**Object Oriented Programming Lab**

**BCS-DS-352**

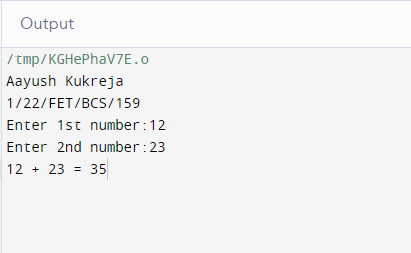
**Program 1**

**Aim:** C++ program to add two numbers.

**Code:**

****

**Output:**

****

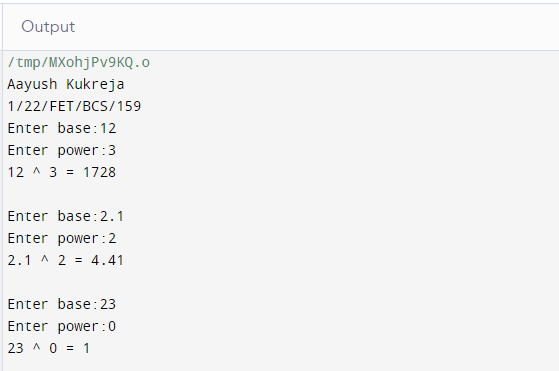
**Program 2**

**Aim:** Write a function called power that takes double values for n and an int value for p. It returns a double value. Use a default argument of 2 for p so that if this argument is omitted the number will be squared. Write a main function that gets value from the user to test this function.

**Code:**

****

**Output:**

****

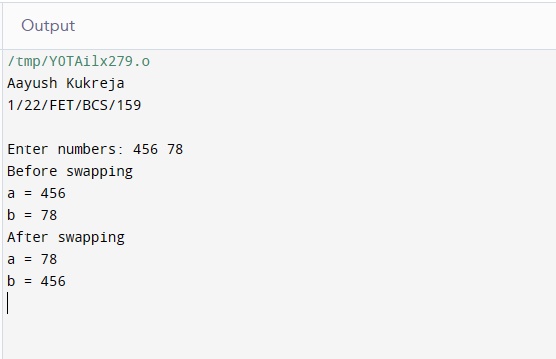
**Program 3**

**Aim:** C++ program to Swap two values using call by value.

**Code:**

****

**Output:**

****

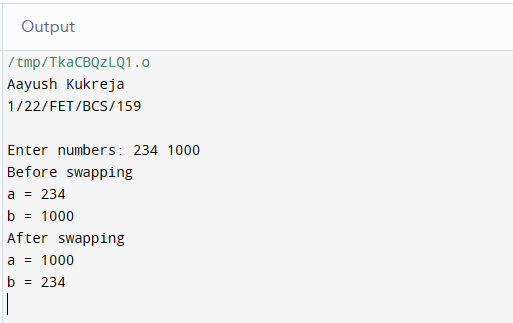
**Program 4**

**Aim:** C++ program to Swap two values using call by reference.

**Code:**

****

**Output:**

****

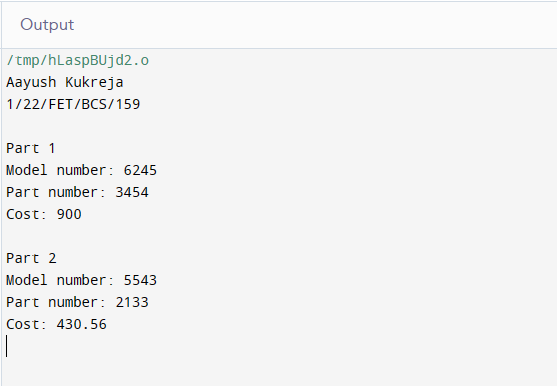
**Program 5**

**Aim:** Program to create a class 'Part' and create two objects of it.

**Code:**

****

**Output:**

****

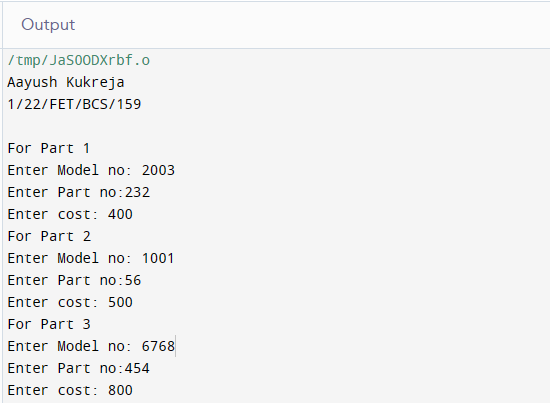
**Program 6**

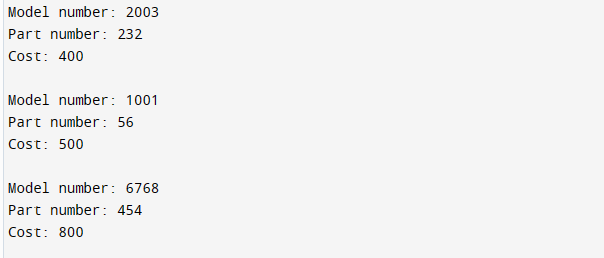
**Aim:** Program to create a class 'Part' and demonstrate the concept of array of objects.

**Code:**

****

**Output:**

****

****

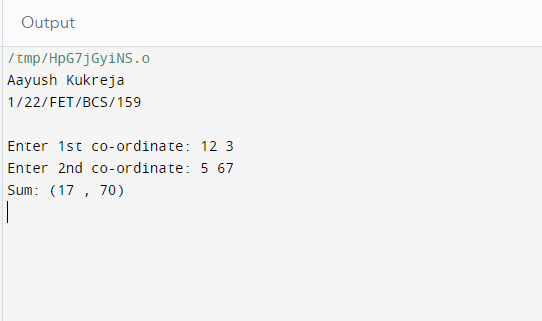
**Program 7**

**Aim:** A point on the two-dimensional plane can be represented by two numbers: an X coordinate and a Y coordinate. For example, (4,5) represents a point 4 units to the right of the origin along the X axis and 5 units up the Y axis. The sum of two points can be defined as a new point that X coordinate is the sum of the X coordinates of the points and whose Y coordinate is the sum of their Y coordinates. Write a program that uses a structure called point to model a point.

**Code:**

****

**Output:**

****

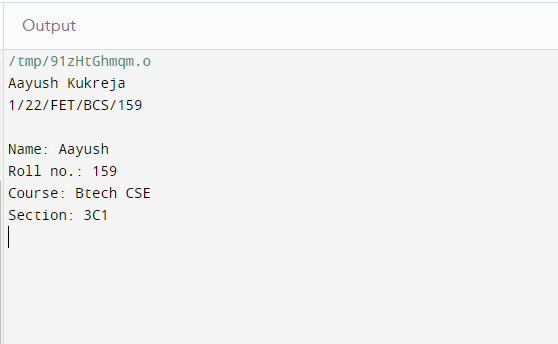
**Program 8**

**Aim:** Create student class and display student data of a single student.

**Code:**

****

**Output:**

****

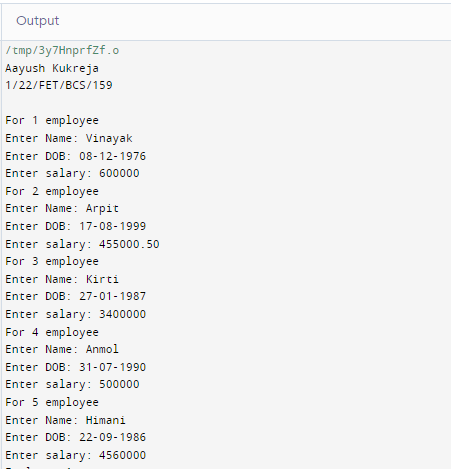
**Program 9**

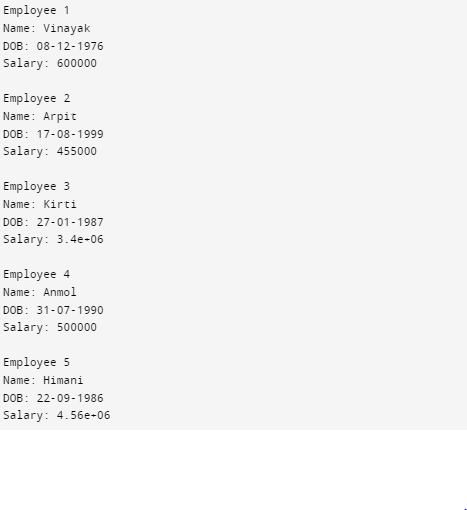
**Aim:** Write a C++ program to take input and display data of 5 employees using array of objects and constructors.

**Code:**

****

**Output:**

****

****

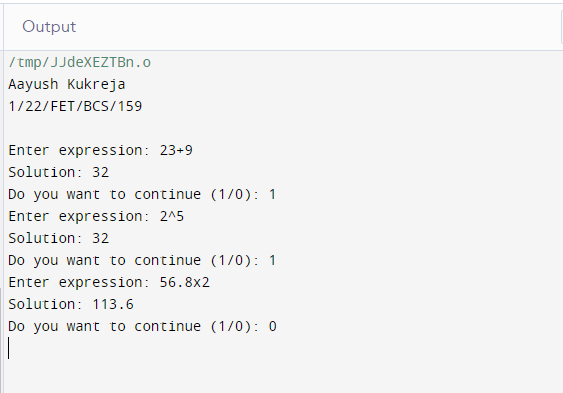
**Program 10**

**Aim:** Create the equivalent of a four-function calculator. The program should request the user to enter a number, an operator and another number. It should then carry out the specified arithmetical operation: adding. subtracting, multiplying or dividing the two numbers (It should use a switch statement to select the operation). Finally, it should display the result. When it finishes the calculation, the program should ask if the user wants to do another calculation.

**Code:**

****

**Output:**

****

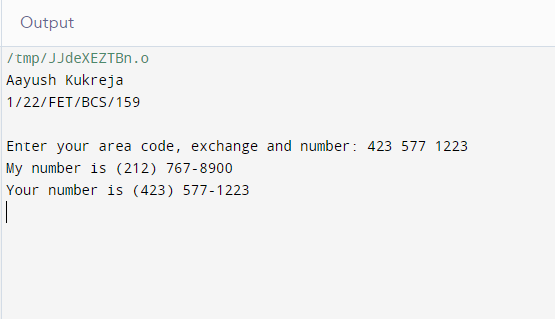
**Program 11**

**Aim:** Phone number, such as (212) 767-8900, can be thought of as having three parts: the area code (212), the exchange (767) and the number (8900). Write a program that uses a structure to store these three parts of a phone number separately. Call the structure phone. Create two structure variables of type phone. Initialize one and have the user input a number for the other one. Then display both numbers.

**Code:**



**Output:**

****

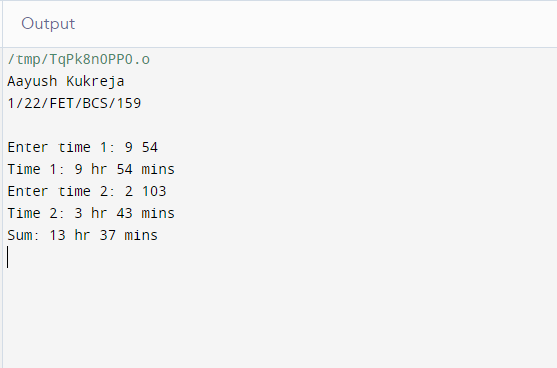
**Program 12**

**Aim:** Create a class Time that store hours and minutes. Inside it create a function that returns sum of two time.

**Code:**

****

**Output:**

****

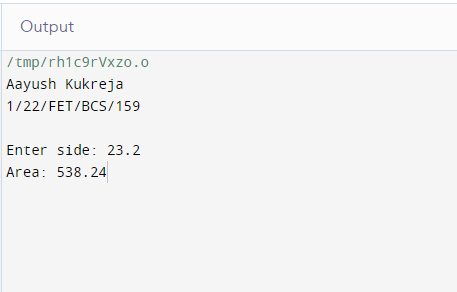
**Program 13**

**Aim:** Create a class ‘Square’ and find area of a square using friend function.

**Code:**



**Output:**



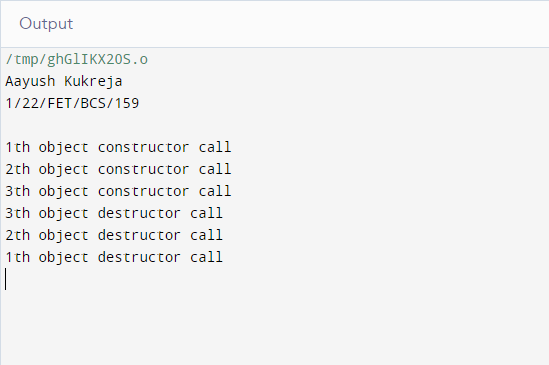
**Program 14**

**Aim:** Create a class and implement concept of destructors.

**Code:**

****

**Output:**

****

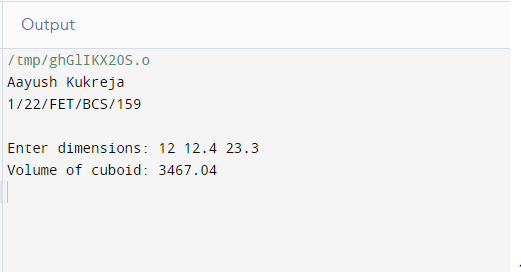
**Program 15**

**Aim:** C++ program to implement inline function.

**Code:**

****

**Output:**

****

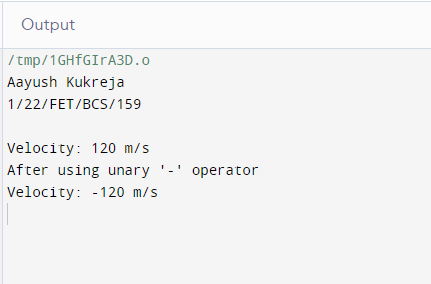
**Program 16**

**Aim:** C++ program to overload a unary '-' operator.

**Code:**

****

**Output:**

****

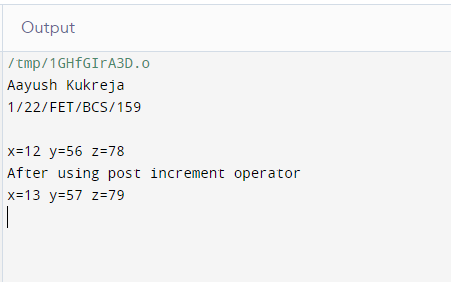
**Program 17**

**Aim:** C++ program to overload unary operator using friend function.

**Code:**



**Output:**

****

**Program 18**

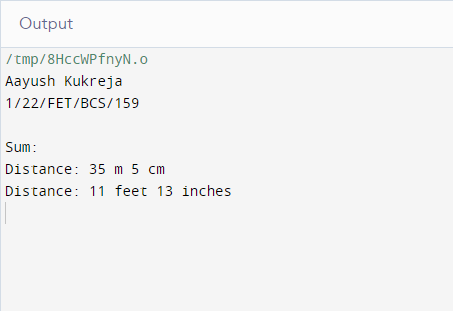
**Aim:** Create two classes DM and DB which store value of distances. DM stores distances in meters and centimeters and DB in feet and inches. Write a program that can read values for the class objects and add one object of DM with another object of DB. Use a friend function to carry out the addition operator. The object that stores the results maybe a DM object or DB object depending on the units in which the results are required. The display should be in the format of feet and inches or meters and centimeters depending on object on display.

**Code:**

****

****

**Output:**

****

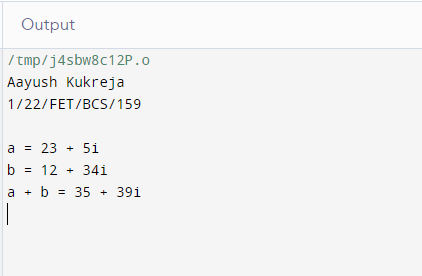
**Program 19**

**Aim:** C++program to overload binary '+' operator to add two complex numbers.

**Code:**

****

**Output:**

****

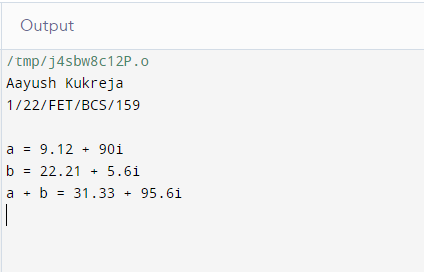
**Program 20**

**Aim:** C++program to overload binary '+' operator to add two complex numbers using friend function.

**Code:**

****

**Output:**

****

**Program 21**

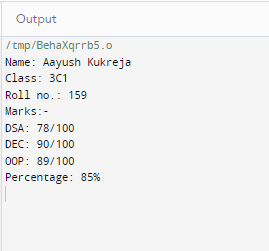
**Aim:** C++program to inherit Student class using single inheritance.

**Code:**

****

****

**Output:**

****

**Program 22**

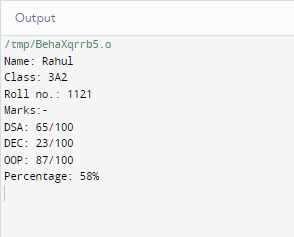
**Aim:** Write a C++ program to demonstrate multilevel inheritance.

**Code:**

****

****

**Output:**

****

**Program 23**

**Aim:** Create two classes named Mammals and Marine Animals. Create another class named Blue Whale which inherits both the above classes. Now, create a function in each of these classes which prints “I am mammal”, “I am a marine animal” and “I belong to both the categories: Mammals as well as Marine Animals” respectively. Now, create an object for each of the above class and try calling

1) function of Mammals by the object of Mammal

2) function of Marine Animal by the object of Marine Animal

3) function of Blue Whale by the object of Blue Whale

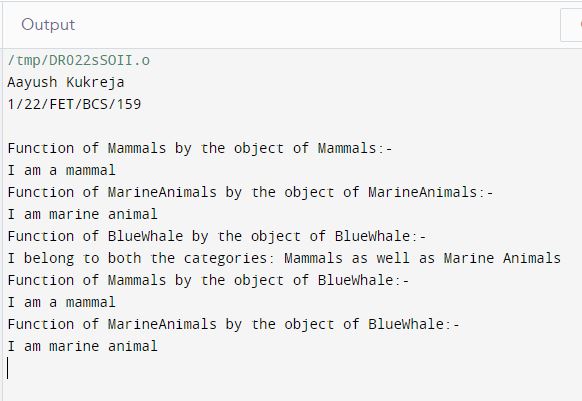
4) function of each of its parent by the object of Blue Whale

**Code:**

****

****

**Output:**

****

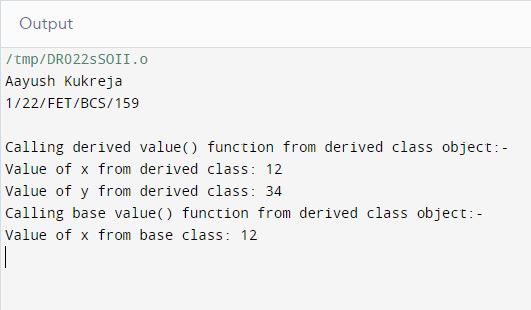
**Program 24**

**Aim:** Write a C++ program to demonstrate how arguments can be passed from a derived class constructor to a base class constructor.

**Code:**

****

**Output:**

****

**Program 25**

**Aim:** Write a C++ program to demonstrate the concept of abstract class and pure virtual function.

**Code:**





**Output:**

****

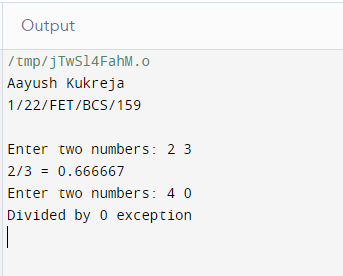
**Program 26**

**Aim:** Write a C++ program for throwing an exception.

**Code:**

****

**Output:**

****

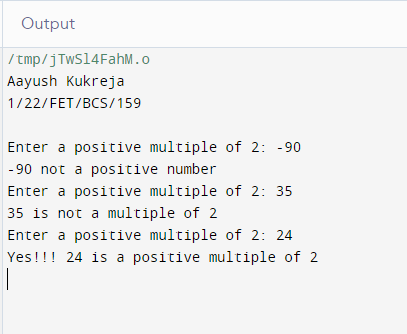
**Program 27**

**Aim:** Write a C++ program to throw multiple exception.

**Code:**

****

**Output:**

****

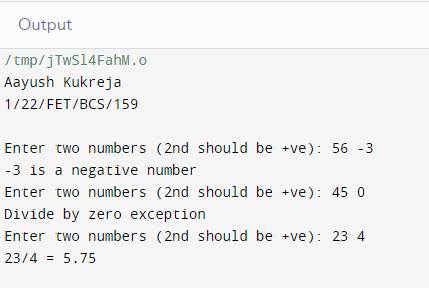
**Program 28**

**Aim:** Write a C++ program for re-throwing an exception.

**Code:**

****

**Output:**

****

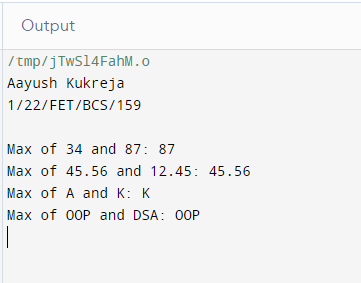
**Program 29**

**Aim:** Write a C++ program to create a function template.

**Code:**

****

**Output:**

****

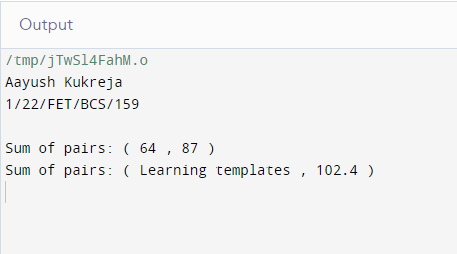
**Program 30**

**Aim:** Write a C++ program to create a class template.

**Code:**

****

**Output:**

****