Object Oriented Programming

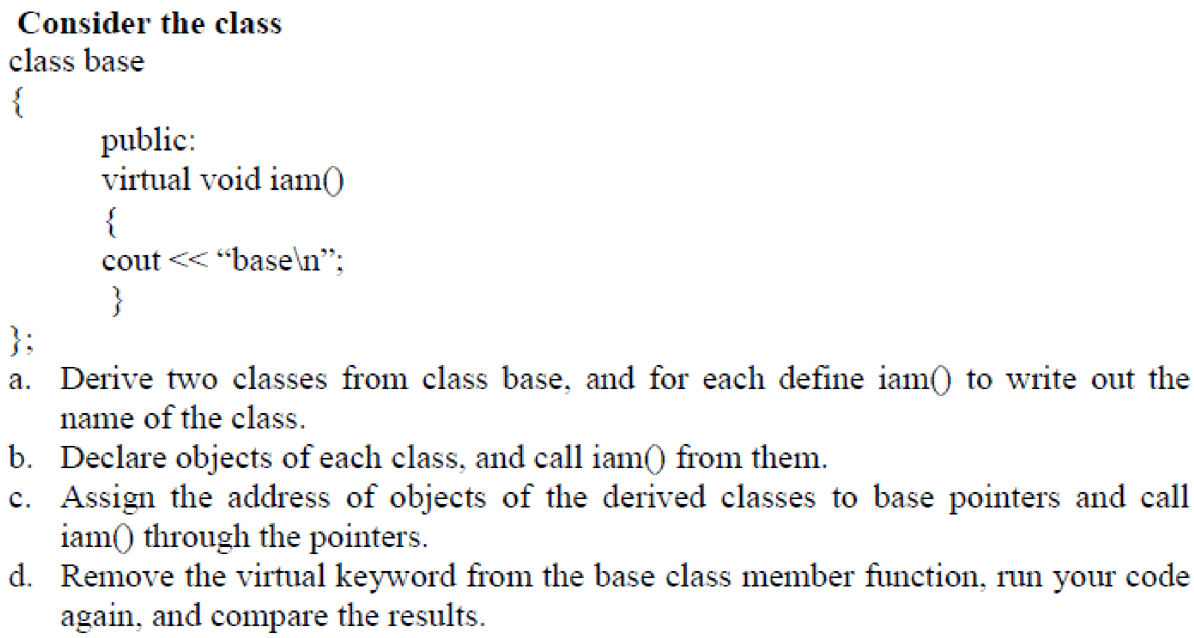
Lab Report

Lab10

Group Members Name & Reg #:	**Muhammad Haris Irfan** **(FA18-BCE-090)**
Class	Object Oriented Programming CSC241 (**BCE-4B**)
Instructor's Name	Maam Amber Madeeha Zeb

In Lab Tasks

5.1 Question 1:



Consider the class

```
class base
{
        public:
        virtual void iam()
        {
        cout << “base\n”;
        }
};
```

a. Derive two classes from class base, and for each define iam() to write out the name of the class.

b. Declare objects of each class, and call iam() from them.

c. Assign the address of objects of the derived classes to base pointers and call iam() through the pointers.

d. Remove the virtual keyword from the base class member function, run your code again, and compare the results.

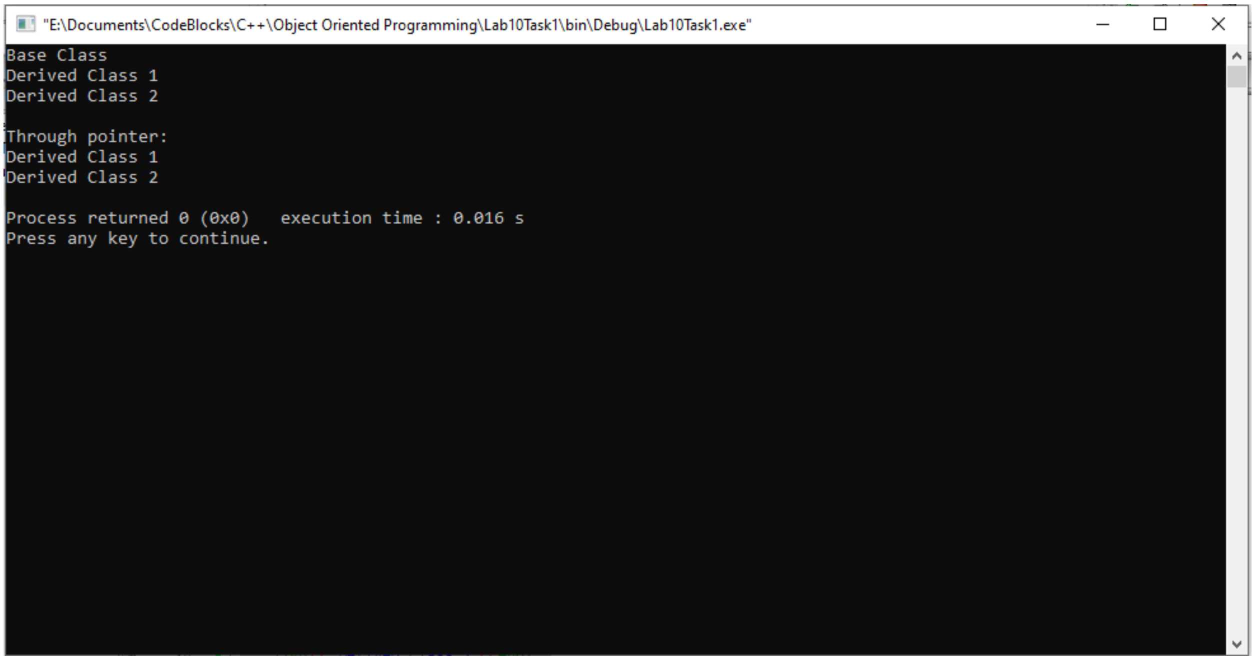
Solution:

The code is given below,

```
#include <iostream>

using namespace std;

class base
{
public:
    virtual void iam()
    {
    cout<<"Base Class"<<endl;
    }
};

class derv1 :public base
{
    public:
     void iam()
    {
    cout<<"Derived Class 1"<<endl;
    }
};

class derv2 :public base
```

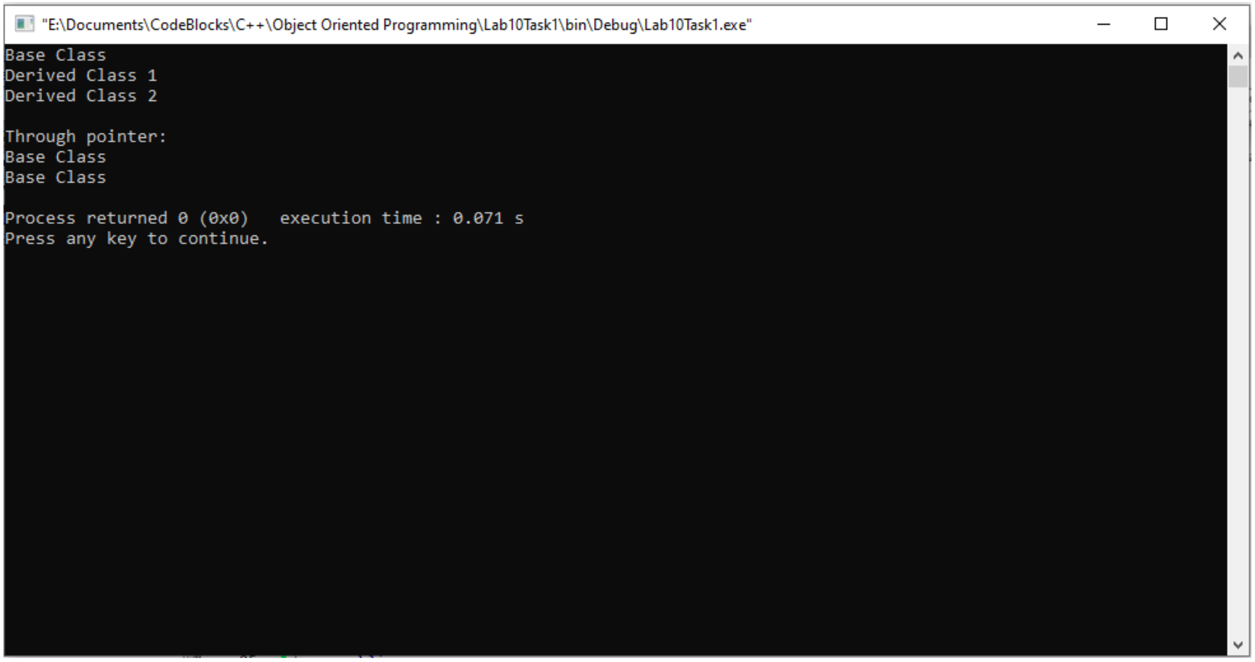
```cpp
{
    public:
     void iam()
    {
    cout<<"Derived Class 2"<<endl;
    }
};
int main()
{

    base b1;
    base *ptr1;
    derv1 d1;
    derv2 d2;


    b1.iam();
    d1.iam();
    d2.iam();

    cout<<endl<<"Through pointer:"<<endl;
    ptr1=&d1;
    ptr1->iam();

    ptr1=&d2;
    ptr1->iam();


    return 0;
}
```

Console Output is shown below.


"E:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab10Task1\bin\Debug\Lab10Task1.exe"

```
Base Class
Derived Class 1
Derived Class 2

Through pointer:
Derived Class 1
Derived Class 2

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

Now after removing the keyword virtual the output is as follows:


"E:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab10Task1\bin\Debug\Lab10Task1.exe"

```
Base Class
Derived Class 1
Derived Class 2

Through pointer:
Base Class
Base Class

Process returned 0 (0x0)   execution time : 0.071 s
Press any key to continue.
```

\

5.2 Question 2:

Develop a simple payroll application. There are three kinds of employees in the system: salaried employee, hourly employee, and commissioned employee. The system takes as input an array containing employee objects, calculates salary polymorphically, and generates report.
Make Employee an abstract class. Declare salary() and display() as pure virtual functions in it. Derive salaried employee (monthly), hourly employee (per hour basis), and commissioned employee (bonus on completing each target) from base class Employee. The display() function should show employee no, employee name, and salary of all employees.

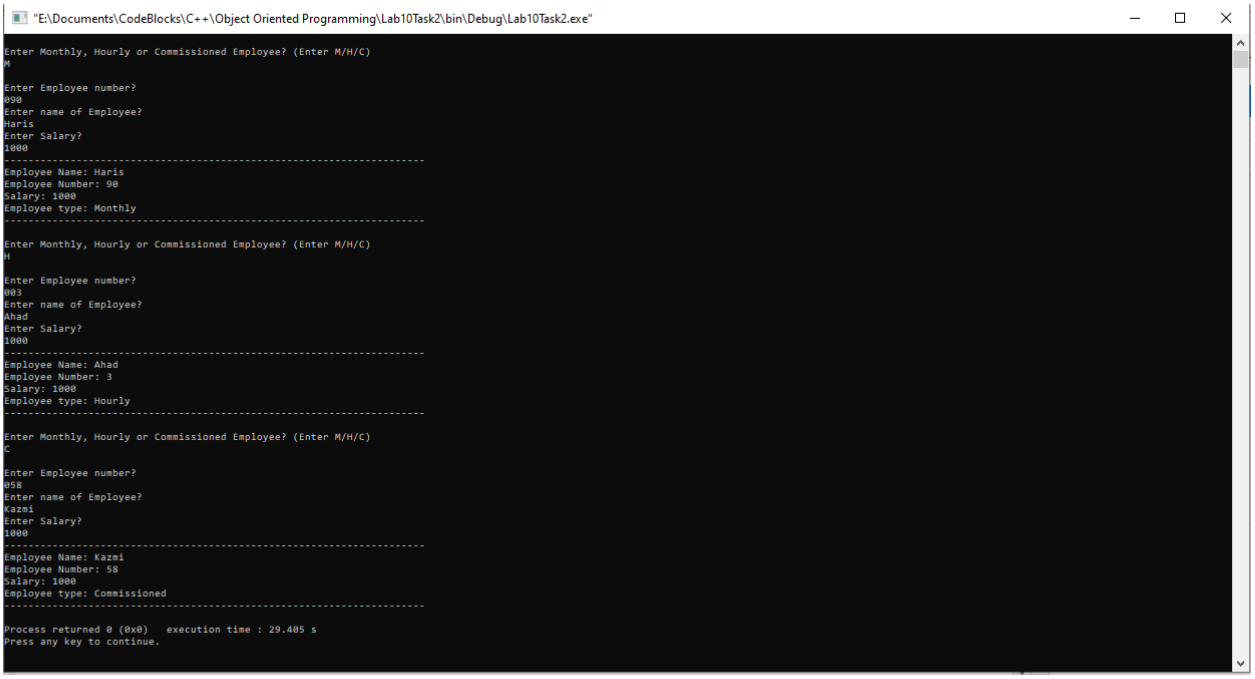
Solution:

The code is given below,

```
#include <iostream>

using namespace std;

class employee
{
private:

    int empno;
    string name;

public:
     virtual void salary()=0;
    virtual void display()=0;

 employee()
 {
   cout<<endl<<"Enter Employee number?"<<endl;
   cin>>empno;
   cout<<"Enter name of Employee?"<<endl;
   cin>>name;

 }

void putdata()
{
    cout<<"----------------------------------------------------------------------"<<endl;
    cout<<"Employee Name: "<<name<<endl;
    cout<<"Employee Number: "<<empno<<endl;
}

};

class monthEmploy : public employee
{
    float sal;
public:
    void salary()
    {
     cout<<"Enter Salary?"<<endl;
     cin>>sal;
    }
    void display()
    {
       employee::putdata();
       cout<<"Salary: "<<sal<<endl;
       cout<<"Employee type: Monthly"<<endl;
       cout<<"----------------------------------------------------------------------"<<endl;
    }
};

class hourlyEmploy : public employee
{
    float sal;
public:
    void salary()
    {
     cout<<"Enter Salary?"<<endl;
     cin>>sal;
    }
     void display()
    {
       employee::putdata();
       cout<<"Salary: "<<sal<<endl;
       cout<<"Employee type: Hourly"<<endl;
       cout<<"----------------------------------------------------------------------"<<endl;
    }
};

class commisEmploy : public employee
{
    float sal;
public:
    void salary()
    {
     cout<<"Enter Salary?"<<endl;
     cin>>sal;

    }

      void display()
    {
       employee::putdata();
       cout<<"Salary: "<<sal<<endl;
       cout<<"Employee type: Commissioned"<<endl;
       cout<<"----------------------------------------------------------------------"<<endl;
    }
};
int main()
{
  employee *ptr[3];
  char op;
  for(int i=0;i<3;i++)
  {
      cout<<endl<<"Enter Monthly, Hourly or Commissioned Employee? (Enter M/H/C)"<<endl;
      cin >>op;
      if(op=='M')
      {
          ptr[i]= new monthEmploy;
          ptr[i]->salary();
          ptr[i]->display();

      }
      if(op=='H')
      {
          ptr[i]= new hourlyEmploy;
          ptr[i]->salary();
          ptr[i]->display();

      }
      if(op=='C')
      {
          ptr[i]= new commisEmploy;
```

```cpp
          ptr[i]->salary();
          ptr[i]->display();

      }

  }
    return 0;
}
```

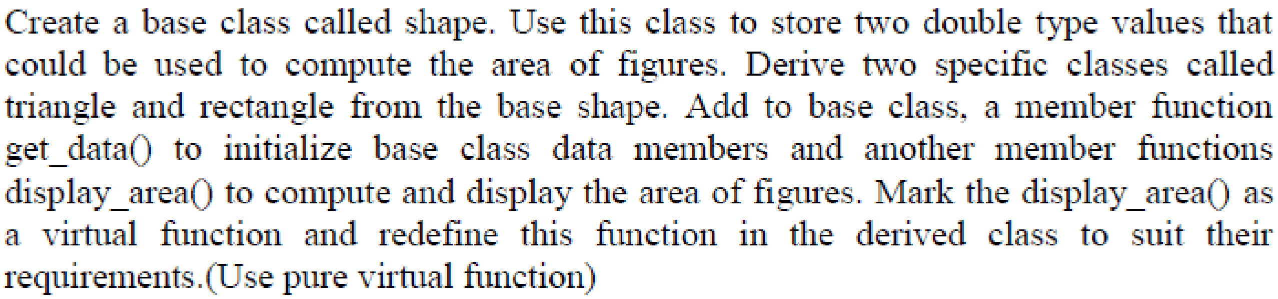
Console Output is shown below.



"E:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab10Task2\bin\Debug\Lab10Task2.exe"

```
Enter Monthly, Hourly or Commissioned Employee? (Enter M/H/C)
M

Enter Employee number?
090
Enter name of Employee?
Haris
Enter Salary?
1000
----------------------------------------------------------------------
Employee Name: Haris
Employee Number: 90
Salary: 1000
Employee type: Monthly
----------------------------------------------------------------------

Enter Monthly, Hourly or Commissioned Employee? (Enter M/H/C)
H

Enter Employee number?
003
Enter name of Employee?
Ahad
Enter Salary?
1000
----------------------------------------------------------------------
Employee Name: Ahad
Employee Number: 3
Salary: 1000
Employee type: Hourly
----------------------------------------------------------------------

Enter Monthly, Hourly or Commissioned Employee? (Enter M/H/C)
C

Enter Employee number?
058
Enter name of Employee?
Kazmi
Enter Salary?
1000
----------------------------------------------------------------------
Employee Name: Kazmi
Employee Number: 58
Salary: 1000
Employee type: Commissioned
----------------------------------------------------------------------

Process returned 0 (0x0)   execution time : 29.405 s
Press any key to continue.
```

5.3 Question 3:



Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to base class, a member function get_data() to initialize base class data members and another member functions display_area() to compute and display the area of figures. Mark the display_area() as a virtual function and redefine this function in the derived class to suit their requirements.(Use pure virtual function)

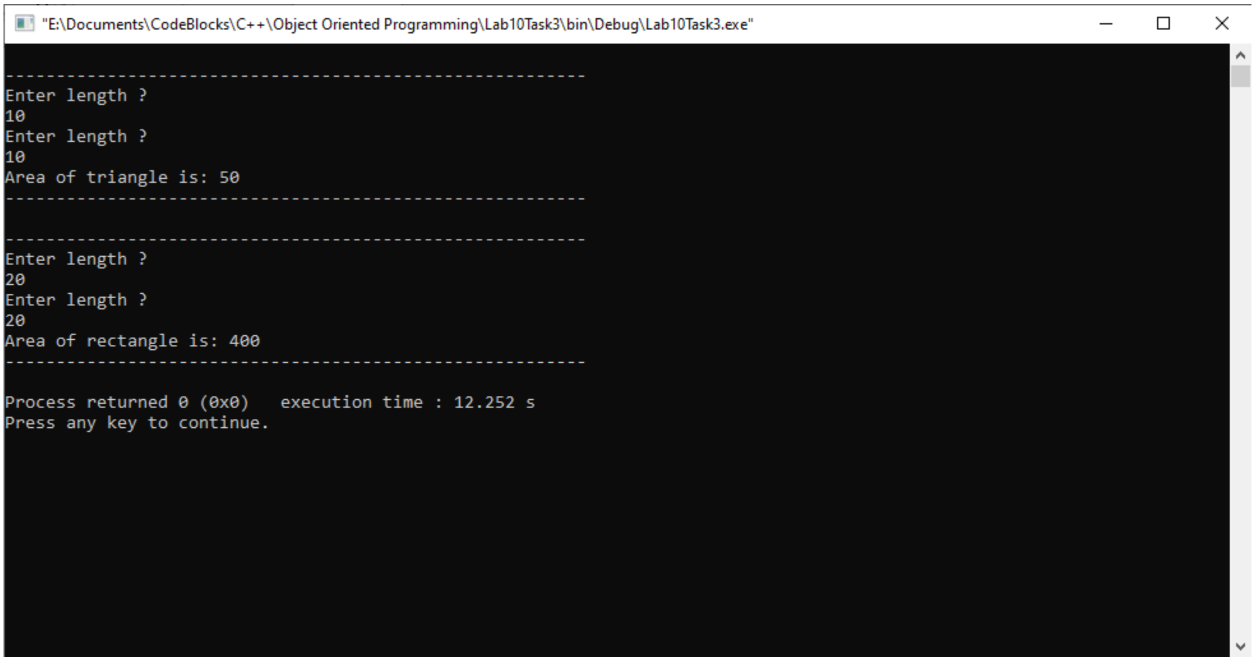
Solution:

The code is given below,

```
#include <iostream>

using namespace std;

class base
{
private:
    double width;
    double length;

public:
 virtual void display_area()=0;
    double getlength()
    {
        cout<< endl;
        cout<<"---------------------------------------------------------"<<endl;
        cout<<"Enter length ?"<<endl;
        cin>>length;
        return length;
    }
    double getwidth()
    {
        cout<<"Enter length ?"<<endl;
        cin>>width;
        return width;
    }

};

class triangle :public base
{
public:
    void display_area()
    {
        double areat;
        areat=0.5*getlength()*getwidth();
        cout<<"Area of triangle is: "<<areat<<endl;
        cout<<"---------------------------------------------------------"<<endl;
    }

};

class rectangle :public base
{
```

```cpp
    public:
    void display_area()
    {
        double arear;
        arear=getlength()*getwidth();
        cout<<"Area of rectangle is: "<<arear<<endl;
        cout<<"---------------------------------------------------------"<<endl;
    }

};
int main()
{
    triangle t1;
    rectangle r1;

    t1.display_area();
    r1.display_area();

    return 0;
}
```

Console Output is shown below.


"E:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab10Task3\bin\Debug\Lab10Task3.exe"

```
---------------------------------------------------------
Enter length ?
10
Enter length ?
10
Area of triangle is: 50
---------------------------------------------------------

---------------------------------------------------------
Enter length ?
20
Enter length ?
20
Area of rectangle is: 400
---------------------------------------------------------

Process returned 0 (0x0)   execution time : 12.252 s
Press any key to continue.
```

POST LAB

6.1 Question 4:

Create a class hierarchy that performs conversions from one system of units to another. Your program should perform the following conversions,

i. Liters to Gallons,

ii. Fahrenheit to Celsius and

iii. Feet to Meters

The base class **convert** declares two variables, val1 and val2, which hold the initial and converted values, respectively. It also defines the functions getinit() and getconv(), which return the initial value and the converted value. These elements of convert are fixed and applicable to all derived classes that will inherit convert. However, the function that will actually perform the conversion, compute(), is a pure virtual function that must be defined by the classes derived from convert. The specific nature of compute() will be determined by what type of conversion is taking place. Three classes will be derived from convert to perform conversions of Liters to Gallons (l_to_g), Fahrenheit to Celsius (f_to_c) and Feet to Meters (f_to_m), respectively. Each derived class overrides compute() in its own way to perform the desired conversion.

Test these classes from main() to demonstrate that even though the actual conversion differs between l_to_g, f_to_c, and f_to_m, the interface remains constant.

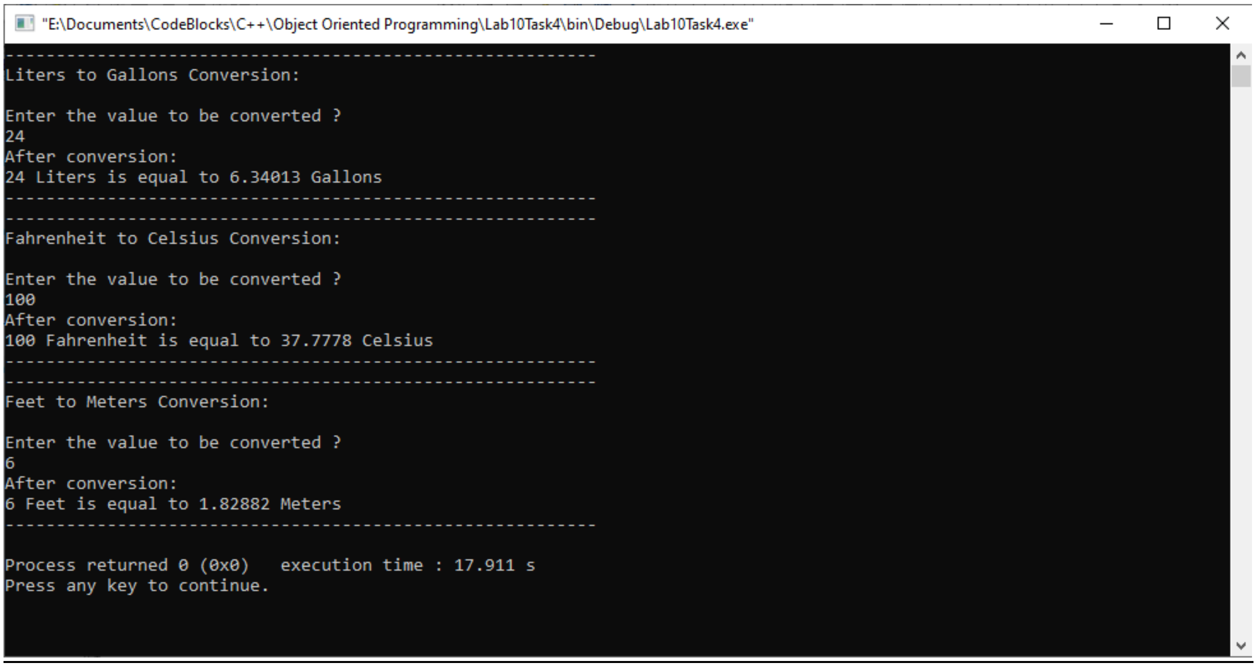
Solution:

I am attaching my code below,

```
#include <iostream>

using namespace std;

class convert
{

    public:
    double val1; ///initial value
    double val2;
    virtual void compute ()=0;

        double getint()
        {

            cout<<endl<<"Enter the value to be converted ?"<<endl;
            cin>>val1;
            return val1;
        }

        double getconv() const
        {

            return val2;
```

```cpp
        }

};
class litToGal:public convert
{
public:

    void compute()
    {
        val2=getint()*0.264172;
        cout<<"After conversion: "<<endl;
        cout<<val1<< " Liters is equal to "<<val2 <<" Gallons"<<endl;
        cout<<"----------------------------------------------------------"<<endl;
    }
};

class farToCel:public convert
{
public:

    void compute()
    {
        val2=(getint()-32)*(0.5555555);
        cout<<"After conversion: "<<endl;
        cout<<val1<< " Fahrenheit is equal to "<<val2 <<" Celsius"<<endl;
        cout<<"----------------------------------------------------------"<<endl;
    }
};


class feetToMet:public convert
{
public:

    void compute()
    {
        val2=getint()/3.2808;
        cout<<"After conversion: "<<endl;
        cout<<val1<< " Feet is equal to "<<val2 <<" Meters"<<endl;
        cout<<"----------------------------------------------------------"<<endl;
    }
};

int main()
{
    litToGal g1;
    farToCel  c1;
    feetToMet  m1;

    cout<<"----------------------------------------------------------"<<endl;
    cout<<"Liters to Gallons Conversion:"<<endl;
    g1.compute();

    cout<<"----------------------------------------------------------"<<endl;
    cout<<"Fahrenheit to Celsius Conversion:"<<endl;
    c1.compute();

    cout<<"----------------------------------------------------------"<<endl;
    cout<<"Feet to Meters Conversion:"<<endl;
    m1.compute();

    return 0;
}
```

The result for this program is shown below,


"E:\Documents\CodeBlocks\C++\Object Oriented Programming\Lab10Task4\bin\Debug\Lab10Task4.exe"

```
----------------------------------------------------------
Liters to Gallons Conversion:

Enter the value to be converted ?
24
After conversion:
24 Liters is equal to 6.34013 Gallons
----------------------------------------------------------
----------------------------------------------------------
Fahrenheit to Celsius Conversion:

Enter the value to be converted ?
100
After conversion:
100 Fahrenheit is equal to 37.7778 Celsius
----------------------------------------------------------
----------------------------------------------------------
Feet to Meters Conversion:

Enter the value to be converted ?
6
After conversion:
6 Feet is equal to 1.82882 Meters
----------------------------------------------------------

Process returned 0 (0x0)   execution time : 17.911 s
Press any key to continue.
```

_______THE END_________