DAILY ASSESSMENT FORMAT

Date:	25-06-2020	Name:	Jagadeesha Hegde
Course:	C++ programming	USN:	4AL17EC036
Topic:	Inheritance and Polymorphism	Semester & Section:	6th A-sec
Github Repository:	Jagadeesha-036		

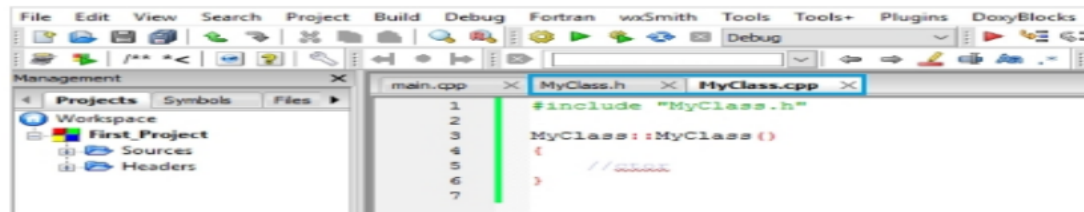| FORENOON SESSION DETAILS |
|---|

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.

Class definition

Class name: MyClass

Arguments:

☐ Has destructor   ☐ Has copy ctor
☑ Virtual destructor   ☐ Has assigment op.

Inheritance

☐ Inherits another class

Ancestor:

Ancestor's include filename: <>

Scope: public

Member variables

Add new:
unsigned int m_Counter
☑ Add "Getter" method
☑ Add "Setter" method
☑ Remove prefix:
m_

Remove   Add

Documentation
☐ Add documentation where appropriate

File policy

☑ Add paths to project   ☐ Use relative path
☑ Header and implementation file shall be in same folder
Folder: C:\sololearn\First_Project\   ...
☐ Header and implementation file shall always be lower case

Header file

Folder: C:\sololearn\First_Project\includ ...

Filename: MyClass.h
☑ Add guard block in header file

Guard block: MYCLASS_H

Implementation file

☑ Generate implementation file

Folder: C:\sololearn\First_Project\sr ...

Filename: MyClass.cpp

Header include: "MyClass.h"

Create   Cancel

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

```
File   Edit   View   Search   Project   Build   Debug   Fortran   wxSmith   Tools   Tools+   Plugins   DoxyBlocks

Management
  Projects   Symbols   Files
  Workspace
  First_Project
    Sources
    Headers

main.cpp   MyClass.h   MyClass.cpp
1    #include "MyClass.h"
2
3    MyClass::MyClass()
4    {
5        //ctor
6    }
7
```

The new files act as templates for our new

Polymorphism:

The word polymorphism means having many forms. Typically, polymorphism occurs when there is a hierarchy of classes and they are related by inheritance.C++ polymorphism means that a call to a member function will cause a different function to be executed depending on the type of object that invokes the function.

Difference between Inheritance and Polymorphism :

Inheritance:

Inheritance is one in which a new class is created that inherits the properties of the already exist class. It supports the concept of code reusability and reduces the length of the code in object-oriented programming.

Types of Inheritance are:

1. Single inheritance

2. Multi-level inheritance

3. Multiple inheritance

4. Hybrid inheritance

5. Hierarchical inheritance

Polymorphism:

Polymorphism is that in which we can perform a task in multiple forms or ways. It is applied to the functions or methods. Polymorphism allows the object to decide which form of the function to implement at compile-time as well as run-time.

Types of Polymorphism are:

1. Compile-time polymorphism (Method overloading)

2. Run-time polymorphism (Method Overriding)

Example of Polymorphism:

Polymorphism in C++

The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. Real life example of polymorphism, a person at the same time can have different characteristic. Like

a man at the same time is a father, a husband, an employee. So the same person posses different behavior in different situations. This is called polymorphism. Polymorphism is considered as one of the important features of Object Oriented Programming.

In C++ polymorphism is mainly divided into two types:

1. Compile time Polymorphism

2. Runtime Polymorphism

C++ Inheritance. In C++, inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. ... In C++, the class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. Inheritance in C++ The capability of a class to derive properties and characteristics from another class is called Inheritance.

Inheritance is one of the most important feature of Object Oriented Programming.

Sub Class: The class that inherits properties from another class is called Sub class or Derived Class. Super Class:The class whose properties are inherited by sub class is called Base Class or Super class.

DAILY ASSESSMENT FORMAT

Date:	25-06-2020	Name:	Jagadeesha Hegde
Course:	C++ programming	USN:	4AL17EC036
Topic:	Templates, exceptions and files	Semester & Section:	6th A-sec

Github Repository:	Jagadeesha-036		

AFTERNOON SESSION DETAILS

Image of session

The header file (.h) holds the function declarations (prototypes) and variable declarations.
It currently includes a template for our new **MyClass** class, with one default constructor.
MyClass.h

```
#ifndef MYCLASS_H
#define MYCLASS_H

class MyClass
{
  public:
    MyClass();
  protected:
  private:
};

#endif // MYCLASS_H
```

The implementation of the class and its methods go into the source file (.cpp).
Currently it includes just an empty constructor.
MyClass.cpp

```
#include "MyClass.h"

MyClass::MyClass()
{
    //ctor
}
```

The #ifndef and #define statements in the header file will be discussed in the upcoming lessons.

Templates in C++

Templates are the foundation of generic programming, which involves writing code in a way that is independent of any particular type.A template is a blueprint or formula for creating a generic class or a function. The library containers like iterators and algorithms are examples of generic programming and have been developed using template concept.There is a single definition of each container, such as vector, but we can define many different kinds of vectors for example, vector or vector .A template is a simple and yet very powerful tool in C++. The simple idea is to pass data type as a parameter so that we don't need to write the same code for different data types. For example, a software company may need sort() for different data types. Rather than writing and maintaining the multiple codes, we can write one sort() and pass data type as a parameter.C++ adds two new keywords to support templates: 'template' and 'typename'. The second keyword can always be replaced by keyword 'class'.How templates work? Templates are expanded at compiler time. This is like macros. The difference is, compiler does type checking before template expansion. The idea is simple, source code contains only function/class, but compiled code may contain multiple copies of same function/class.

Files :

C++ provides the following classes to perform output and input of characters to/from files:

 1.ofstream: Stream class to write on files

 2.ifstream: Stream class to read from files

 3.fstream: Stream class to both read and write from/to files.

 These classes are derived directly or indirectly from the classes istream and ostream. We have already used objects whose types were these classes: cin is an object of class istream and cout is an object of class ostream. Therefore, we have already been using classes that are related to our file streams. And in fact, we can use our file streams the same way we are already used to use cin and cout, with the only difference that we have to associate these streams with physical files.

Exceptions :

An exception is a problem that arises during the execution of a program. A C++ exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero. Exceptions provide a way to transfer control from one part of a program to another. C++ exception handling is built upon three keywords: try, catch, and throw.

1. throw – A program throws an exception when a problem shows up. This is done using a throw keyword.
2. catch – A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.
3. try – A try block identifies a block of code for which particular exceptions will be activated. It's· followed by one or more catch blocks.

Assuming a block will raise an exception, a method catches an exception using a combination of the try and catch keywords. A try/catch block is placed around the code that might generate an exception.