Examples and Exercises

Prof. Sai-Keung Wong

TA:xyz

Let’s do some exercises

Encapsulation

Question

```
class A {
public:
        A() { … }
        A(int x, int y): a(x), b(y){ }
        void add() { a +=y; }
        int a;
        int b;
};
```

```
void main () {
  A c;
  c.a = 10;
  c.b = 20;

  c.add( );
}
//any error?
```

Answer

```
class A {
public:
        A() { … }
        A(int x, int y): a(x), b(y){ }
        void add() { a+=b;}
        int a;
        int b;
};
```

```
void main () {
  A c;
  c.a = 10;
  c.b = 20;

  c.add( );
}
//NO error
```

Question

Change the data representation: a, b are changed to a[2]

```
class A {
public:
        A() { … }
        A(int x, int y) {
                a[0] = x;
                a[1] = y;
        }
        void add() { x += y; }
        int a[2];        //
};
```

```
A c;
c.a = 10;
c.b = 20;

c.add( );

//errors? Why?
```

Answer

Change the data representation: a, b are changed to a[2]

```
class A {
public:
        A() { … }
        A(int x, int y) {
                a[0] = x;
                a[1] = y;
        }
        void add() { a[0] += a[1];}
        int a[2];        //
};
```

```
A c;
c.a[0] = 10;
c.a[1] = 20;

c.add( );
```

Question
Hide the data representation from the client.

```
class A {
public:
        A() { … }
        A(int x, int y) {
                a[0] = x;
                a[1] = y;
        }
        void add() { a[0] += a[1];}
private:
        int a[2];
};
```

```
A c;
c.a[0] = 10;
c.a[1] = 20;



// any errors?
```

Answer
Hide the data representation from the client.

```
class A {
public:
        A() { … }
        A(int x, int y) {
                a[0] = x;
                a[1] = y;
        }
        void add() { a[0] += a[1];}
private:
        int a[2];
};
```

```
//A c;
//c.a[0] = 10;
//c.a[1] = 20;



A c(10, 20);
c.add( );
```

Question

Hide the data representation from the client.

```
#define NUM 1000
class A {
        public:
        A() : n(0) {  }
        void push(int v) {
                a[n++] = v;
        }
        void add() { ……}
private:
        int a[NUM];
        int n;
};
```

```
A *c;
c.push(10);
c.push(20);
c.add( );



// any errors?
```

Answer

Hide the data representation from the client.

```
#define NUM 1000
class A {
        public:
        A() : n(0) {  }
        void push(int v) {
                a[n++] = v;
        }
        void add() { ……}
private:
        int a[NUM];
        int n;
};
```

```
A *c = new A;
c->push(10);
c->push(20);
c->add( );
```

Inheritance

Derived classes: Class access specifiers

```
class A {
        public: int x;
        protected: int y;
        private: int z;
};
```

```
class B : public A {
        …
};
```

What are the access restrictions of the data members and function for **different** class access specifiers?

Class access specifiers: public, protected, private

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
        protected: int y;
        private: int z;
};
```

```
class B : public A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

public, protected, private

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
        protected: int y;
        private: int z;
};
```

```
class B : protected A {
        …
};
```

public, protected, private

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
        protected: int y;
        private: int z;
};
```

```
class B : private A {
        …
};
```

Example One: public, protected, private

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : public A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Question

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : public A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Any error?

Answer

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : public A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Any error? no

Question

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : protected A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Any error?

Answer

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : protected A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Question

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : private A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Any error?

Answer

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        public: int x;
};
```

```
class B : private A {
        …
};
```

```
A p;
B q;
p.x = 10;
q.x = 12;
```

Question

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        private: int x;
        public: int y;
};
```

```
class B : protected A {

};
```

```
class C : public B {
        void foo( ) {
                x = 10;
                y = 10;
        }
};
```

Any error?

Answer

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        private: int x;
        public: int y;
};
```

```
class B : protected A {

};
```

```
class C : public B {
        void foo( ) {
                x = 10;
                y = 10;
        }
};
```

Question

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        private: int x;
        public: int y;
};
```

```
class B : private A {

};
```

```
class C : public B {
        void foo( ) {
                x = 10;
                y = 10;
        }
};
```

Any error?

Answer

Base class		public	protected	private
Derived class	public	public	protected	private
	protected	protected	protected	private
	private	private	private	private

```
class A {
        private: int x;
        public: int y;
};
```

```
class B : private A {

};
```

```
class C : public B {
        void foo( ) {
                x = 10;
                y = 10;
        }
};
```

Polymorphism

Must have at least one virtual function in base class.

Polymorphism

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

```
A *p;
A x;
B y;
C z;

p = &x;
p = &y;
p = &z;
```

Polymorphism

Draw a graph about the relation between the three classes

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```


```
A *p;
A x;
B y;
C z;

p = &x;
p = &y;
p = &z;
```

Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

Base class

A

↓

B

↓

C

```
A *p;
A x;
B y;
C z;

p = &x;
p = &y;
p = &z;
```

Which is no error?

```
B *q;
A x;
B y;
C z;

q = &x;
q = &y;
q = &z;
```

Answer

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

A

↓

B

↓

C

```
B *q;
A x;
B y;
C z;

q = &x;
q = &y;
q = &z;
```

✗

ok

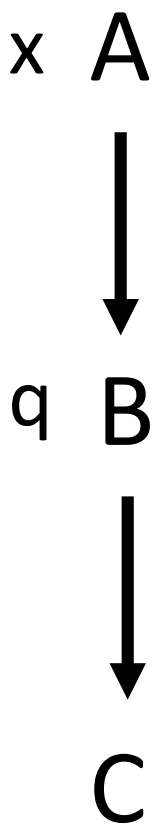
ok

Cast from base to derived class. Need dynamic_cast or static_cast.

Answer

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

Which is no error?




```
B *q;
A x;
B y;
C z;

q = &x;
q = &y;
q = &z;
```

✗ ok ok

Cast from base to derived class. Need dynamic_cast or static_cast.

Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

A

↓

B

↓

C

Any error?

```
B *q; C *w;
A x;
B y;
C z;

q = &x;
w = &y;
w = &z;
```

Answer

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

A

↓

B

↓

C

```
B *q; C *w;
A x;
B y;
C z;

q = &x;
w = &y;
w = &z; ok
```

Cast from base to derived class. Need dynamic_cast or static_cast.

Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

A

↓

B

↓

C

Any error?

```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
p = q;
```

Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```

No error

```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
p = q;
```

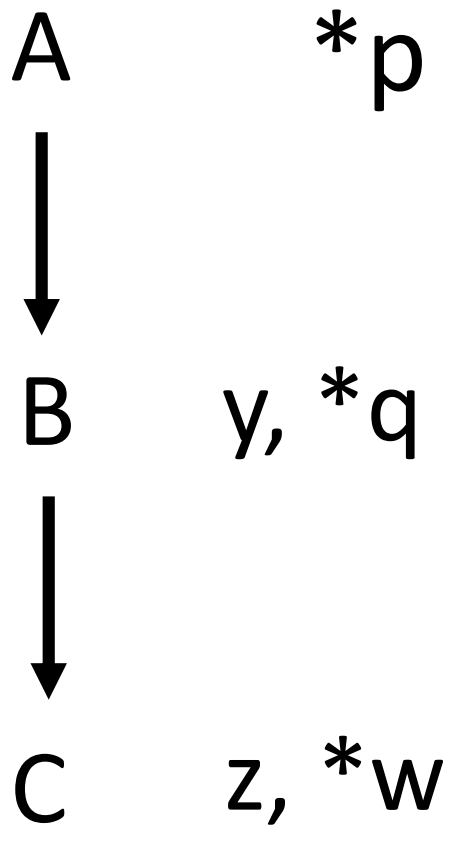
Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```


```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
w = &z;
p = q;
q = w;
```

Question. Draw a diagram first!

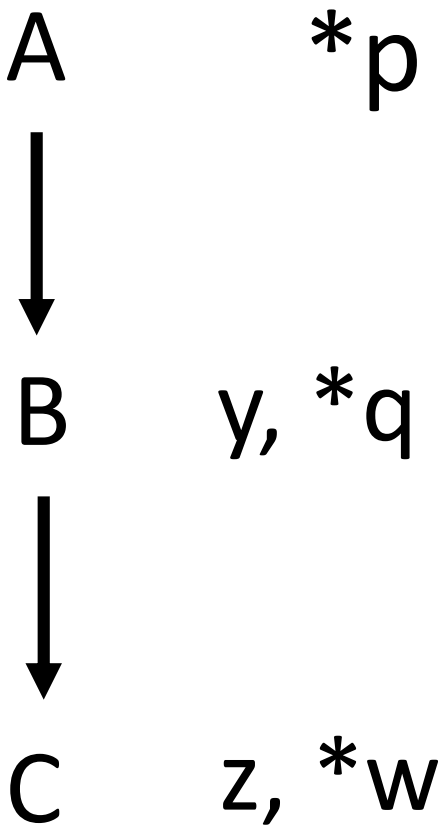
```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```




```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
w = &z;
p = q;
q = w;
```

Question

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```



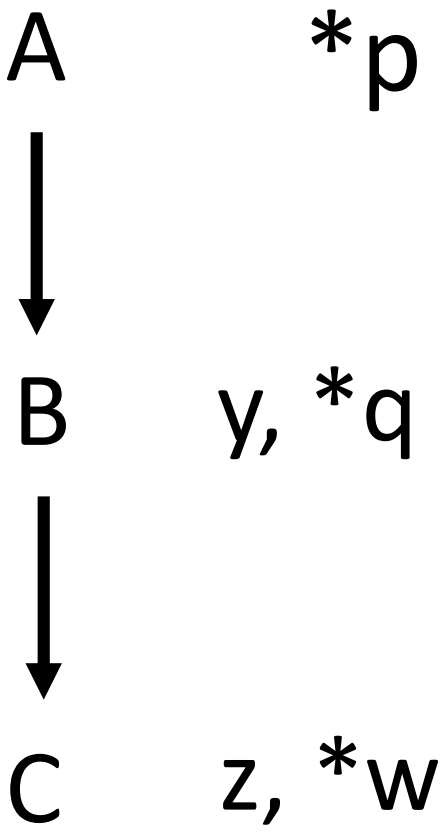

```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
w = q;
```

What about that?

Question

Cast from base to derived class.
Need dynamic_cast or static_cast.

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```



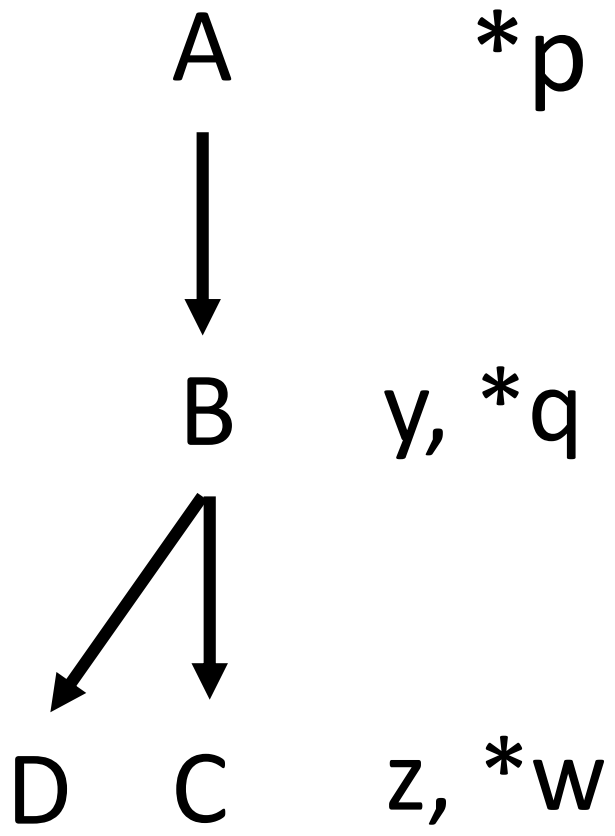

```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
w = q;
```

Any error?

Answer

Consider that there's a class D which is derived from B.

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```



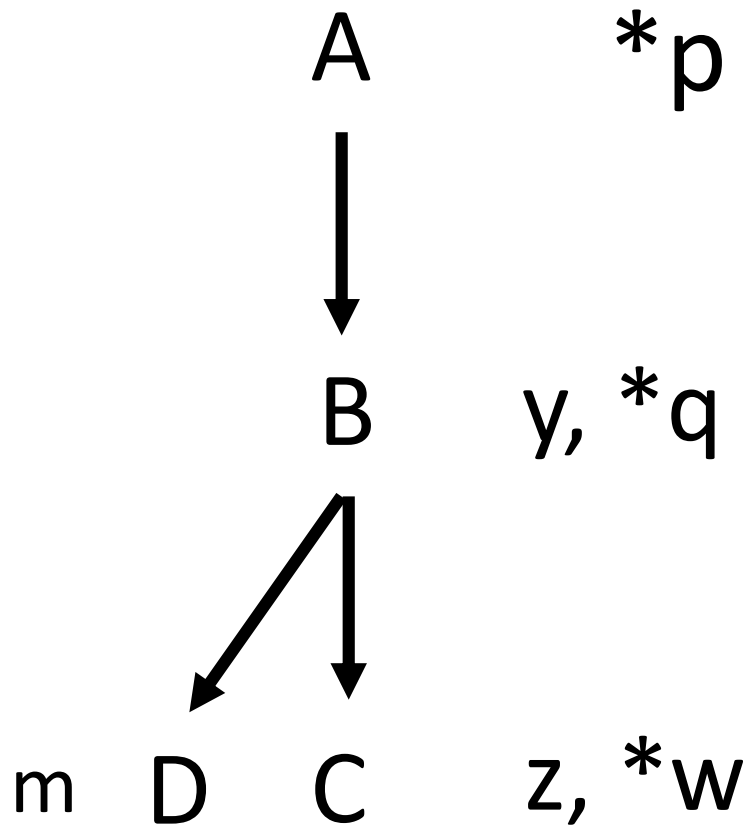

```
A *p;
A x;
B y, *q;
C z, *w;

q = &y;
w = q;
```

Why not allowed?

Answer

Consider that there's a class D which is derived from B.

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```



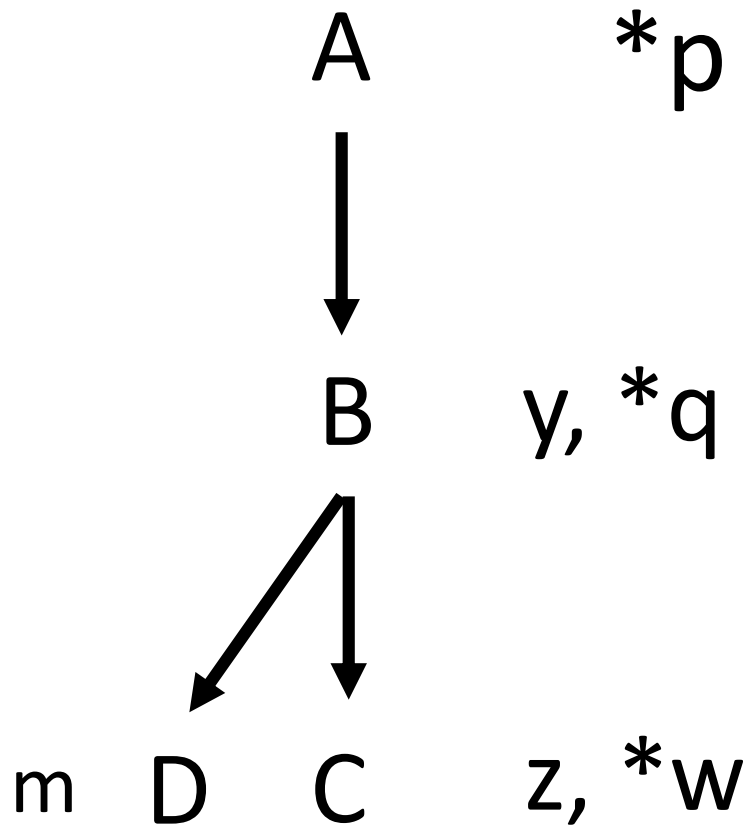

```
A *p;
A x;
B y, *q;
C z, *w;
D m;

q = &m;
w = q;
```

An object of C is NOT an object of D.

An object of C is NOT an object of D.

What should we do then?

```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```



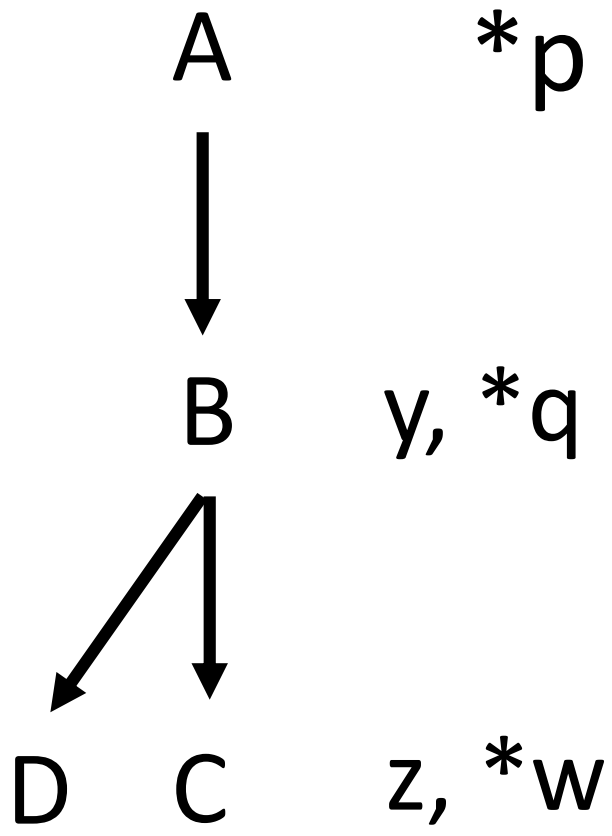

```
A *p;
A x;
B y, *q;
C z, *w;
D m;

q = &m;
w = q;
```

An object of C is NOT an object of D.

Answer

Use dynamic_cast

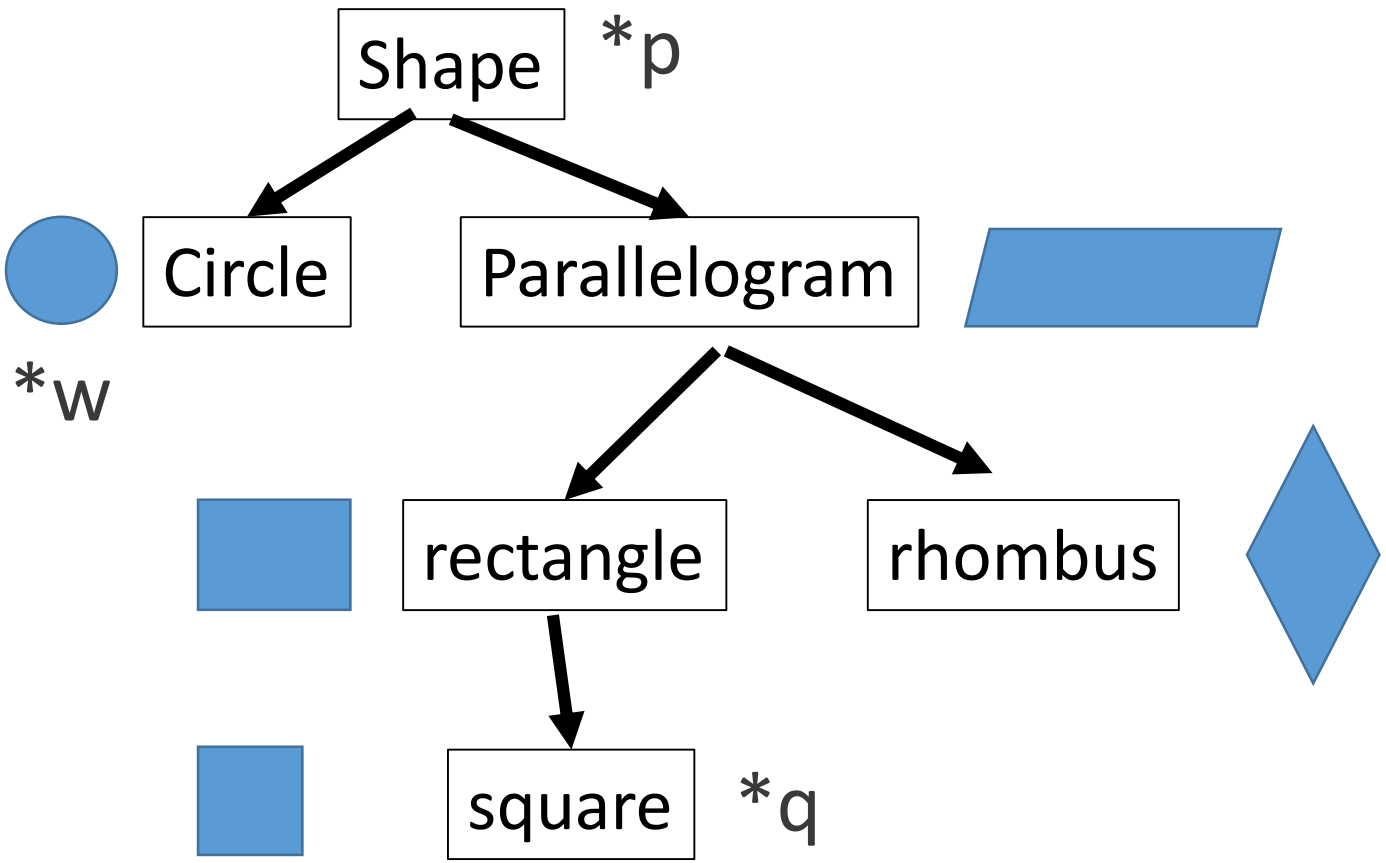
```
class A {
…
};

class B :public A {
…
};

class C: public B {
…
};
```




```
A *p;
A x;
B y, *q;
C z, *w;
D m;

q = &m;
w = dynamic_cast<C*>(q);
```

Check whether
w is NULL or not.

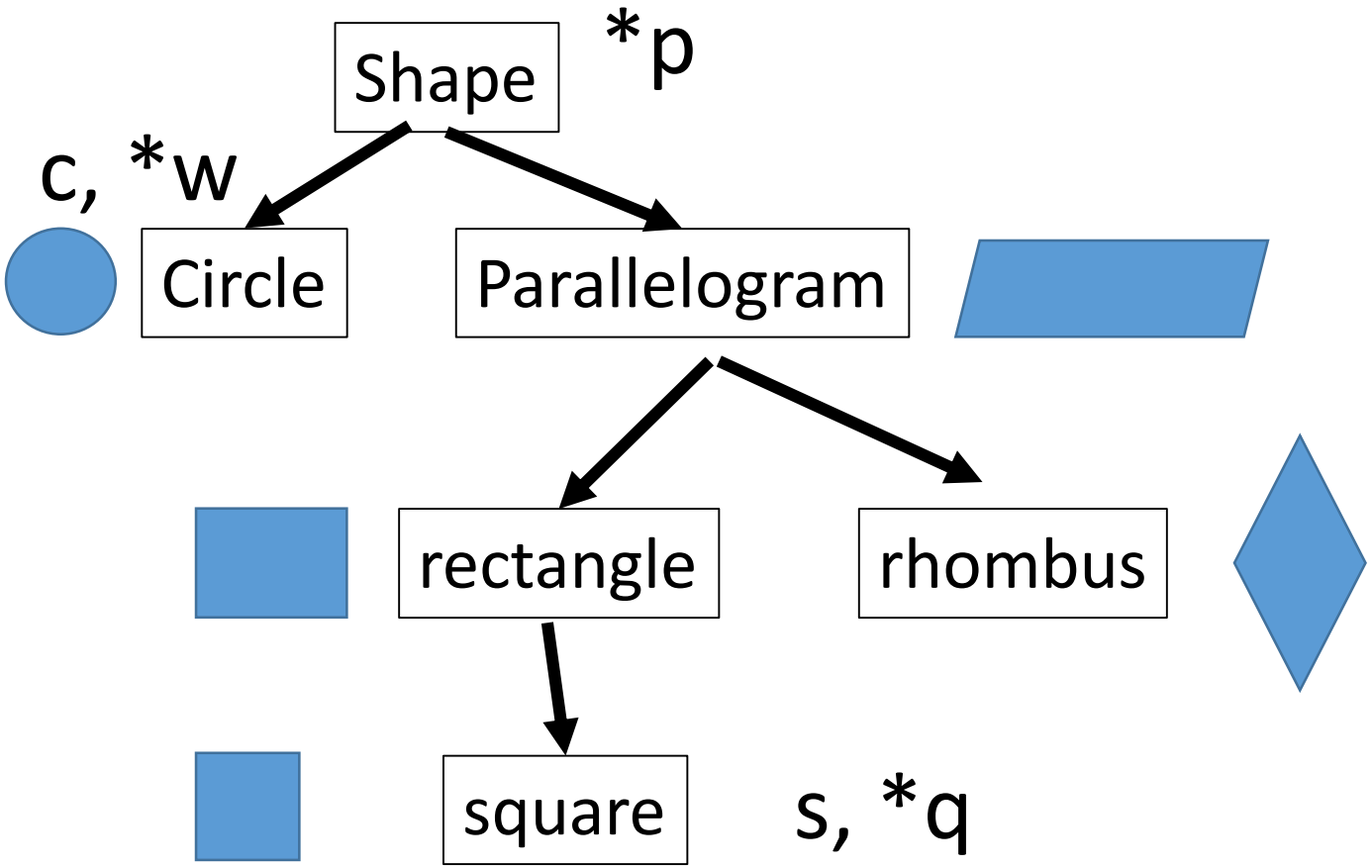
Question about Polymorphism




```
Shape *p;
Circle c, *w;
Square s, *q;

q = &s;
p = q;
w = p;
```

Is there any error?
Explain briefly.
Specify the line(s) which has(have) an error.

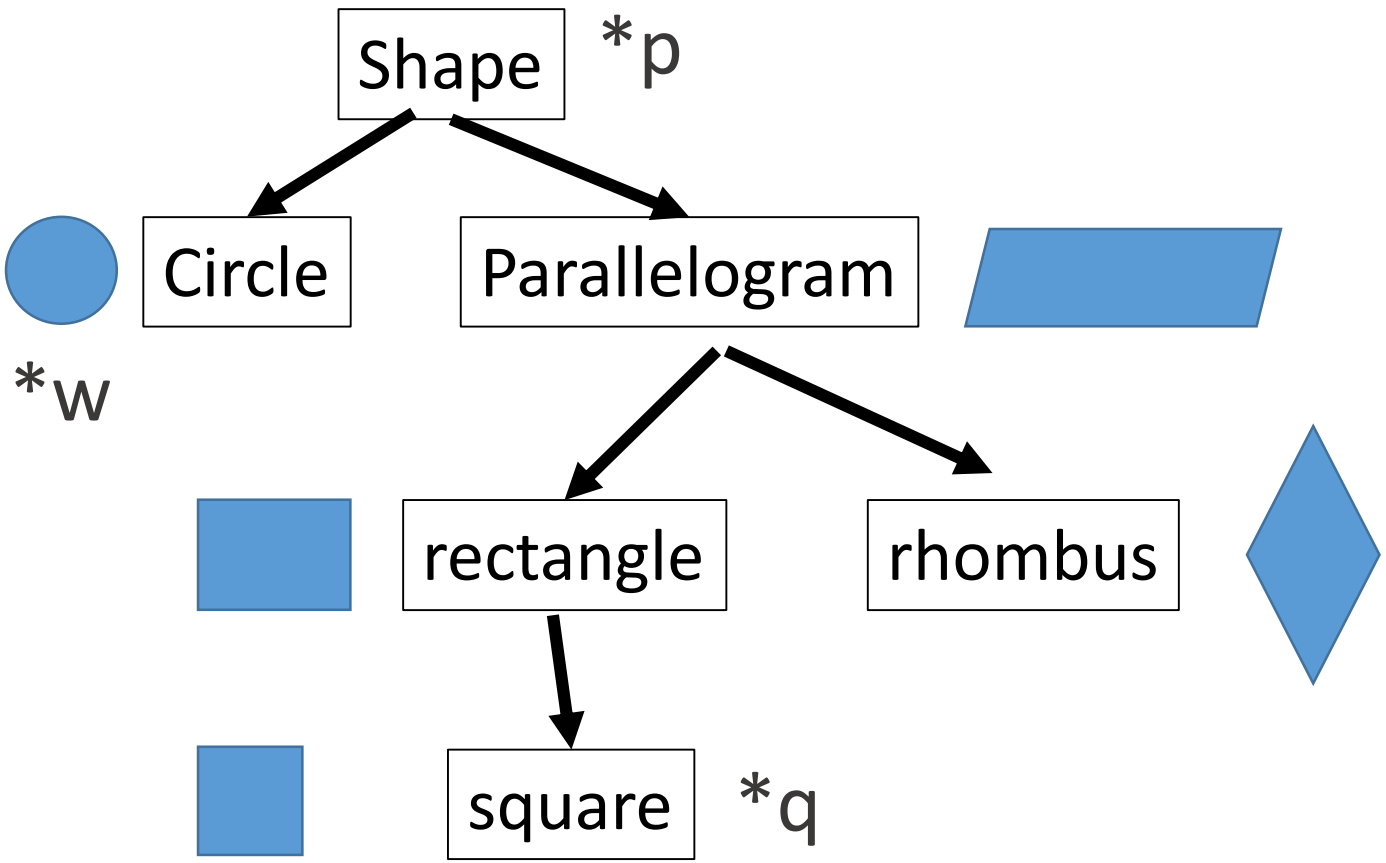
Draw a better diagram first!




```
Shape *p;
Circle c, *w;
Square s, *q;

q = &s;
p = q;
w = p;
```

Is there any error?
Explain briefly.
Specify the line(s) which has(have) an error.

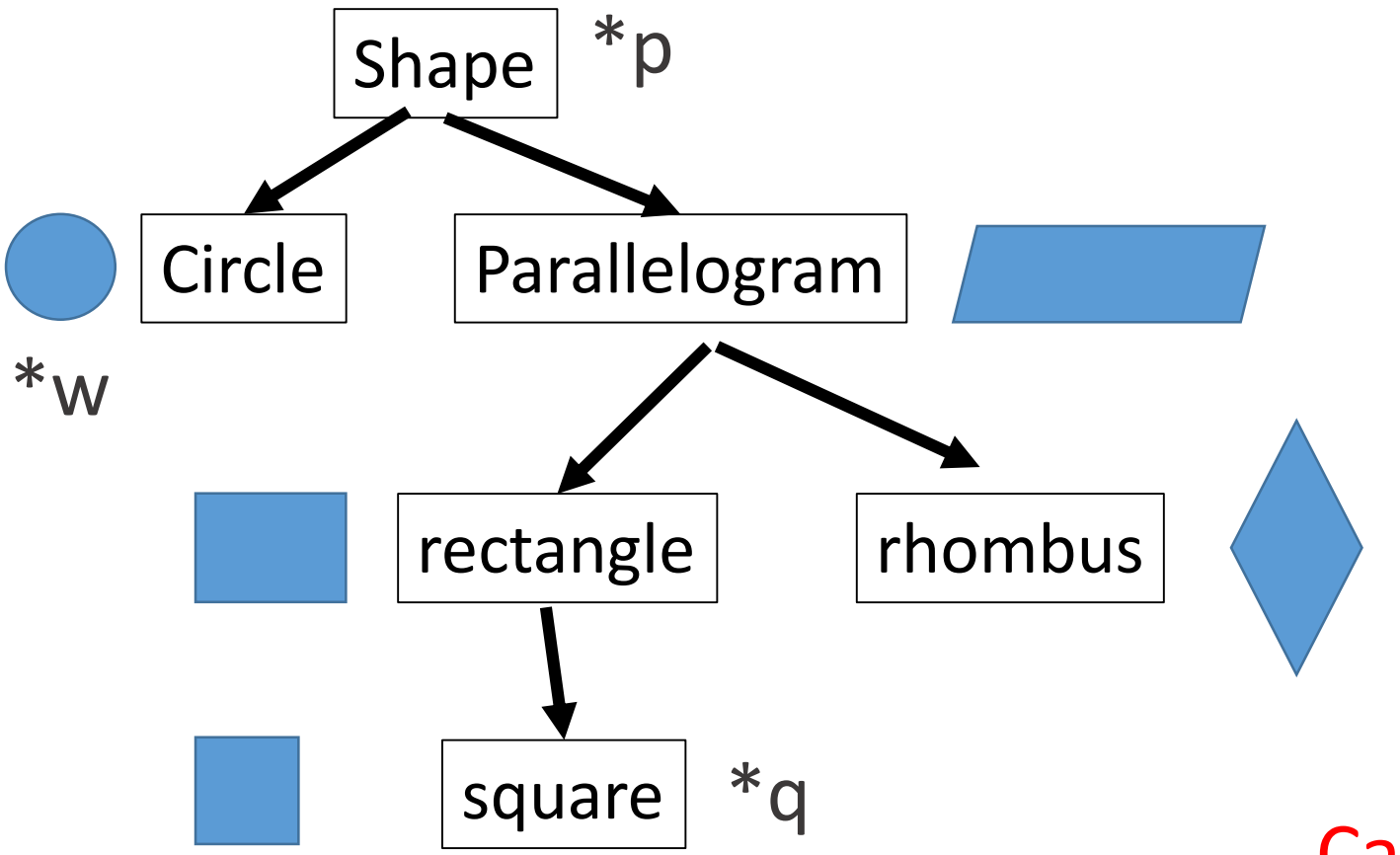
Question




```
Shape *p;
Circle c, *w;
Square s, *q;

q = &s;
p = q;
w = p;
```

Any errors?
And how to solve each error?

Answer




```
Shape *p;
Circle c, *w;
Square s, *q;

q = &s;
p = q;
w = dynamic_cast<Circle*>(p);

w = p; // error
```

Cast from base to derived class.

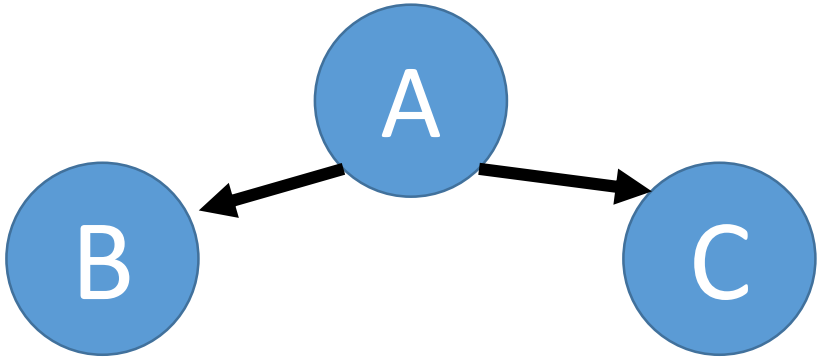
Need dynamic_cast or static_cast.

static_cast vs dynamic_cast

Question

```
class A {
};

class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};

class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

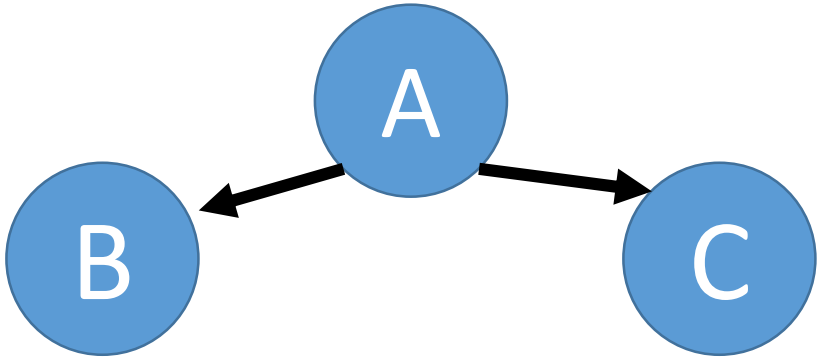
```
A *x;
B *y = new B;
C *z;

x = y;
z = static_cast<C*>( x);
z->foo();

//can we use static_cast?
```



Answer

```
class A {
};

class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};

class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

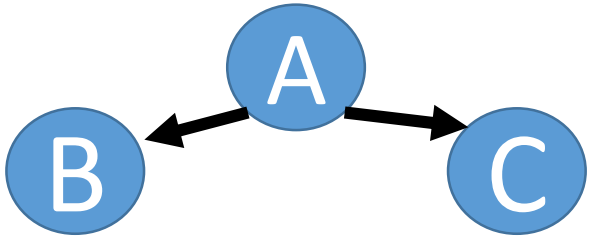
```
A *x;
B *y = new B;
C *z;

x = y;
z = static_cast<C*>( x);
z->foo();

//No. static_cast
//no run-time check
```



Question

```
class A {
};

class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};

class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

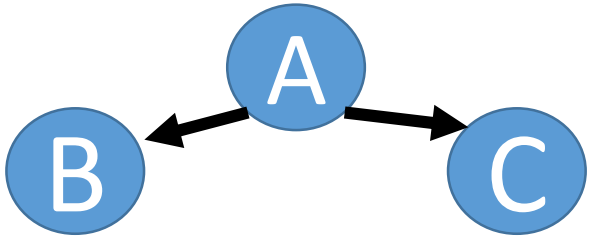
```
A *x;
B *y = new B;
C *z;

x = y;
z = dynamic_cast<C*>( x);
z->foo();

//dynamic_cast
//run-time check

//Any error?
```



Answer

```
class A {
};

class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};

class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

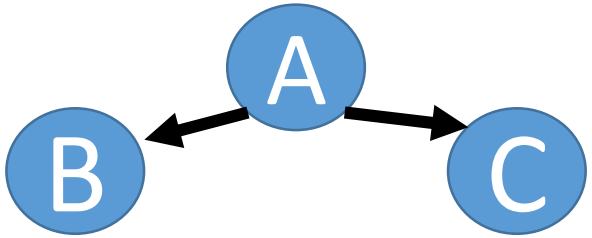
```
A *x;
B *y = new B;
C *z;

x = y;
z = dynamic_cast<C*>( x);
if (z) z->foo();

// check if z != 0 before
// we use it
```



Any other error?

```
class A {
};

class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};

class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

```
A *x;
B *y = new B;
C *z;

x = y;
z = dynamic_cast<C*>( x);
if (z) z->foo();

// check if z != 0 before
// we use it
```

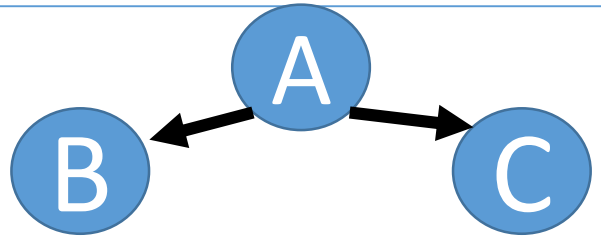


Answer 1

```
class A {
public: virtual void foo()  { }
};
class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};
class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

```
A *x;
B *y = new B;
C *z;

x = y;
z = dynamic_cast<C*>( x);
if (z) z->foo();
```

To use dynamic_cast, the base class must be a class with virtual member function(s).

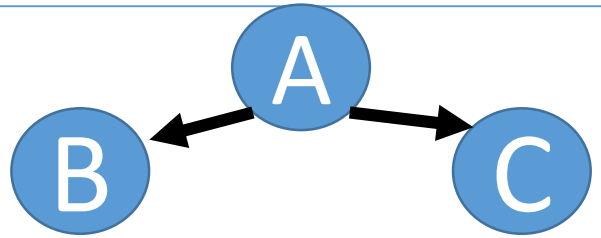


Answer 2

```
class A {
public: virtual void k( ) { } // good!
};
class B: public A {
public:
   B() : b(2){ }
   void foo() { cout << "b:" << b << endl;}
   int b;
};
class C: public A {
public:
   C() : c(3){ }
   void foo() { cout << "c:" << c << endl;}
   int c;
};
```

```
A *x;
B *y = new B;
C *z;

x = y;
z = dynamic_cast<C*>( x);
if (z) z->foo();
```

To use dynamic_cast, the base class must be a class with virtual member function(s).



Virtual Functions

Question

```
class A {
 public: A( ) { }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { }
   void foo( ) { cout << “B” << endl; }
};
```

```
A x;
B y;
x.foo( );
y.foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { }
   void foo( ) { cout << “B” << endl; }
};
```

```
A x;
B y;
x.foo( );
y.foo( );
```

What are the output?

A

B

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A x;
B y, *z;
x.foo( );
y.foo( );
```

What are the output?

Incorrect Answer

```
class A {
 public: A( ) { cout << "AC" << endl; }
   void foo( ) { cout << "A" << endl; }
};

class B : public A {
 public: B( ) { cout << "BC" << endl; }
   void foo( ) { cout << "B" << endl; }
};
```

```
A x;
B y, *z;
x.foo( );
y.foo( );
```

What are the output?

AC ?
BC ?
A ?
C ?

Something is incorrect.

Answer

```
class A {
 public: A( ) { cout << "AC" << endl; }
   void foo( ) { cout << "A" << endl; }
};

class B : public A {
 public: B( ) { cout << "BC" << endl; }
   void foo( ) { cout << "B" << endl; }
};
```

```
A x;
B y, *z;
x.foo( );
y.foo( );
```

What are the output?

AC
AC
BC
A
C

To create an object of B y here, the default constructor of A and B are invoked.

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A x;
B *z;
z = new B; // instantiate
z->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A x;
B *z;
z = new B; // instantiate
z->foo( );
```

What are the output?

AC

AC

BC

B

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
B *z;
x = new B;
x->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
B *z;
x = new B;
x->foo( );
```

What are the output?

AC

BC

A

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   virtual void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
B *z;
x = new B;
x->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   virtual void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
B *z;
x = new B;
x->foo( );
```

What are the output?

AC

BC

B

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   virtual void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B;
x->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   virtual void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B;
x->foo( );
```

What are the output?

AC

BC

A

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   A(int x) { cout << “ACC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   B(int x) { cout << “BCC” << endl; }
   virtual void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   A(int x) { cout << “ACC” << endl; }
   void foo( ) { cout << “A” << endl; }
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   B(int x) { cout << “BCC” << endl; }
   virtual void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
```

What are the output?

AC

BCC

A

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) = 0; // declare
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B;
x->foo( );
```

What are the output?

Answer

```
class A {
 public: A( ) { cout << “AC” << endl; }
   void foo( ) = 0; // declare
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   void foo( ) { cout << “B” << endl; }
};
```

```
A *x;
x = new B;
x->foo( );
```

What are the output?

No output.

Compilation error.

Answer
Fix the problem

```
class A {
 public: A( ) { cout << “AC” << endl; }

 virtual void foo( ) = 0;

};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
  void foo( ) { cout << “A” << endl; }
};
```

```
A *x;
x = new B;
x->foo( );
```

What are the output?

virtual void foo( ) **= 0** ;

Question

```
class A {
 public: A( ) { cout << “AC” << endl; }
   A(int x) { cout << “ACC” << endl;}
   virtual void foo( ) = 0; // declare
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   B(int x) : A(x) { cout << “BCC” << endl;}
   void foo( ) { cout << “B” << endl; }
};
```

```
A y;
A *x;
x = new B;
x->foo( );
x = &y;
x->foo( );
```

What are the output?

Answer

```
class A {
  public: A( ) { cout << “AC” << endl; }
   A(int x) { cout << “ACC” << endl;}
   virtual void foo( ) = 0; // declare
};

class B : public A {
 public: B( ) { cout << “BC” << endl; }
   B(int x) : A(x) { cout << “BCC” << endl;}
   void foo( ) { cout << “B” << endl; }
};
```

```
A y;   // A is an abstract class
A *x;
x = new B;
x->foo( );
x = &y;
x->foo( );
```

No output. Compilation error.

A is an abstract class. Cannot create an object.

Question

```
class A {
A( ) { cout << "AC" << endl; }
   A(int x) { cout << "ACC" << endl;}
   virtual void foo( ) = 0; // declare
};
class B : public A {
B( ) { cout << "BC" << endl; }
   B(int x) : A(x) { cout << "BCC" << endl;}
   void foo( ) { cout << "B" << endl; }
};
class C: public A {
C() { cout << "CC" << endl; }
   void foo( ) { cout << "C" << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
C y;
x = &y;
x->foo( );
```

What are the output?

Answer

```
class A {
A( ) { cout << "AC" << endl; }
   A(int x) { cout << "ACC" << endl;}
   virtual void foo( ) = 0; // declare
};
class B : public A {
B( ) { cout << "BC" << endl; }
   B(int x) : A(x) { cout << "BCC" << endl;}
   void foo( ) { cout << "B" << endl; }
};
class C: public A {
C() { cout << "CC" << endl; }
   void foo( ) { cout << "C" << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
C y;
x = &y;
x->foo( );
```

What are the output?

Compilation error.

No output

Answer

```
class A {
A( ) { cout << "AC" << endl; }
    A(int x) { cout << "ACC" << endl;}
    virtual void foo( ) = 0; // declare
};
class B : public A {
B( ) { cout << "BC" << endl; }
    B(int x) : A(x) { cout << "BCC" << endl;}
    void foo( ) { cout << "B" << endl; }
};
class C: public A {
C() { cout << "CC" << endl; }
    void foo( ) { cout << "C" << endl; }
};
```

Compilation error.
No output.
A's constructors are private to B.

A's default constructor is private to C.

Question

```
class A {
public: A( ) { cout << "AC" << endl; }
    A(int x) { cout << "ACC" << endl;}
    virtual void foo( ) = 0; // declare
};
class B : public A {
    public: B( ) { cout << "BC" << endl; }
    B(int x) : A(x) { cout << "BCC" << endl;}
    void foo( ) { cout << "B" << endl; }
};
class C: public A {
    public: C() { cout << "CC" << endl; }
    void foo( ) { cout << "C" << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
C y;
x = &y;
x->foo( );
```

What are the output?

Answer

```
class A {
public: A( ) { cout << "AC" << endl; }
    A(int x) { cout << "ACC" << endl;}
    virtual void foo( ) = 0; // declare
};
class B : public A {
    public: B( ) { cout << "BC" << endl; }
    B(int x) : A(x) { cout << "BCC" << endl;}
    void foo( ) { cout << "B" << endl; }
};
class C: public A {
    public: C() { cout << "CC" << endl; }
    void foo( ) { cout << "C" << endl; }
};
```

```
A *x;
x = new B(10);
x->foo( );
C y;
x = &y;
x->foo( );
```

What are the output?

ACC

BCC

B

AC

CC

C

Doing exercises not only improve the body but also have fun