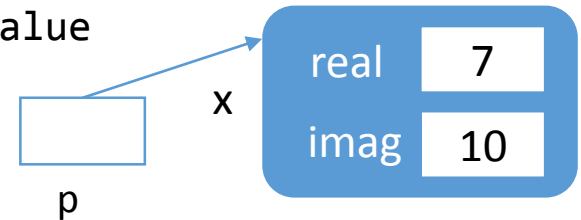
Complex add1(const Complex& x) // Return by value

```

```

{
    real += x.real; imag += x.imag;
    return (*this);
}

```



```

Complex* add2(const Complex& x)
// Return by value using pointer
{
    real += x.real; imag += x.imag;
    return this;
}

```

```

Complex& add3(const Complex& x)
// Return by reference
{
    real += x.real; imag += x.imag;
    return (*this);
}

```

```

};

```

Output

```

( 3 , 4 )

```

Return by value

```

( 1 , 2 )

```

```

( 7 , 10 )

```

```

( 4 , 6 )

```

Return its pointer by value

```

( 7 , 10 )

```

Return by reference

```

( 1 , 2 )

```

```

( 7 , 10 )

```

```

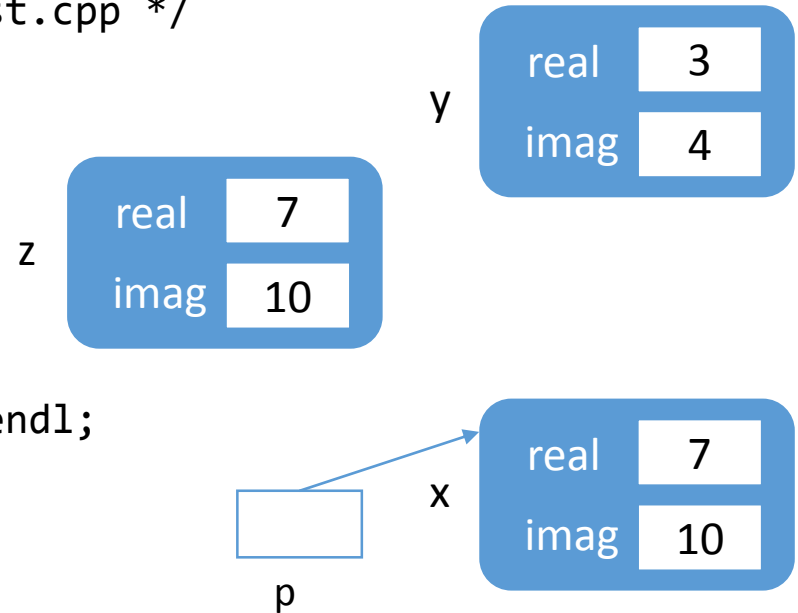
( 7 , 10 )

```



```
#include <iostream> /* File: complex-test.cpp */
using namespace std;
#include "complex.h"
```

```
int main()
{
    Complex y(3, 4);
    y.print();
    cout << endl << "Return by value" << endl;
    Complex x(1, 2);
    x.print();
    x.add1(y).add1(y).print();
    x.print();
    cout << endl << "Return its pointer by value" << endl;
    Complex* p = x.add2(y);
    p->print();
    cout << endl << "Return by reference" << endl;
    Complex z(1, 2);
    z.print();
    z.add3(y).add3(y).print();
    z.print();
    return 0;
}
```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

(7 , 10)

Return by reference

(1 , 2)

(7 , 10)

(7 , 10)