EXAMPLE 61: UNDERSTAND HOW NAME HIDING WORKS (AND WHY IT IS A PROBLEM)

EXAMPLE 61: UNDERSTAND HOW NAME HIDING WORKS (AND WHY IT IS A PROBLEM)

BASE CLASS

```
class Shape
{
  void printNumber(int number)
  {
    cout << "I am a shape - printing number" << number << endl;
  }
};
```

DERIVED CLASS

```
class Rectangle : public Shape
{
  void printNumber()
  {
    cout << "I am a rectangle - printing number" <<  endl;
  }
};
```

EXAMPLE 60: UNDERSTAND HOW NAME HIDING WORKS (AND WHY IT IS A PROBLEM)

BASE CLASS

```
void printNumber(int number)
```

DERIVED CLASS

```
void printNumber()
```

SAME FUNCTION, BUT DIFFERENT SIGNATURES

DONT EVER DO THIS

BASE CLASS

DERIVED CLASS

SAME FUNCTION, BUT
DIFFERENT SIGNATURES

DONT EVER DO THIS

BECAUSE THE BASE CLASS NAME
GETS ENTIRELY "HIDDEN"

BASE CLASS

DERIVED CLASS

SAME FUNCTION, BUT DIFFERENT SIGNATURES

DONT EVER DO THIS

BECAUSE THE BASE CLASS NAME GETS ENTIRELY "HIDDEN"

```
Rectangle r;
r.printNumber();  // works OK
r.printNumber(10); // unexpected compile error!
```

```
Example61.cpp:53:5: error: too many arguments to function call, expected 0, have 1; did you mean 'Shape::printNumber'?
  r.printNumber(10); // unexpected compile error!
    ^~~~~~~~~~~
    Shape::printNumber
Example61.cpp:10:8: note: 'Shape::printNumber' declared here
  void printNumber(int number)
       ^
1 error generated.
```

ADDING THE SCOPE RESOLUTION OPERATOR WILL UNHIDE IT

```
Rectangle r;
r.printNumber();   // works OK
r.Shape::printNumber(10); // works OK now
```

BUT JUST NEVER "HIDE NAMES" LIKE THIS. ITS JUST WEIRD.