Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

Lab Number:	**6**
Student Name:	**Sarika Laxmikant Galphade**
Roll No :	**36**

Title:

1. To perform Multiple Inheritance in C++. Create a student class representing student roll number, name and branch and an exam class (derived class of student) representing the scores of the student in various subjects (maths, physics and chemistry) and sports class representing the score in sports. The sports and exam class isinherited by a result class which adds the exam marks and sports score to generate the final result.
2. To perform Hierarchical Inheritance in C++. Create an Employee class with attributes EmpID and EmpSalary. Also create necessary methods/constructors to accept these values from the user. Create classes permenantEmployee and TemporaryEmployee which will be derived classes of Employee. Mention hike attribute in these derived classes and calculate the total salary using generate_salary() method for respective types of employees. Objects of the derived classes should be created and salaries for the permanent and temporary employees should be calculated and displayed on the screen.

Learning Objective:

- Students will be able to perform multiple inheritance using C++.

Learning Outcome:

- Understanding the inheritance concept and reusability of the code.

Course Outcome:

ECL304.2	Comprehend building blocks of OOPs language, inheritance, package and interfaces

Theory:

- Explain in details about inheritance, its types, syntaxes and block diagrams.

Inheritance is a mechanism of acquiring the features and behaviours of a class by another class. The class whose members are inherited is called the base class, and the class that inherits those members is called the derived class.

Syntax:

```
// Base class
class A
{
```

```
        public void score()
        {
        // TO DO:
        }
}
// Derived Class
Class B:A
{
public void score()
{
//TO DO:
}
}
```

DIFFERENT TYPES OF INHERITANCE.

OOPS support the six different types of inheritance as given below:

1. Single inheritance: In this inheritance, a derived class is created from a single base class.
   In the given example, Class A is the parent class and Class B is the child class since Class B inherits the features and behavior of the parent class A.

2. Multi-level inheritance: In this inheritance, a derived class is created from another derived class.
   In the given example, class c inherits the properties and behavior of class B and class B inherits the properties and behavior of class B. So, here A is the parent class of B and class B is the parent class of C. So, here class C implicitly inherits the properties and behavior of class A along with Class B i.e there is a multilevel of inheritance.

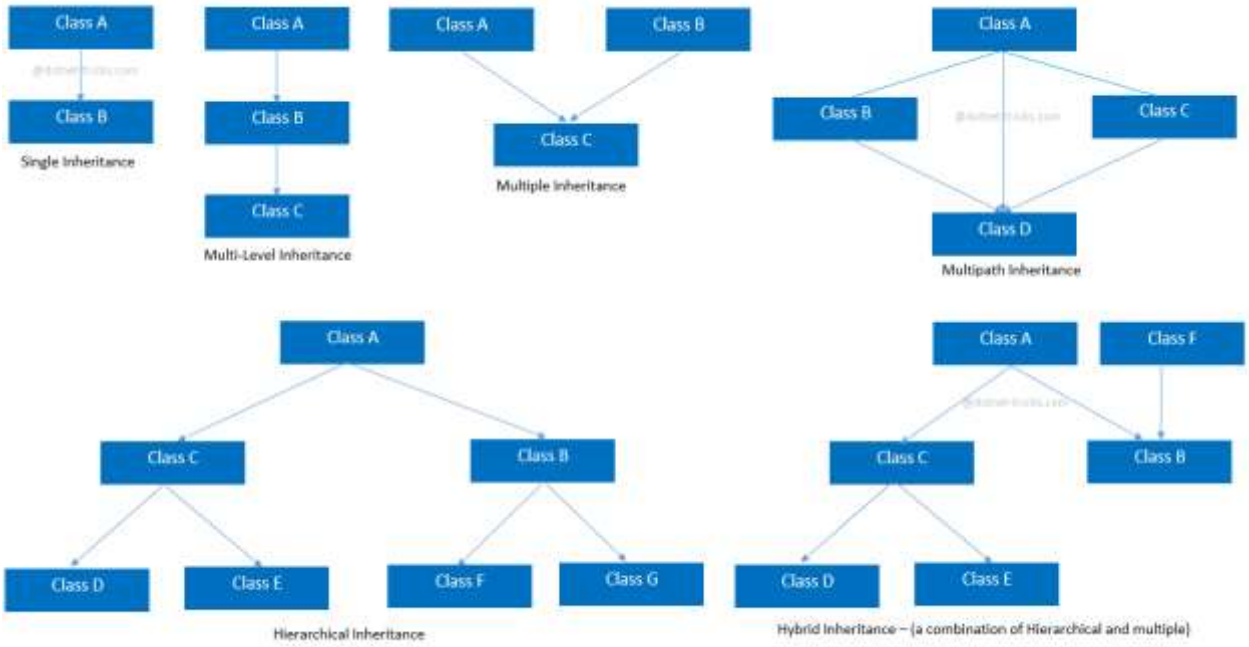
3. Multiple inheritance: In this inheritance, a derived class is created from more than one base class. This inheritance is not supported by .NET Languages like C#, F# etc. and Java Language.In the given example, class c inherits the properties and behavior of class B and class A at same level. So, here A and Class B both are the parent classes for Class C.

4.Multipath inheritance: In this inheritance, a derived class is created from another derived classes and the same base class of another derived classes. This inheritance is not supported by .NET Languages like C#, F# etc.
In the given example, class D inherits the properties and behavior of class C and class B as well as Class A. Both class C and class B inherits the Class A. So, Class A is the parent for Class B and Class C as well as Class D. So it's making it Multipath inheritance.

5.Hierarchical Inheritance: In this inheritance, more than one derived classes are created from a single base class and futher child classes act as parent classes for more than one child classes.
In the given example, class A has two childs class B and class D. Further, class B and class C both are having two childs - class D and E; class F and G respectively.

Don Bosco Institute of Technology, Kurla(W)
Department of Electronics and Tele-Communication Engineering
ECL304 - Skill Lab: C++ and Java Programming
Sem III
2021-22

.



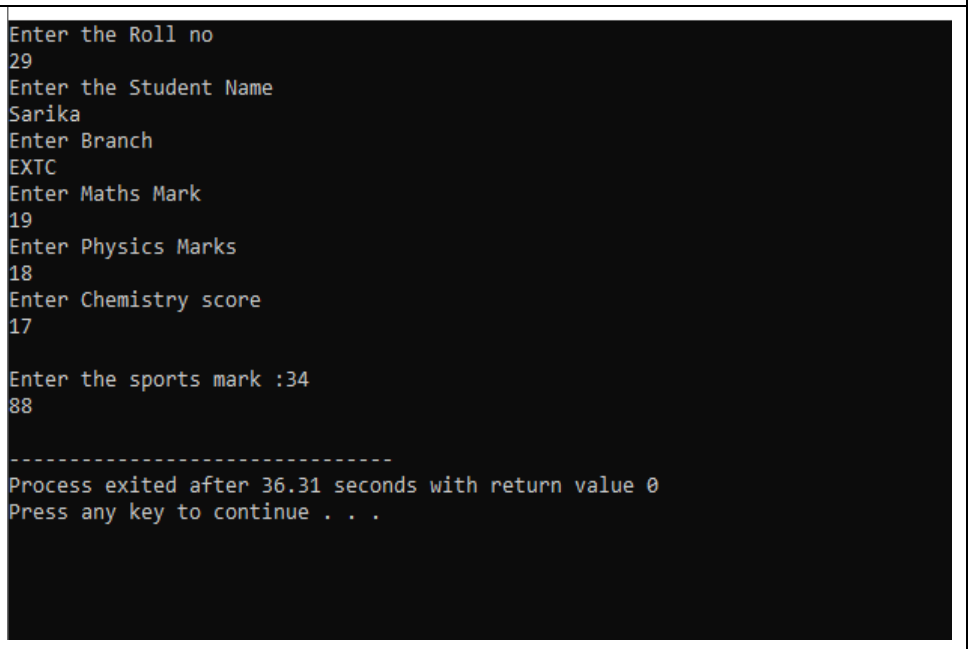
Algorithm :	• Step 1: Start the program. • Step 2: Declare the base class student. • Step 3: Declare and define the function get() to get the student details. • Step 4: Declare the other class sports. • Step 5: Declare and define the function getsm() to read the sports mark. • Step 6: Create the class statement derived from student and sports. • Step 7: Declare and define the function display() to find out the total and average. • Step 8: Declare the derived class object,call the functions get(),getsm() and display(). • Step 9: Stop the program.
Program:	#include<iostream>

Faculty: Ms. Deepali Kayande

```
using namespace std;
class student {
public:
   int Rollno;
        string name;
        string Branch;
student(){
      cout << "Enter the Roll no"<<endl;
      cin>>Rollno;
      cout << "Enter the Student Name"<<endl;
      cin >>name;
      cout<<"Enter Branch"<<endl;
      cin>>Branch;
   }
};
class exam{
        public:
        int maths;
        int physics;
        int chemistry;
        void getexam()
        {
                cout<<"Enter Maths Mark"<<endl;
                cin>>maths;
                cout<<"Enter Physics Marks"<<endl;
                cin>>physics;
                cout<<"Enter Chemistry score"<<endl;
                cin>>chemistry;
```

	}
	};
	class sports {
	public:
	int sport;
	void getsm() {
	cout << "\nEnter the sports mark :";
	cin>>sport;
	}
	};
	class result : public exam, public sports {
	public:
	int total;
	result() {
	total = maths + physics + chemistry + sport ;
	cout << total<<endl;
	}
	};
	
	int main() {
	result obj;
	return 0;
	}

Input given:	**Roll no.29** **Name: Sarika** **Branch: EXTC** **Maths score: 17** **Physics score:18**

	Chemistry score:17 **Sports Sore: 34**
Output Screenshot:	

Algorithm :	• Step 1: Start the program. • Step 2: Declare the base class student. • Step 3: Declare and define the function get() to get the student details. • Step 4: Declare the other class sports. • Step 5: Declare and define the function getsm() to read the sports mark. • Step 6: Create the class statement derived from student and sports. • Step 7: Declare and define the function display() to find out the total and average. • Step 8: Declare the derived class object,call the functions

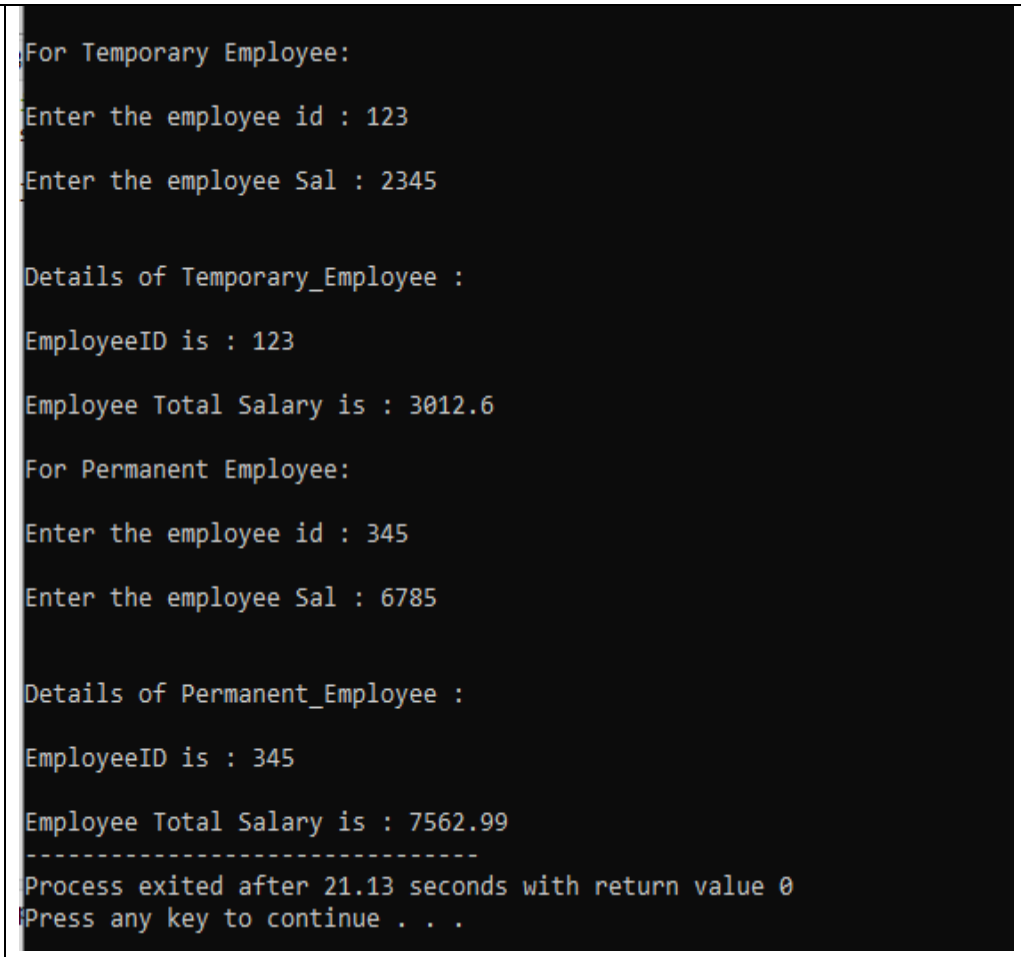
	get(),getsm() and display(). • Step 9: Stop the program.

Program:	#include <iostream>
	using namespace std;
	
	class Employee
	{
	protected:
	string EmpID = "";
	
	double Empsalary;
	
	public:
	Employee()
	{
	cout<<endl<<"Enter the employee id : ";
	cin>>EmpID;
	cout<<endl<<"Enter the employee Sal : ";
	cin>>Empsalary;
	
	}
	
	void getDetails()
	{
	cout <<endl<< "EmployeeID is : " << EmpID;
	cout <<endl<<endl<<"Employee Total Salary is : " << Empsalary;

```
        }



};


class Permanent_Employee: public Employee
{
        double hike;


        public:


        Permanent_Employee( double increment)
        {
                hike = increment;
        }


   void getDetails()
        {
                cout <<endl<< "EmployeeID is : " << EmpID;
                cout <<endl<<endl<<"Employee Total Salary is : " <<
generate_salary();
        }


        float generate_salary()
        {
                return (Empsalary + hike);
        }
};
```

```
class Temporary_Employee: public Employee
{
        double hike;

        public:

        Temporary_Employee( double increment)
        {
                hike = increment;
        }

   void getDetails()
        {
                cout <<endl<< "EmployeeID is : " << EmpID;
                cout <<endl<<endl<<"Employee Total Salary is : " <<
generate_salary();
        }

        float generate_salary()
        {
                return (Empsalary + hike);
        }
};

int main()
{
```

	cout<<endl<<"For Temporary Employee: "<<endl; Temporary_Employee T(667.6); cout << endl << endl << "Details of Temporary_Employee : " << endl; T.getDetails(); cout<<endl<<endl<<"For Permanent Employee: "<<endl; Permanent_Employee P(777.99); cout << endl << endl << "Details of Permanent_Employee : " << endl; P.getDetails(); return 0; }
Input given:	**Employee id 123 ,employee sal 2345** **Employee id 345 ,employee sal 6785**

Output Screenshot:	

```
For Temporary Employee:

Enter the employee id : 123

Enter the employee Sal : 2345


Details of Temporary_Employee :

EmployeeID is : 123

Employee Total Salary is : 3012.6

For Permanent Employee:

Enter the employee id : 345

Enter the employee Sal : 6785


Details of Permanent_Employee :

EmployeeID is : 345

Employee Total Salary is : 7562.99
--------------------------------
Process exited after 21.13 seconds with return value 0
Press any key to continue . . .
```