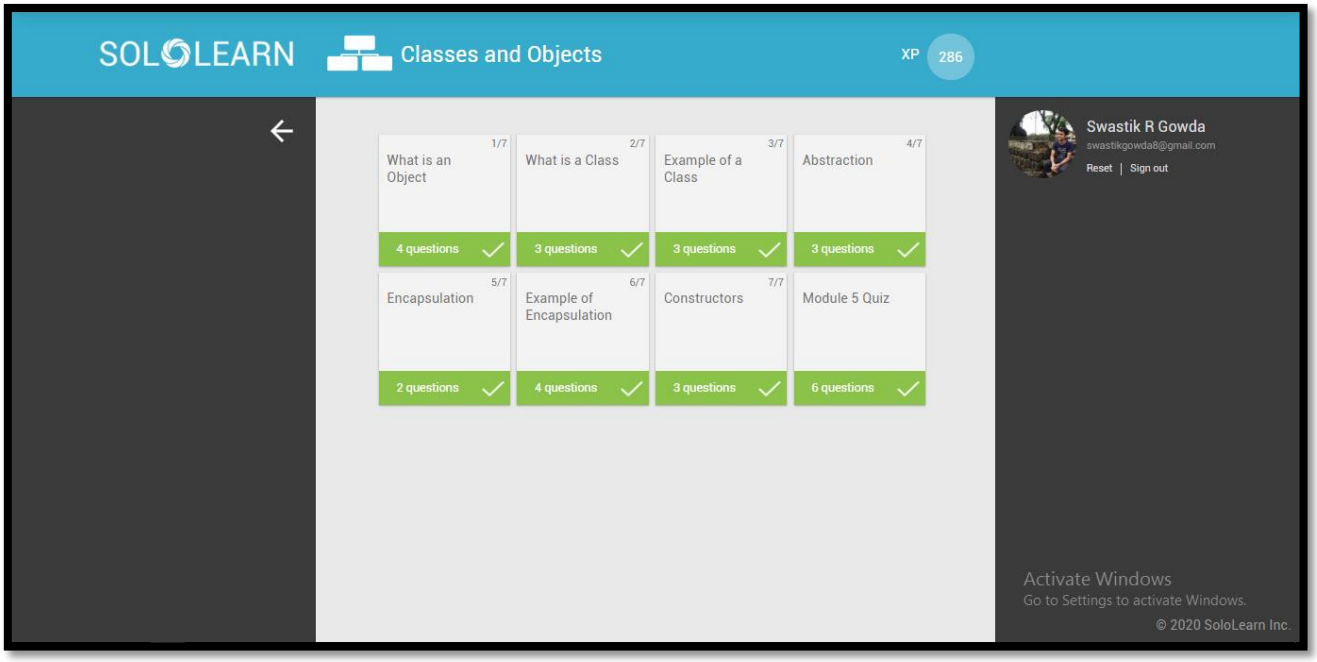
DAILY ASSESSMENT

Date:	**24-June-2020**	**Name:**	**Swastik R Gowda**
Course:	**Solo-Learn C++**	**USN:**	**4AL17EC091**
Topic:	**Module - 5 : Classes And Objects**	**Semester & Section:**	**6th Sem 'B' Sec**
Github Repository:	**swastik-gowda**		

FORENOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.

Object

- Object Oriented Programming is a programming style that is intended to make thinking about programming closer to thinking about the real world.
- In programming, objects are independent units, and each has its own identity, just as objects in the real world do.
- Objects also have characteristics that are used to describe them.
- An attribute describes the current state of an object.
- Objects can have multiple attributes

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.
- 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.

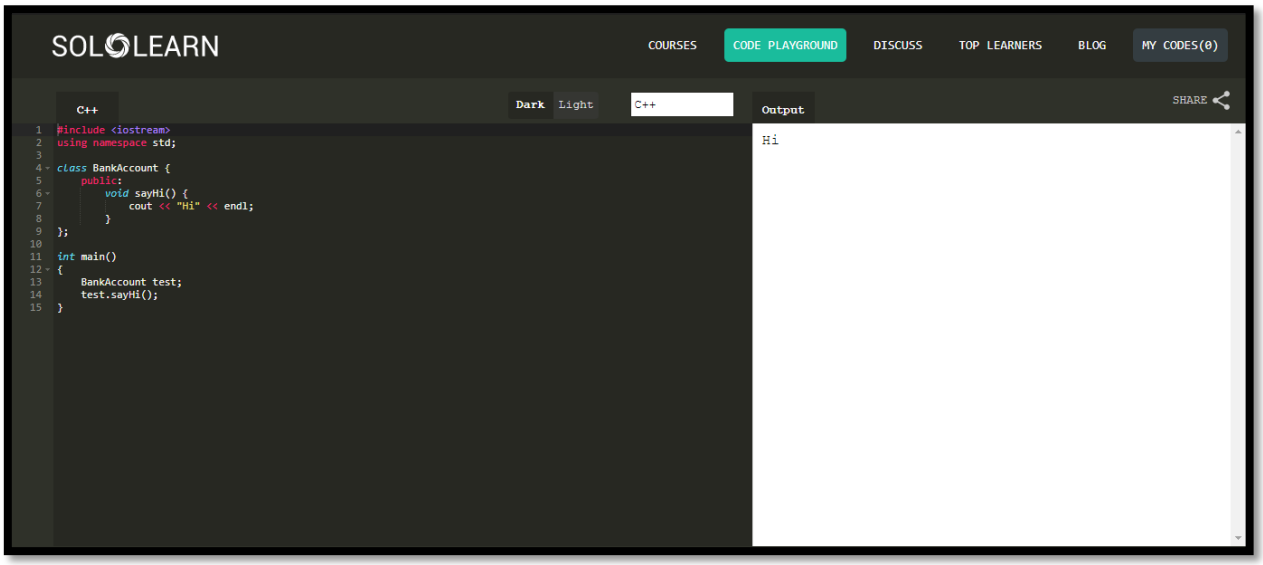
Methods

- Method is another term for a class' behavior. A method is basically a function that belongs to a class.
- Methods are similar to functions - they are blocks of code that are called, and they can also perform actions and return values.

name: BankAccount
attributes: accountNumber, balance, dateOpened
behavior: open(), close(), deposit()

The class specifies that each object should have the defined attributes and behavior. However, it doesn't specify what the actual data is; it only provides a definition.

Once we've written the class, we can move on to create objects that are based on that class.
Each object is called an instance of a class. The process of creating objects is called instantiation.



Abstraction

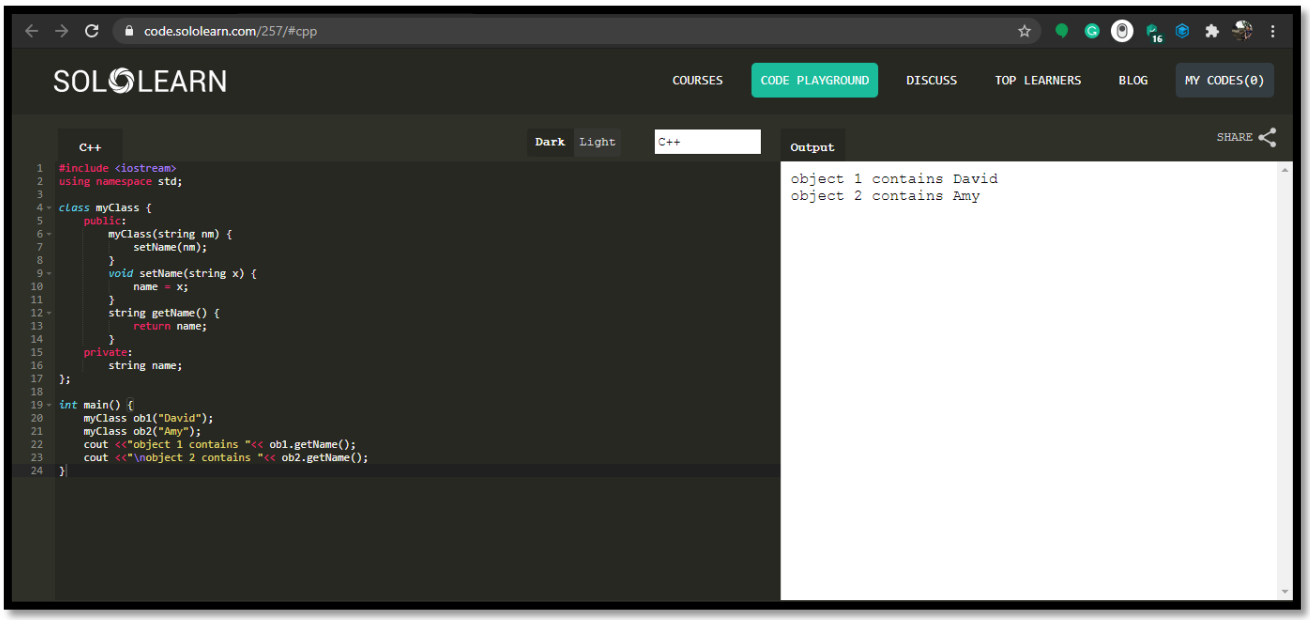
- Data abstraction is the concept of providing only essential information to the outside world.
- It's a process of representing essential features without including implementation details.
- Abstraction means, that we can have an idea or a concept that is completely separate from any specific instance.
- It is one of the fundamental building blocks of object oriented programming.
- Abstraction allows us to write a single bank account class, and then create different objects based on the class, for individual bank accounts, rather than creating a separate class for each bank account.

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.

Constructors

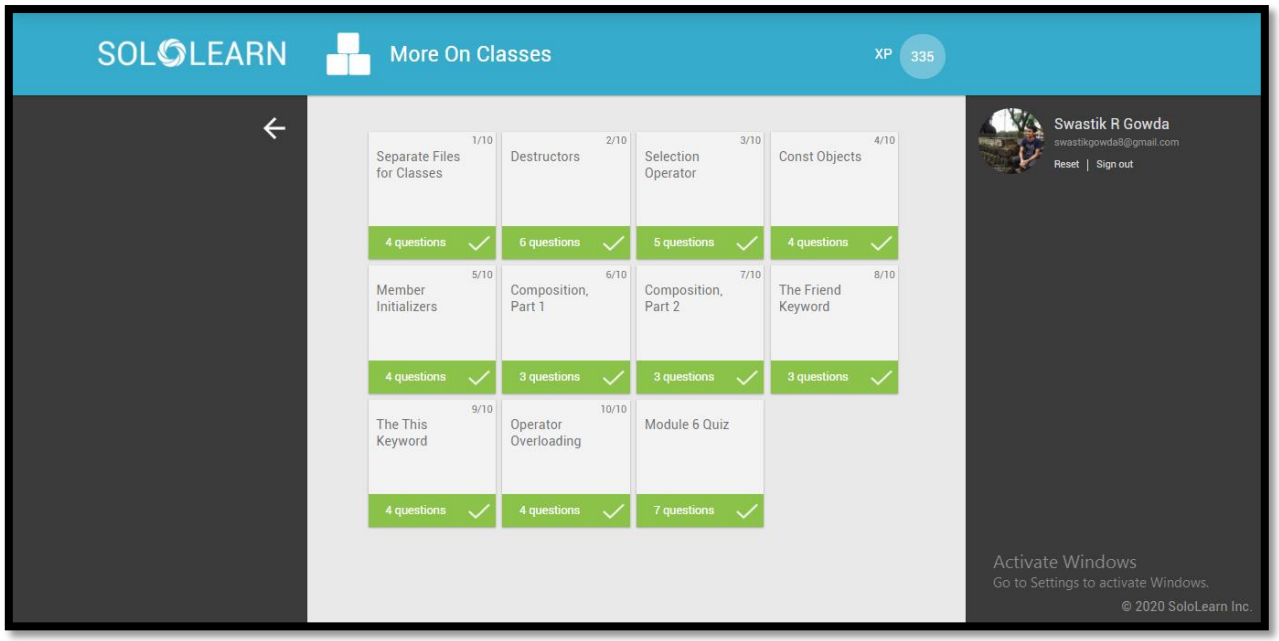
- Class constructors are special member functions of a class. They are executed whenever new objects are created within that class.
- The constructor's name is identical to that of the class. It has no return type, not even void.
- Constructors can be very useful for setting initial values for certain member variables.
- A default constructor has no parameters. However, when needed, parameters can be added to a constructor.



Date:	24-June-2020	Name:	Swastik R Gowda
Course:	Solo-Learn C++	USN:	4AL17EC091
Topic:	Module - 6 : More On Classes	Semester & Section:	6th Sem 'B' Sec

AFTERNOON SESSION DETAILS

Image of session



Report – Report can be typed or hand written for up to two pages.

Friend Functions

- Normally, private members of a class cannot be accessed from outside of that class.
- However, declaring a non-member function as a friend of a class allows it to access the class' private members.
- This is accomplished by including a declaration of this external function within the class, and preceding it with the keyword friend.
- In the example below, someFunc (), which is not a member function of the class, is a friend of MyClass and can access its private members.

```
class MyClass {
 public:
  MyClass() {
   regVar = 0;
  }
 private:
  int regVar;

  friend void someFunc(MyClass &obj);
};
```

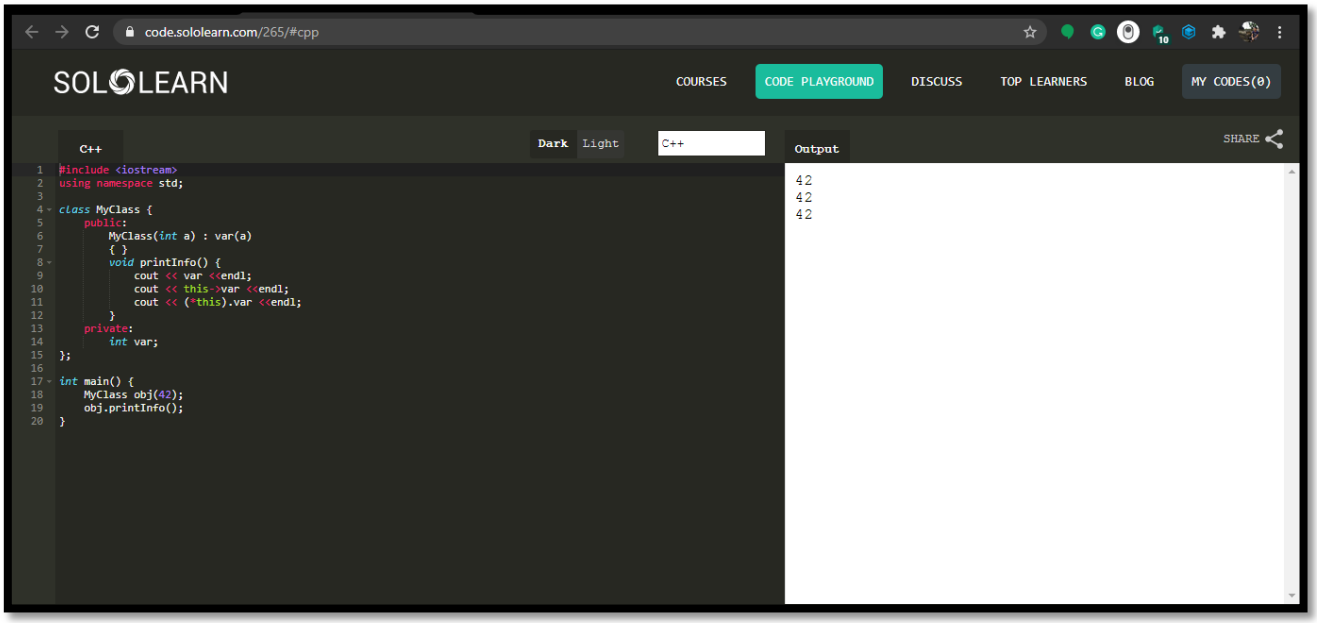
This pointer

Every object in C++ has access to its own address through an important pointer called the ***this*** pointer. Inside a member function this may be used to refer to the invoking object.

```
#include <iostream>
using namespace std;

class MyClass {
 public:
  MyClass(int a) : var(a)
  { }
  void printInfo() {
   cout << var <<endl;
   cout << this->var <<endl;
   cout << (*this).var <<endl;
  }
 private:
  int var;
};

int main() {
  MyClass obj(42);
  obj.printInfo();
}
```



Operator Overloading

- Most of the C++ built-in operators can be redefined or overloaded.
- Thus, operators can be used with user-defined types as well (for example, allowing you to add two objects together).
- Overloaded operators are functions, defined by the keyword operator followed by the symbol for the operator being defined.
- An overloaded operator is similar to other functions in that it has a return type and a parameter list.

This chart shows the operators that can be overloaded.

+	-	*	/	%	^
&	|	~	!	,	=
<	>	<=	>=	++	--
<<	>>	==	!=	&&	||
+=	-=	/=	%=	^=	&=
|=	*=	<<=	>>=	[]	()
->	->*	new	new[]	delete	delete[]