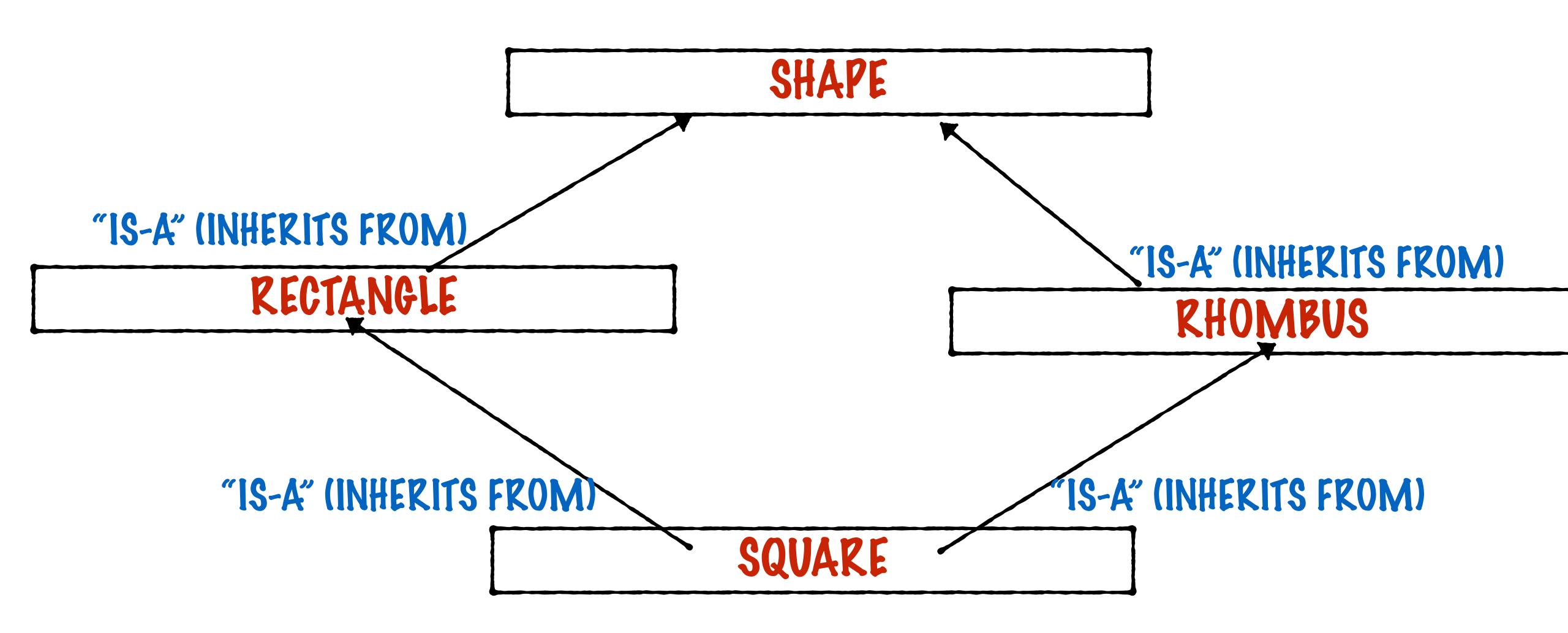
EXAMPLE 56: 'VIRTUAL' INHERITANCE, AND SIMPLIFYING A DIAMOND HIERARCHY

WE JUST LEARNT ABOUT MULTIPLE INHERITANCE AND A DIAMOND-SHAPED CLASS HIERARCHY

IT IS POSSIBLE FOR A DERIVED CLASS OBJECT TO END UP WITH 2 COPIES OF A BASE CLASS OBJECT INSIDE IT



FOR INSTANCE, IT IS POSSIBLE FOR A PERIVEP CLASS OBJECT TO

NOW THIS IS ALL PERFECTLY REASONABLE FROM A REAL-WORLD POINT-OF-VIEW.

"IS-A" (INHERITS FROM)

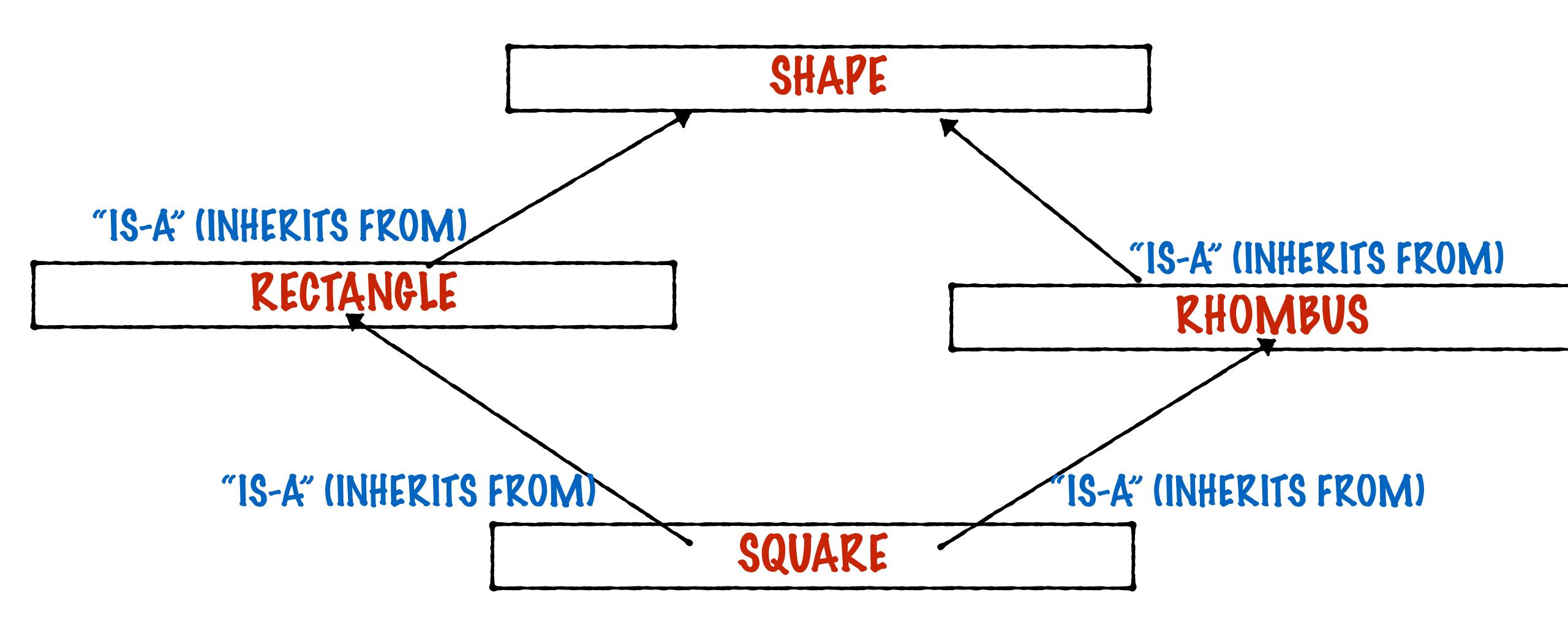
..BUT THIS PIAMOND-SHAPED CLASS PIAGRAM LEAVES FROM THE SQUARE OBJECT WITH 2 COPIES OF THE SHAPE OBJECT

"IS-A" (INHERITS FROM)

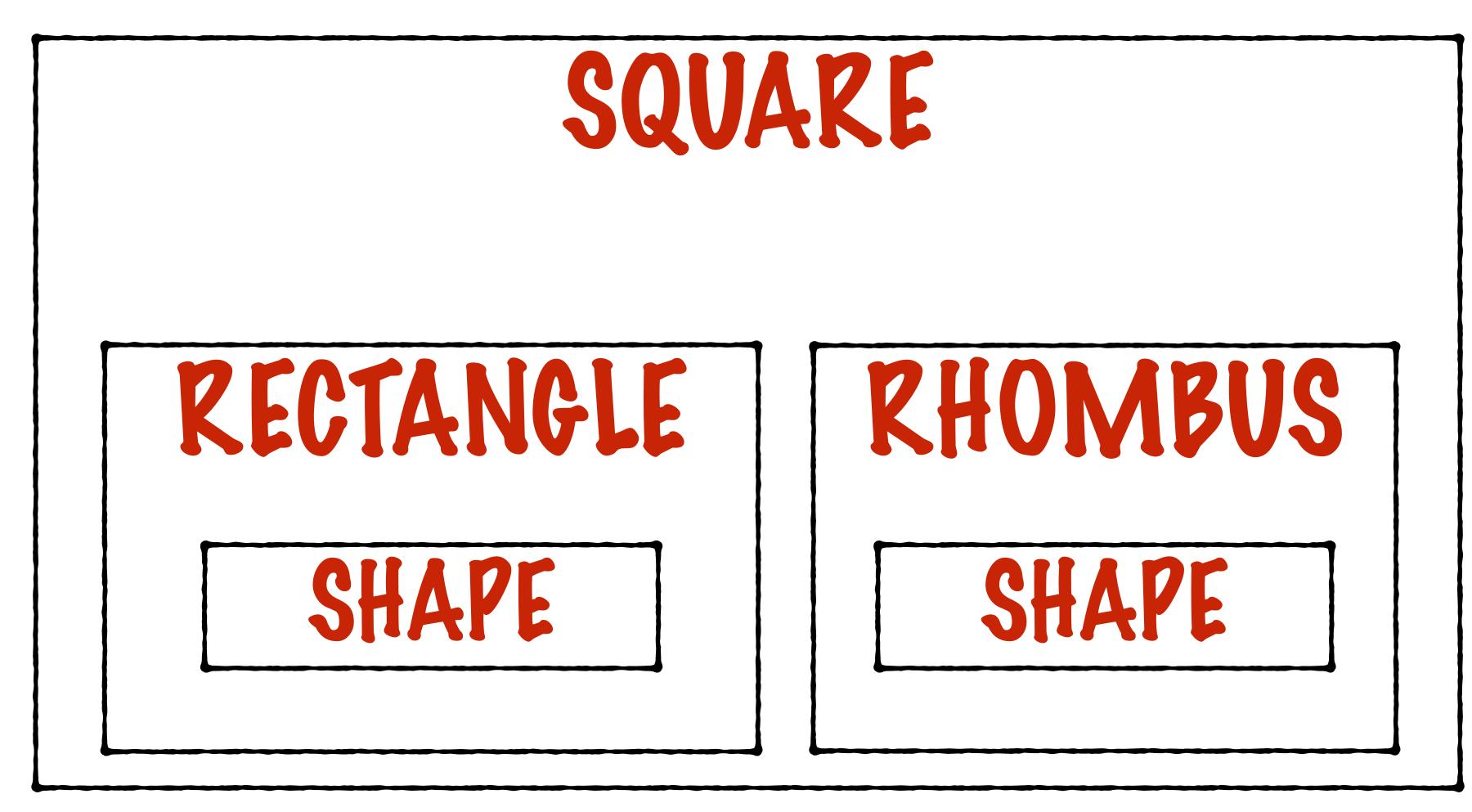
IS-A" (INHERITS FROM)

SQUARE

..BUT THIS DIAMOND-SHAPED CLASS DIAGRAM LEAVES THE SQUARE OBJECT WITH 2 COPIES OF THE SHAPE OBJECT



THIS DIAMOND-SHAPED CLASS DIAGRAM LEAVES THE SQUARE OBJECT WITH 2 COPIES OF THE SHAPE OBJECT



LAYOUT OF AN OBJECT OF CLASS SQUARE

IT TURNS OUT C++ HAS A SPECIAL WAY TO MAKE SURE THAT ONLY 1 INSTANCE OF THE SHAPE OBJECT SHOWS UP

ITS CALLED VIRTUAL INHERITANCE

REGULAR MULTIPLE INHERITANCE

SQUARE RHONBUS RECTANGLE SHAPE SHAPE

LAYOUT OF AN OBJECT OF CLASS SQUARE

SQUARE RECTANGLE RHOMBUS SHAPE

LAYOUT OF AN OBJECT OF CLASS SQUARE

VIRTUAL MULTIPLE INHERITANCE class Shape

SHAPE

"IS-A" (INHERITS FROM)

RECTANGLE

class Rhombus: virtual public Shape
"IS-A" (INHERITS FROM)

RHOMBUS

"IS-A" (INHERITS FROM)

IS-A" (INHERITS FROM)

SQUARE

class Square : public Rhombus, public Rectangle

VIRTUAL MULTIPLE INHERITANCE class Shape

SHAPE

NOW, NO NEED FOR THE SCOPE RESOLUTION OPERATOR WHILE ACCESSING THE SHAPE PORTIONS OF A SQUARE!

"IS-A" (INHERITS FROM)

IS-A" (INHERITS FROM)

SQUARE

class Square : public Rhombus, public Rectangle

```
class Shape
private:
public:
 string shapeType;
  Shape()
    shapeType = "Unknown";
    cout << "Inside the Shape constructor" << endl;</pre>
  ~Shape()
    cout << "Inside the Shape destructor" << endl;</pre>
};
```

NOW, NO NEED FOR THE SCOPE RESOLUTION OPERATOR WHILE ACCESSING THE SHAPE PORTIONS OF A SQUARE!

```
class Shape
private:
public:
 string shapeType;
 Shape()
    shapeType = "Unknown";
    cout << "Inside the Shape constructor" << endl;</pre>
 ~Shape()
    cout << "Inside the Shape destructor" << endl;</pre>
```

```
Square s1;
cout << s1.shapeType;
// this is only possible because of the
// from rectangle, rhombus to shape. Els
// s1.Rectangle::shapeType</pre>
```

NOW, NO NEED FOR THE SCOPE RESOLUTION OPERATOR WHILE ACCESSING THE SHAPE PORTIONS OF A SQUARE!

```
Square s1;
cout << s1.shapeType;

// this is only possible because of the virtual inheritance
// from rectangle, rhombus to shape. Else would have needed
// s1.Rectangle::shapeType</pre>
```

NOW, NO NEED FOR THE SCOPE RESOLUTION OPERATOR WHILE ACCESSING THE SHAPE PORTIONS OF A SQUARE!