Using Objects

Matthew Mussomele

Objects So Far

We've worked with creating objects and defining their behaviors.

We haven't used them yet.

Objects are very useful for taking complicated ideas and processes and reducing them to some simpler concept.

Today we'll learn how to create objects you have defined and how to use them once they've been created.

Creating Objects

You can declare objects just like any other variable.

```
<class_name> my_object;
```

There is one major difference.

When objects are declared in this manner, their zero-argument constructor is called and the object is instantiated.

When objects are instantiated in this way, the variable used to store them is a reference.

Creating Objects

You can create objects in other ways to. See the following:

```
    <class_name> my_object(<parameters>);
```

This does the same thing as the method on the last slide, except it calls the constructor matching the passed parameters.

Once again, this returns a reference to the new object and stores it in the variable.

Creating Objects

There is one last way to create object instances.

```
<class_name> * my_variable = new <class_name>(<parameters>);
```

This method will create a new instance of the object using the corresponding constructor.

A pointer to the new object will be returned to the variable using this way.

Using Created Objects

Once an object is created, you can use them to achieve various things, depending on their definitions.

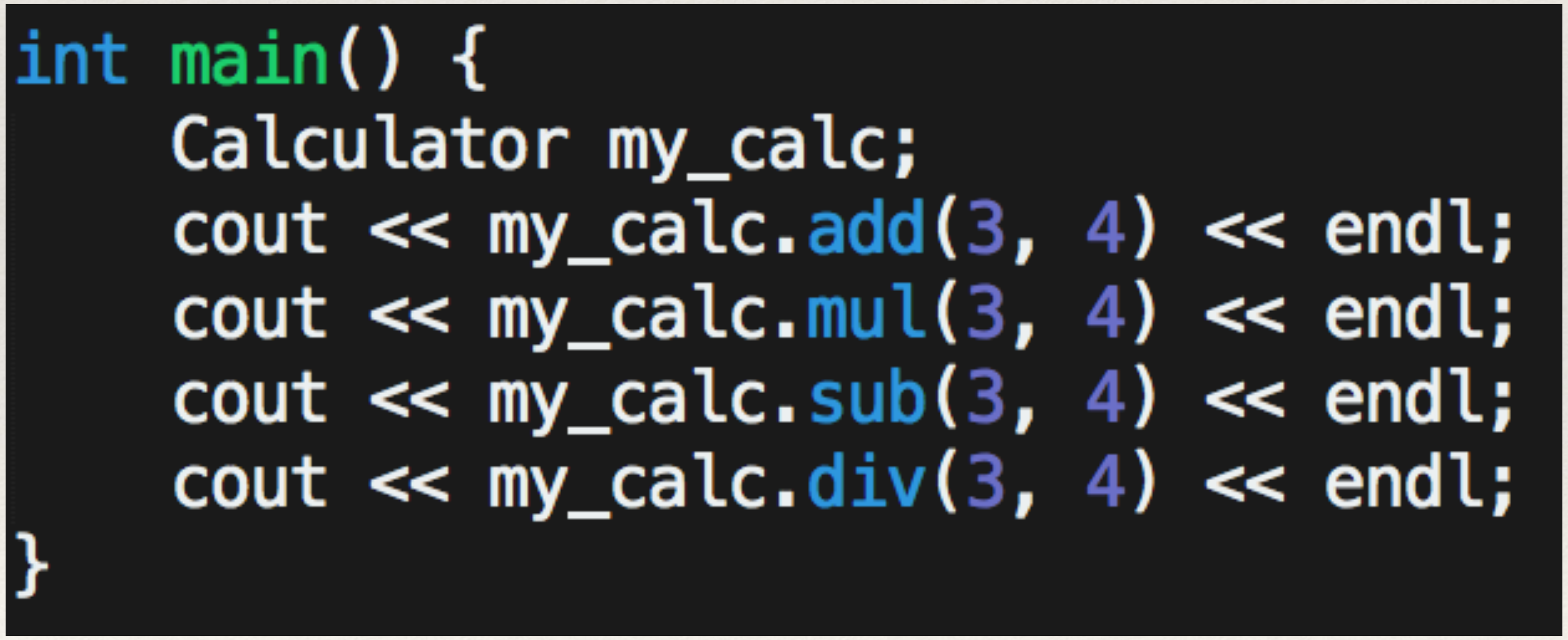
In order to do this, you can use the functions and variables that belong to the object.

We will use the calculator class you completed in the last task to exemplify this.

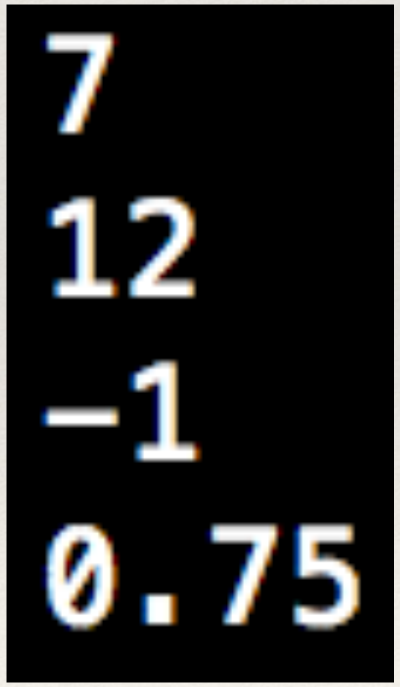
Using Object References

In the first two ways to create objects, we got references to the objects as a result.

Observe the code snippet below.



```
int main() {
    Calculator my_calc;
    cout << my_calc.add(3, 4) << endl;
    cout << my_calc.mul(3, 4) << endl;
    cout << my_calc.sub(3, 4) << endl;
    cout << my_calc.div(3, 4) << endl;
}
```



```
7
12
-1
0.75
```

Using Object References

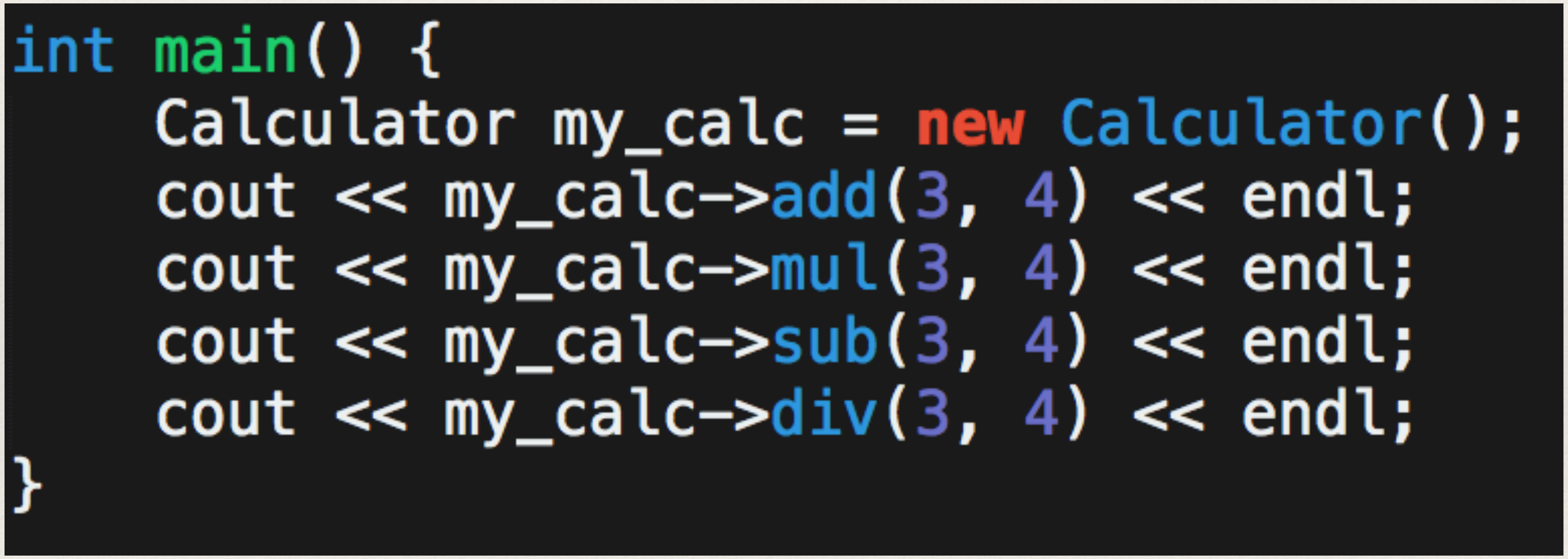
When we were defining objects, we used the :: operator to tell C++ that we were defining functions belonging to the given class.

Now we use the . operator to tell C++ we want to use the function belonging to a specific instance of an object.

If we have a pointer to an object, like with the second method of object creation, simply replace the . with a -> operator.

Using Object Pointers

The code below is equivalent to the snippet on the last two slides, except it uses a pointer instead.



```cpp
int main() {
    Calculator my_calc = new Calculator();
    cout << my_calc->add(3, 4) << endl;
    cout << my_calc->mul(3, 4) << endl;
    cout << my_calc->sub(3, 4) << endl;
    cout << my_calc->div(3, 4) << endl;
}
```