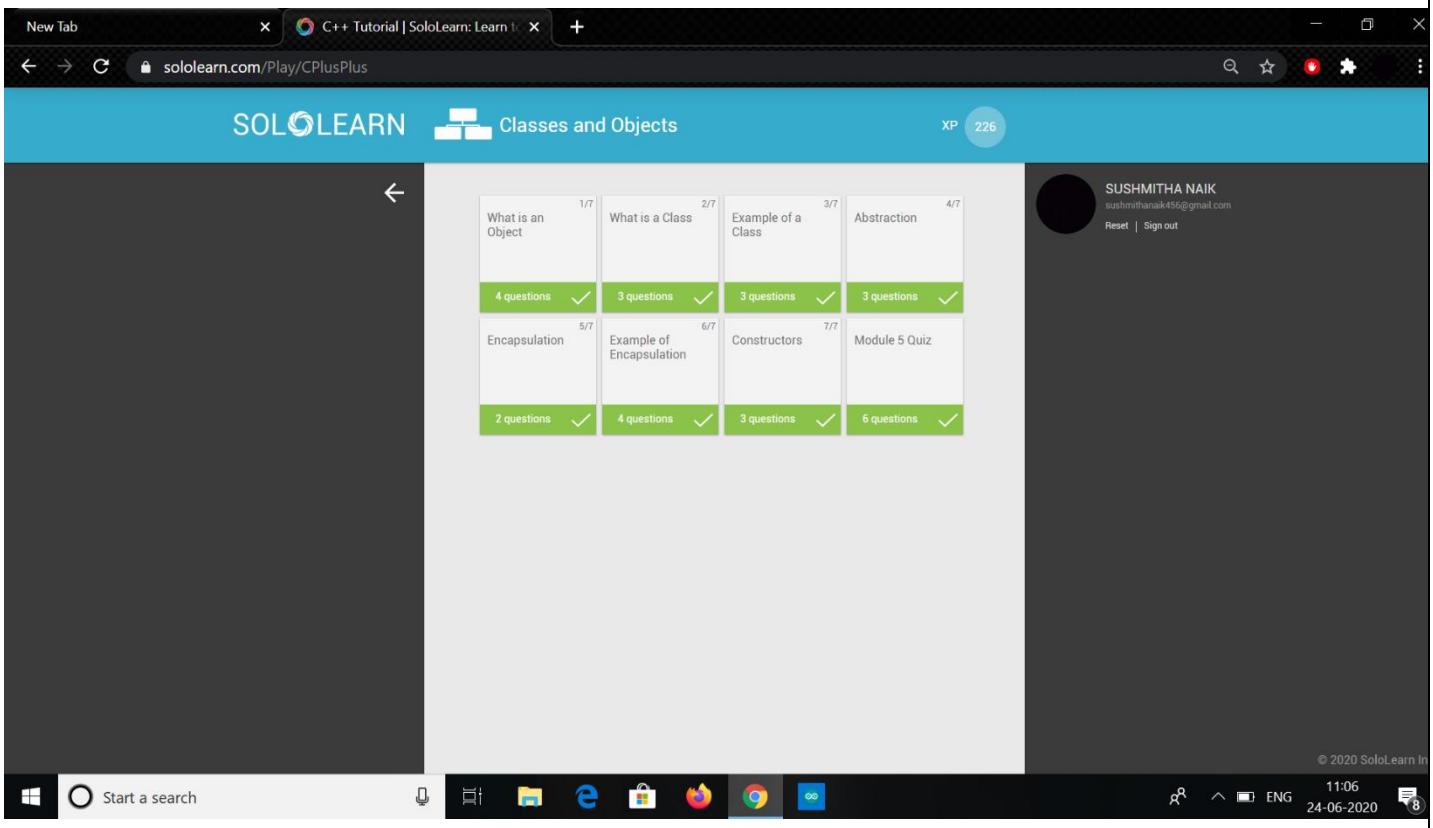
DAILY ASSESSMENT FORMAT

Date:	**24th June 2020**	**Name:**	**Sushmitha R Naik**
Course:	**C++ programming**	**USN:**	**4AL17EC090**
Topic:	**Classes and Objects**	**Semester & Section:**	**6th sem & B sec**
Github Repository:	**Sushmitha_naik**		

FORENOON SESSION DETAILS

Image of session



Objects

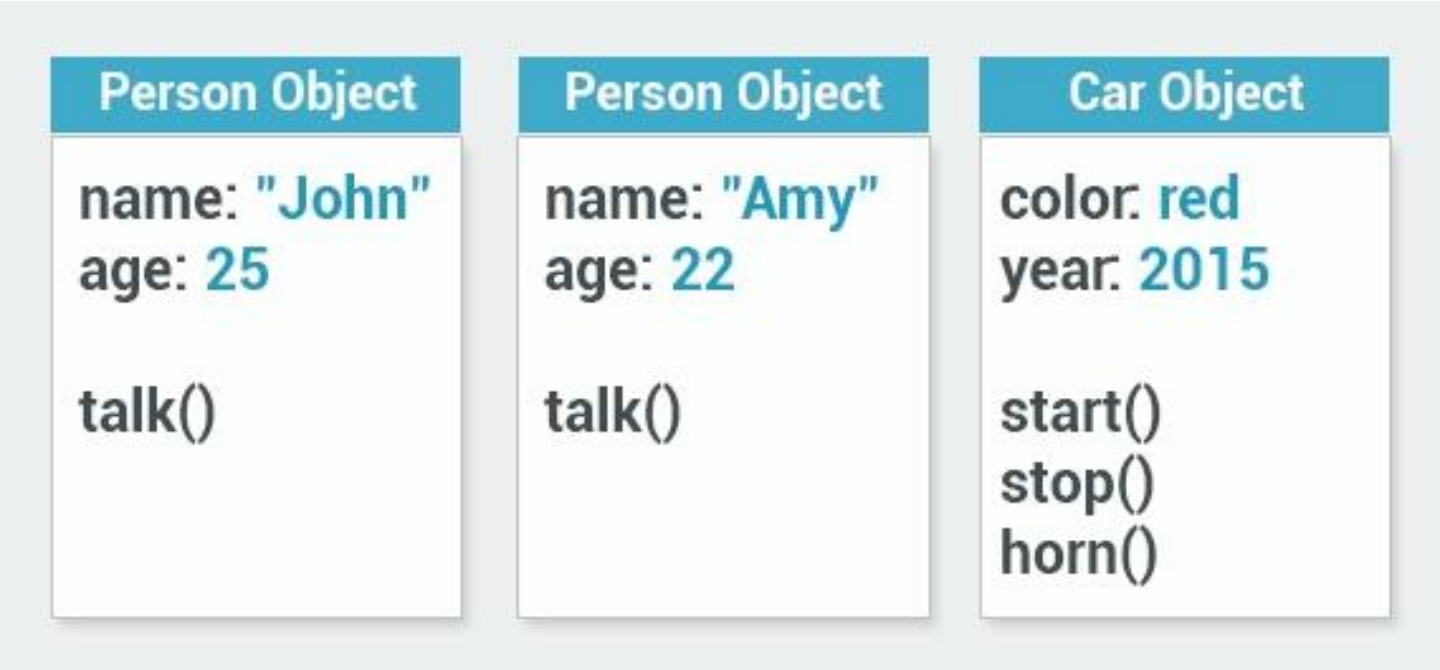
An object might contain other objects but they're still different objects.

Objects also have characteristics that are used to describe them. For example, a car can be red or blue, a mug can be full or empty, and so on. These characteristics are also called attributes. An attribute describes the current state of an object.

In programming, an object is self-contained, with its own identity. It is separate from other objects.

Each object has its own attributes, which describe its current state. Each exhibits its own behavior, which

demonstrates what they can do.



In computing, objects aren't always representative of physical items.
For example, a programming object can represent a date, a time, a bank account. A bank account is not tangible; you can't see it or touch it, but it's still a well-defined object - it has its own identity, attributes, and behavior.

Class:

Objects are created using classes, which are actually the focal point of OOP.

The class describes what the object will be, but is separate from the object itself.
In other words, a class can be described as an object's blueprint, description, or definition.
You can use the same class as a blueprint for creating multiple different objects. For example, in preparation to creating a new building, the architect creates a blueprint, which is used as a basis for actually building the structure. That same blueprint can be used to create multiple buildings.

Programming works in the same fashion. We first define a class, which becomes the blueprint for creating objects.
Each class has a name, and describes attributes and behavior.

In programming, the term type is used to refer to a class name: We're creating an object of a particular type.

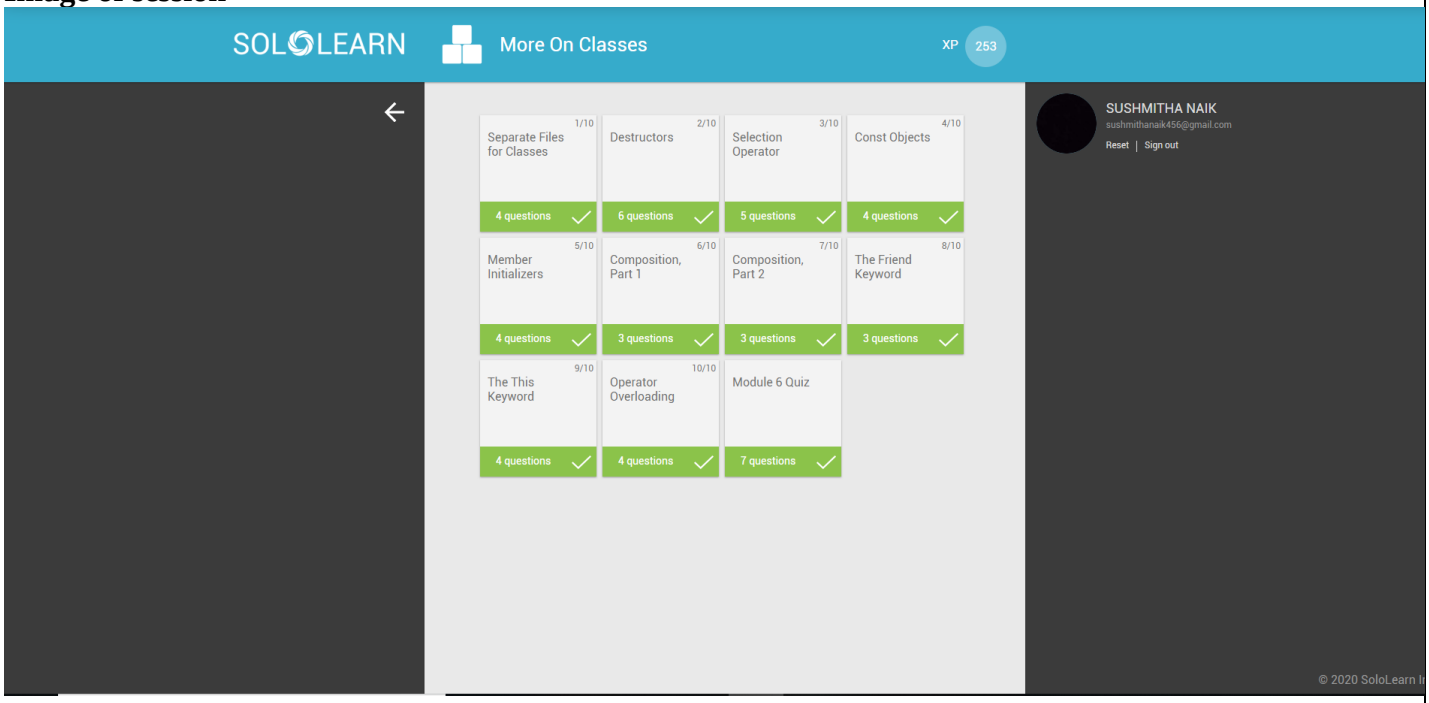
Encapsulation:

Part of the meaning of the word encapsulation is the idea of "surrounding" an entity, not just to keep what's inside together, but also to protect it.
In object orientation, encapsulation means more than simply combining attributes and behavior together within a class; it also means restricting access to the inner workings of that class.
The key principle here is that an object only reveals what the other application components require to effectively run the application. All else is kept out of view.

Date:	**24th June 2020**	**Name:**	**Sushmitha R Naik**
Course:	**C++ programming**	**USN:**	**4AL17EC090**
Topic:	**More on classes**	**Semester & Section:**	**6th sem & B sec**
Github Repository:	**Sushmitha_naik**		

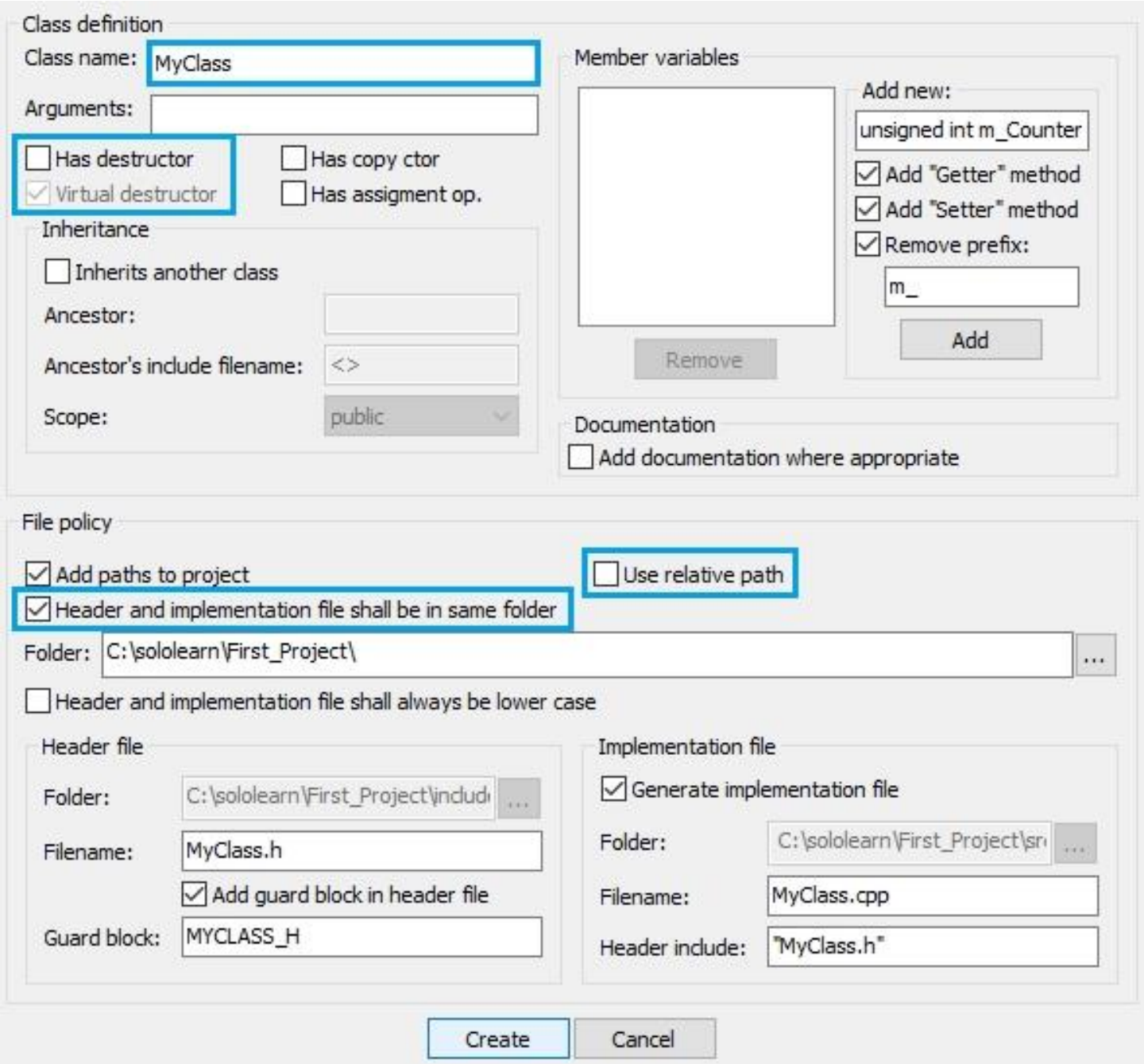
AFTERNOON SESSION DETAILS

Image of session



Creating a New Class

It is generally a good practice to define your new classes in separate files. This makes maintaining and reading the code easier.
To do this, use the following steps in CodeBlocks:
Click File->New->Class...
Give your new class a name, uncheck "Has destructor" and check "Header and implementation file shall be in same folder", then click the "Create"
button.



Note that two new files have been added to your project:

Destructors:

Remember constructors? They're special member functions that are automatically called when an object is created.
Destructors are special functions, as well. They're called when an object is destroyed or deleted.