**Classes and Objects**

1. **Class definition**

To create a class (should be defined in header file)

We can define attributes and methods: (can access class attributes inside methods)

Text

Description automatically generated

//<-- notice this semicolon!

\*\* Unless we have a mostly empty class, it’s common to split function declarations from definitions. We declare methods inside the class (in a header), then define the methods outside the class (in a .cpp file of the same name).

To define a method outside a class, we add Classname:: before method name

Text

Description automatically generated with low confidence

🡪 We also need to include the header file in the .cpp file where we define the methods

1. **Creating Objects**

To instantiate an object, we can do like how we create a new instance of int, double:

<Classname> <InstanceName> ;

Then we can access/change the attributes and functions of the objects accordingly using “.”

1. **Access Control: Public and Private**

By default, everything in a class is private, meaning class members (attributes and methods) are limited to the scope of the class. This makes it easier to keep data from mistakenly being altered, and abstracts away all the nitty gritty details. If you try to access a private class member, you’ll get an error.

🡪 Therefore, sometimes if we need to get access to class members, we need to make the members public. Everything below the “public:” will be made public.

🡪 If we want to make something private under public, we can also use the private access modifier.

Text

Description automatically generated

1. **Constructor**

A constructor is a special kind of method that defines how an object of a class is created.

It has the same name as the class and no return type.

🡪 We can use parameters and a member initializer list to initialize attributes to values passed in

Text

Description automatically generated

**or** we can also write like this:

Text

Description automatically generated

1. **Destructors**

Object destructors will help prevent memory leaks and tidy up 🡪 Destroy objects

Like the constructor, it has the same name as the class and no return type, but is preceded by a ~ operator and takes no parameters.

Text

Description automatically generated

|  |
| --- |
| We generally don’t need to call the destructor; it will be called automatically in any of the scenarios:   * The object moves out of scope. * The object is explicitly deleted. * When the program ends. |