Agenda

- Template
- Friend Function
- Operator Overloading
- Association
- Inheritance
- Mode of Inheritance
- Diamond Problem
- Virtual Base Class
- Virtual Function
- Abstarct class

Function Template (demo01.cpp)

- It is function that can work for any type of datatype

class Template (demo02.cpp & demo03.cpp)

- It is class that can work for any type of datatype

Friend Function (demo04.cpp)

- friend function is a non member function of a class which is able to access the private members of the class.
- to make the function friend declare that function as friend inside the class

Operator Overloading (demo05.cpp)

- the operators does not work with user defined types.
- to make them work with user defined types we have tp overload the operators

Hirerachy

- It is divided into 2 categories based on relationship
- 1. Association
  - if has-a relationship exists between two entities
- 2. Inheritance
  - if is-a relationship exists between two entities

Association (demo06.cpp)

- has-a relationship
- it has 2 sub categories
    1. Composition
        - if has-a relation is tightly coupled
    2. Aggegration
        - if has-a relation is loosly coupleds

Inheritance (demo07.cpp)

- is-a relationship
- Types of inheritance

Mode of Inheritance (demo08.cpp)

- Default mode of inheritance is private

Diamond Problem(demo09.cpp)

- If hybrid inheritance is formed then the most derived class gets confusted from where it should access the memebers from.
- This ambugity is called as diamond problem
- To resolve this ambugity we have to make the base class as virtual at the time of inheritance

Upcasting (demo10.cpp)

- keeping the object of derived class in to base class pointer is called as upcasting

Downcasting(demo10.cpp)

- Converting the base class pointer in to derived class pointer is called as downcasting
- at the time of downcastig the base class pointer should contain the object of derived class i.e upcasting should have been done before performing downcasting

Object slicing(demo10.cpp)

- If upcasting is done the the base class pointer will only point at the memebrs of the base class that are inherited into the derived class.
- It will not point at the memebrs of the derived class.

Function overriding (demo11.cpp)

- If at the time of inheritance the derived class redefines the same function once again that already exists in base class then this is called as function overriding

virtual function (demo12.cpp)

- if you want to call the derived class overriden functions using the base class pointer then declare such functions as virtual inside base class

Abstract Class (demo13.cpp)

- If the base class is not aware about the implementation of the function and it wants the derived class to compulsary implement that function then make such functions as pure virtual in base class.
- If a pure virtual function exists into any class then the class becomes abstract class.
- We cannot cretae the object of the abstract class