

Step-by-step walkthrough for example on
page 34 – 35 of the lecture notes:
Data Abstraction & Classes

```

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
```

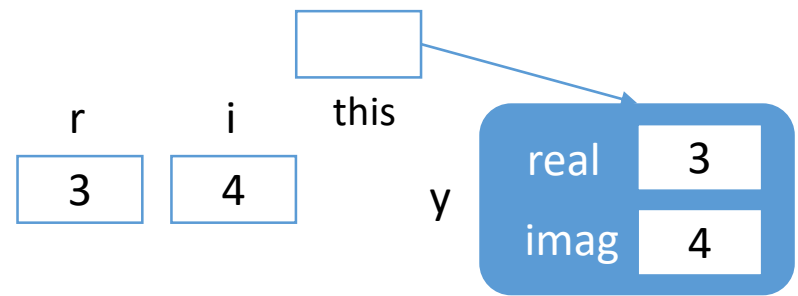
```
{
```

```
    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

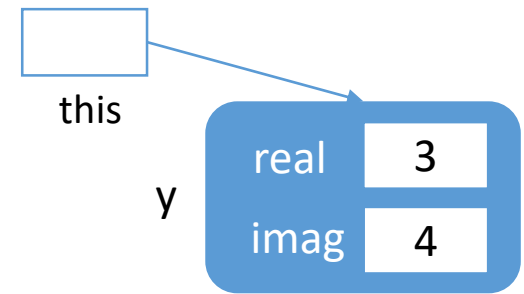
real	3
imag	4

```

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;
```

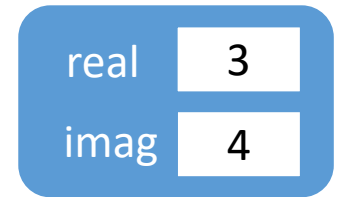
```
            return (*this);
```

```
        }
```

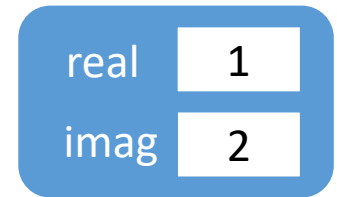
```
};
```



`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)
```

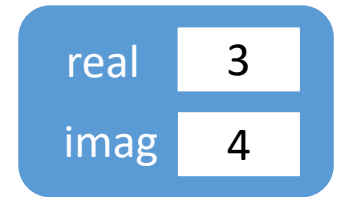
```
    // Return by reference
```

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

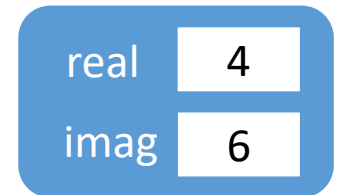
```
};
```

Nickname at add1

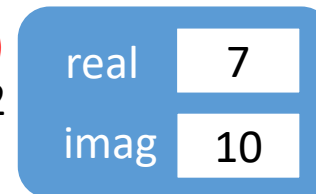
x
y



x



Generated by return (*this)
temp2



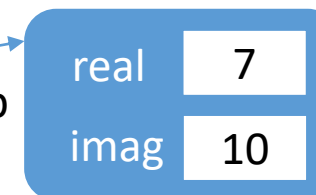
Output

(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;
```

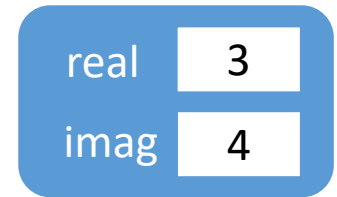
```
        return (*this);
```

```
    }
```

```
};
```

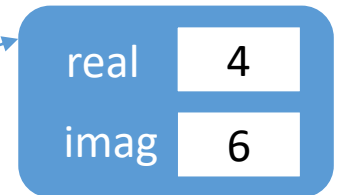
Nickname at add1

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;
```

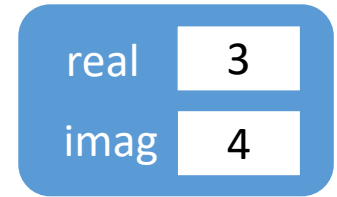
```
        return (*this);
```

```
    }
```

```
};
```

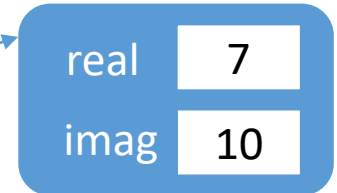
Nickname at add1

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)
```

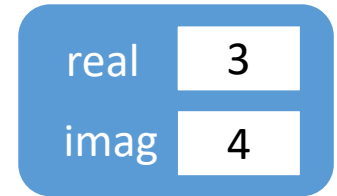
```
        // Return by reference
```

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

```
};
```

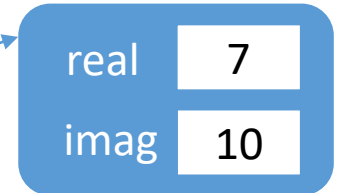
Nickname at add1

x
y



this

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

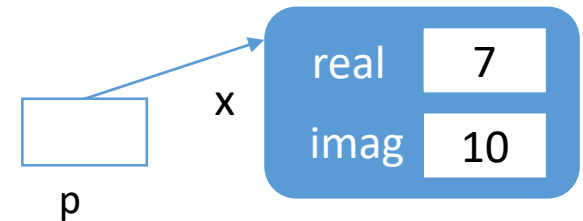
Return its pointer by value

```

#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)

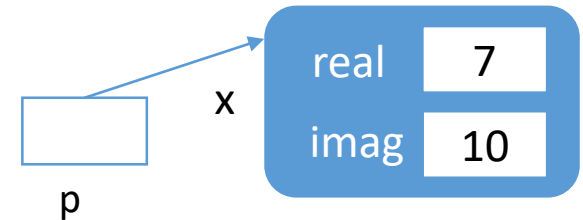
Return its pointer by value

```

#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;
        }
```

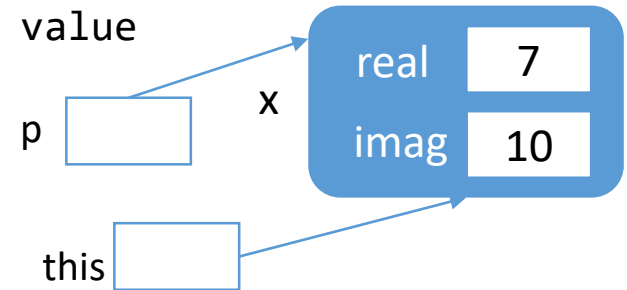
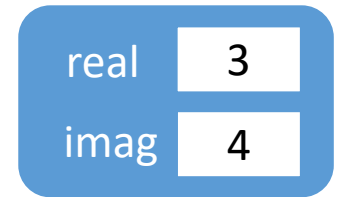
```
        Complex& add3(const Complex& x)
```

```
        // Return by reference
```

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

```
};
```

x
y



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

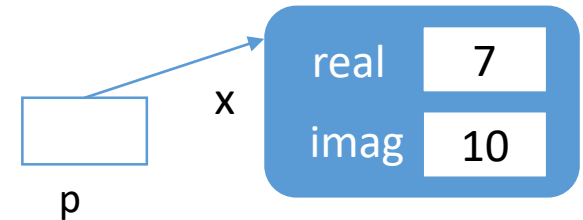
(4 , 6)

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)

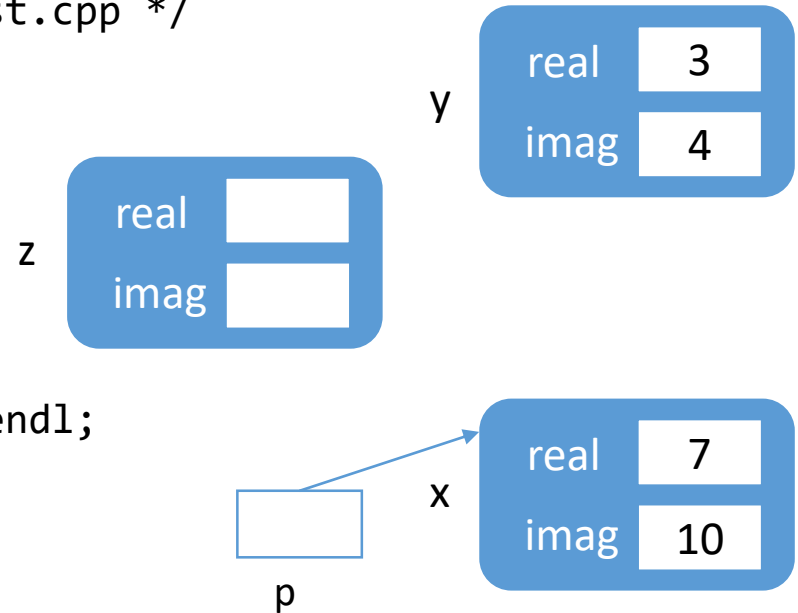
Return its pointer by value

(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 pointer `z` pointing to the first object, and a pointer `x` pointing to the second object.

```

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

```

```

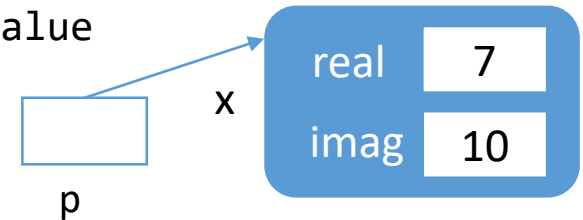
Complex& add3(const Complex& x)
// Return by reference
{
    real += x.real; imag += x.imag;
    return (*this);
}

```

```

};

```



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

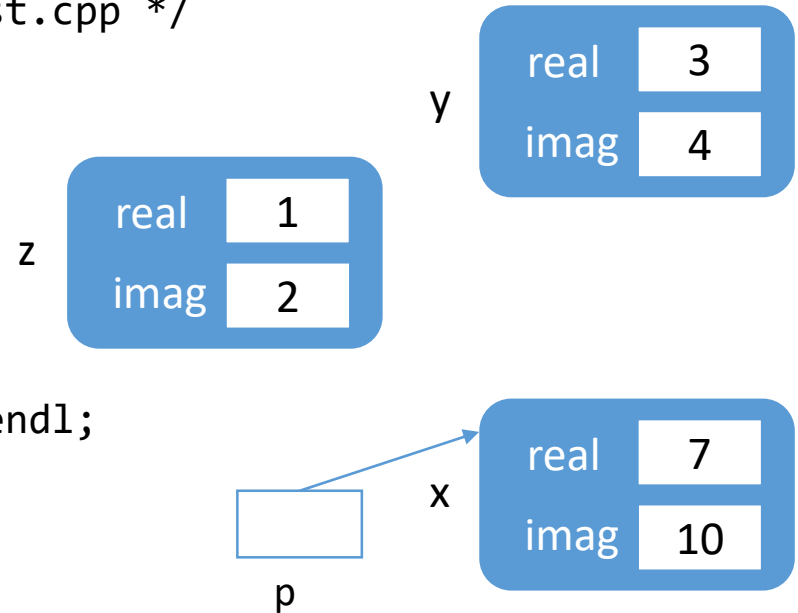
Return its pointer by value

(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:

- A box labeled `this` points to the `Complex` object `z`.
- Object `z` has `real = 1` and `imag = 2`.
- Object `y` has `real = 3` and `imag = 4`.
- Object `x` has `real = 7` and `imag = 10`.
- A box labeled `p` points to object `x`.

Output:

```

( 3 , 4 )

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

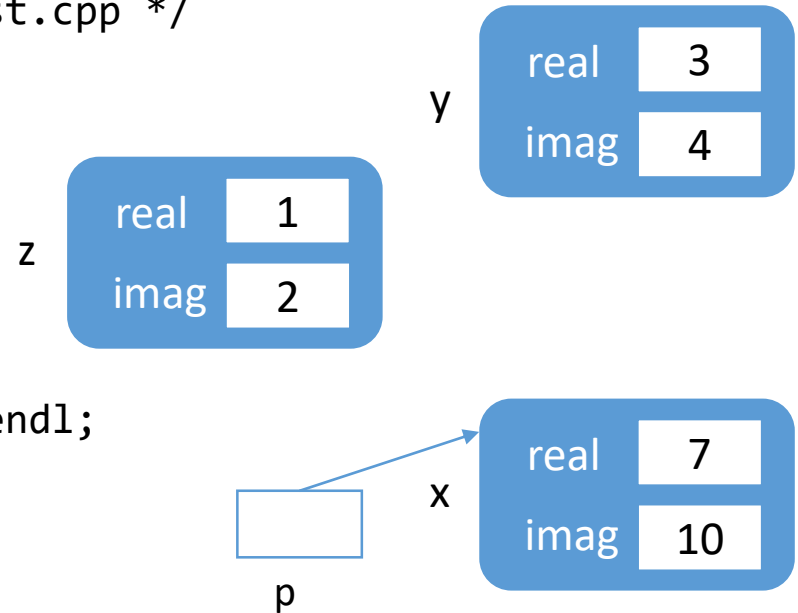
Return its pointer by value
( 7 , 10 )

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();
    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:

- A variable `z` (represented by a box) points to a `Complex` object with `real = 1` and `imag = 2`. This object is labeled `this`.
- A variable `x` (represented by a box) points to a `Complex` object with `real = 3` and `imag = 4`. This object is labeled `x`.
- A variable `y` (represented by a box) points to a `Complex` object with `real = 3` and `imag = 4`. This object is labeled `y`.
- A variable `p` (represented by a box) points to a `Complex` object with `real = 7` and `imag = 10`. This object is labeled `p`.

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 the `Complex` class:

- A variable `z` (represented by a box) points to a `Complex` object with `real = 4` and `imag = 6`.
- A variable `x` (represented by a box) points to a `Complex` object with `real = 3` and `imag = 4`.
- A variable `p` (represented by a box) points to a `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);
```

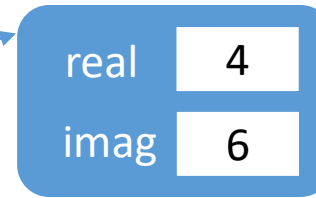
```
}
```

```
};
```

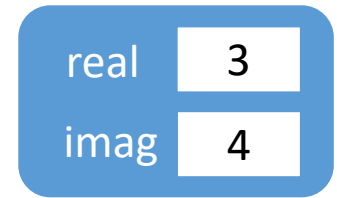


this

z

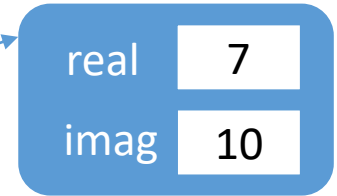


x
y



p

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

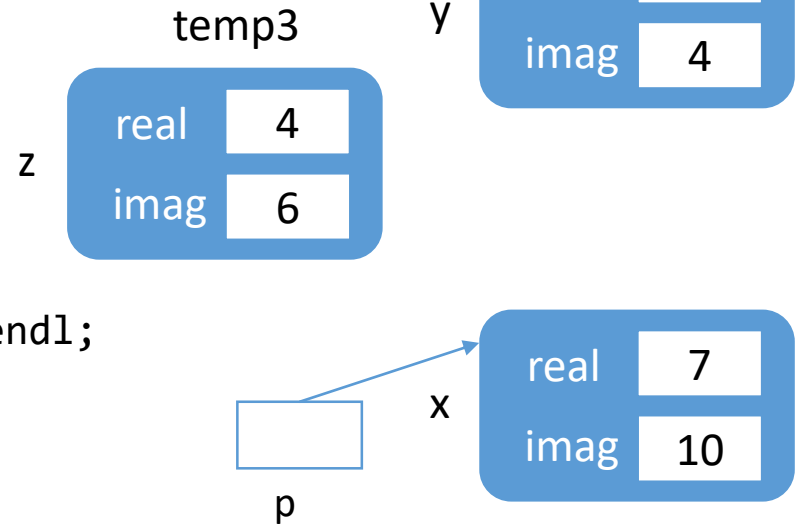
(7 , 10)

Return by reference

(1 , 2)


```
#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);
    }
};

```

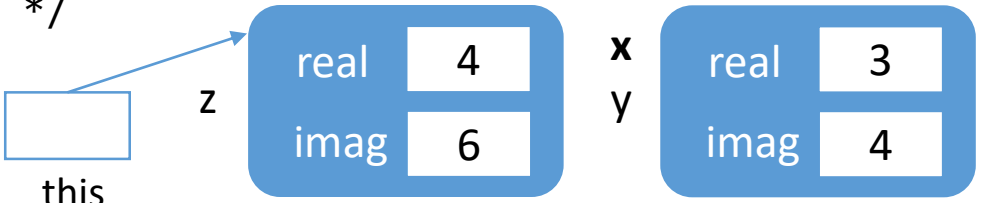


Diagram illustrating the state of variables `z` and `x` after the first two lines of the `main` function. Variable `z` points to a `Complex` object with `real = 4` and `imag = 6`. Variable `x` points to a `Complex` object with `real = 3` and `imag = 4`.

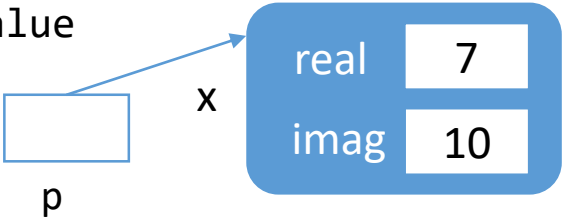


Diagram illustrating the state of variable `p` after the `add1` function call. Variable `p` points to a `Complex` object with `real = 7` and `imag = 10`, which is the result of the `add1` function.

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 the `Complex` class:

- A variable `z` (represented by a box) points to a `Complex` object with `real = 7` and `imag = 10`. This object is labeled `this`.
- A variable `x` (represented by a box) points to a `Complex` object with `real = 7` and `imag = 10`.
- A variable `y` (represented by a box) points to a `Complex` object with `real = 3` and `imag = 4`.

Diagram illustrating the state of the `Complex` class after the `add1` function call:

- A variable `p` (represented by a box) points to a `Complex` object with `real = 7` and `imag = 10`. This object is labeled `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 )

```

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
```

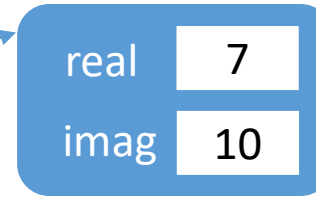
```
{  
    real += x.real; imag += x.imag;  
    return (*this);  
}
```

```
};
```

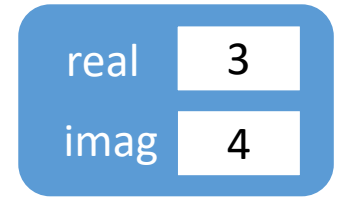


this

z

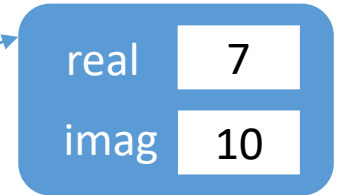


x
y



p

x



Output

(3 , 4)

Return by value

(1 , 2)

(7 , 10)

(4 , 6)

Return its pointer by value

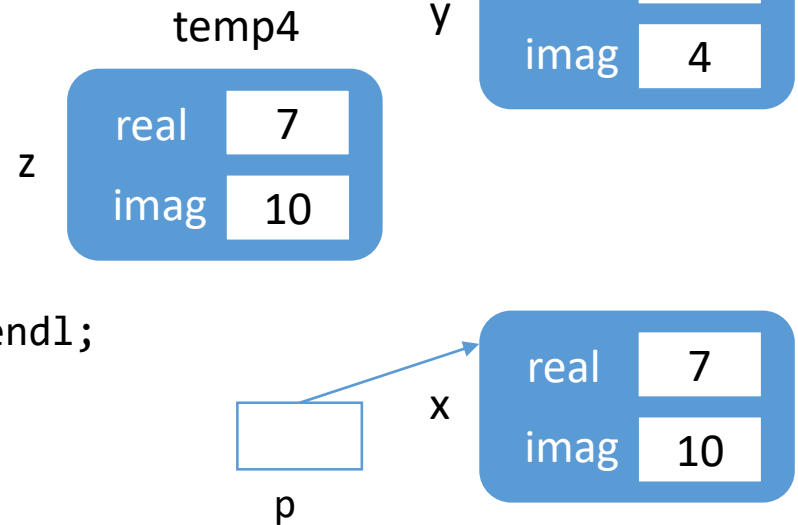
(7 , 10)

Return by reference

(1 , 2)

```
#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;
```

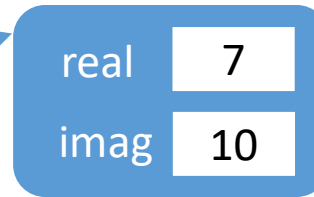
```
            return (*this);
```

```
        }
```

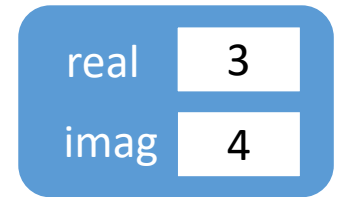
```
};
```



z

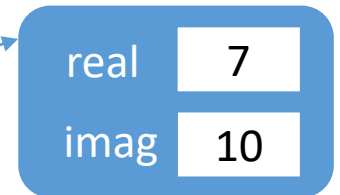


x
y



p

x



Output

```
( 3 , 4 )
```

```
Return by value
```

```
( 1 , 2 )
```

```
( 7 , 10 )
```

```
( 4 , 6 )
```

```
Return its pointer by value
```

```
( 7 , 10 )
```

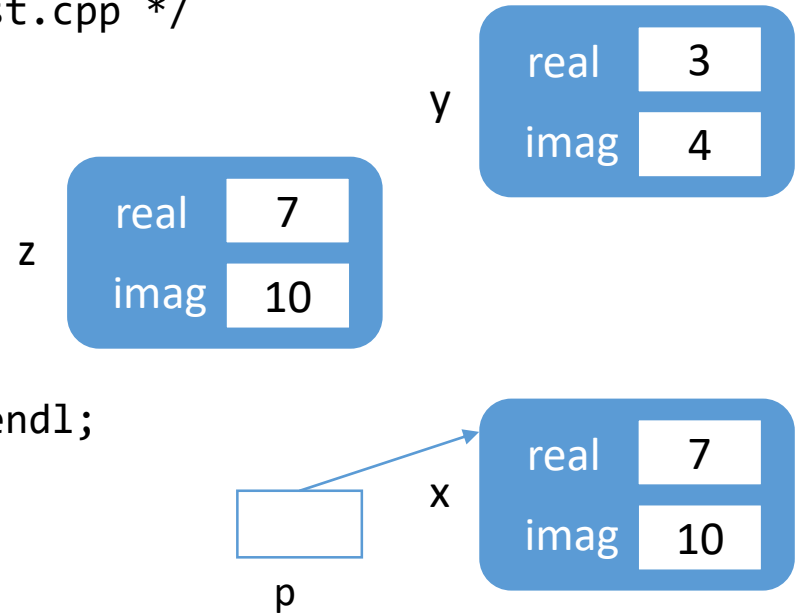
```
Return by reference
```

```
( 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; }

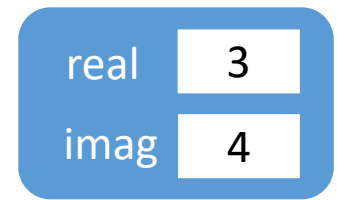
```



z



x
y



```

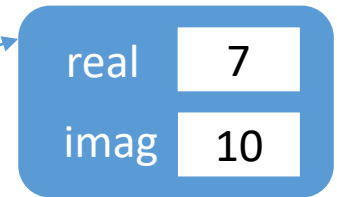
Complex add1(const Complex& x) // Return by value
{
    real += x.real; imag += x.imag;
    return (*this);
}

```



p

x



```

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

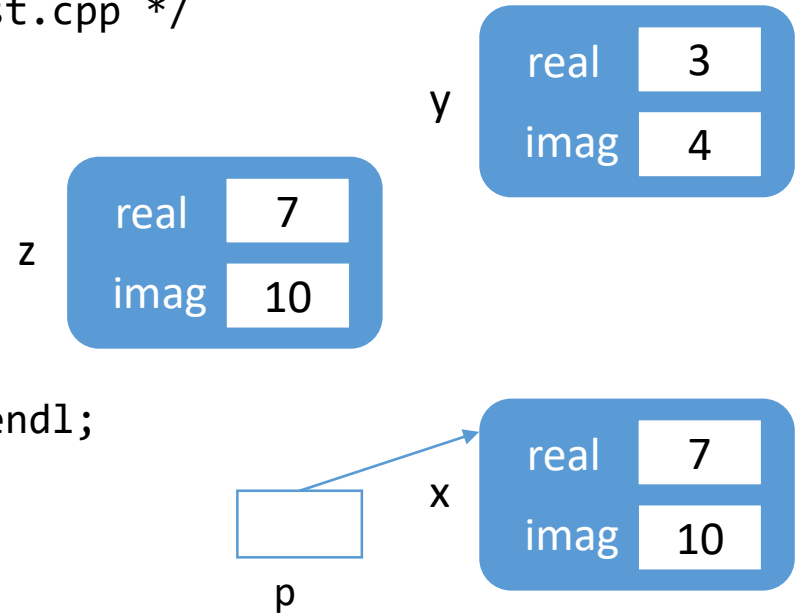
(1 , 2)

(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 )
```