## SAVEETHA SCHOOL OF ENGINEERING

## SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

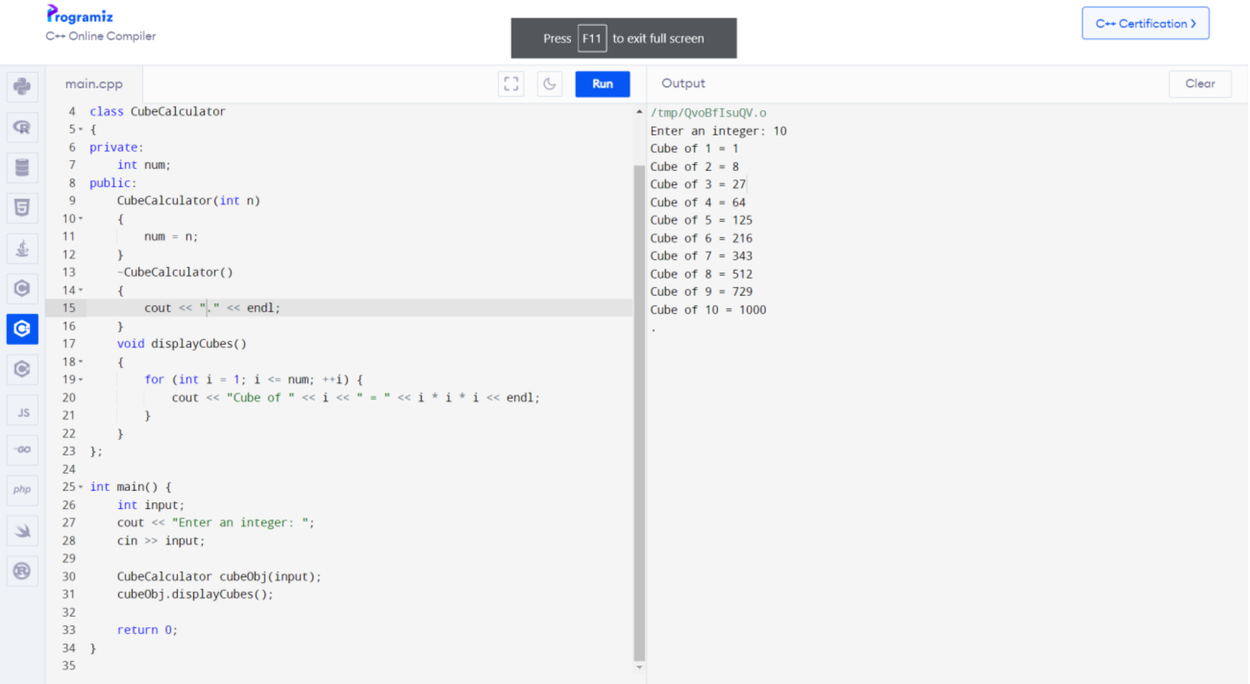
Engineer to Excel

**SUB CODE & NAME** : DSA01/ Object Oriented Programming with C++

## LAB DAY 3/18-03-2024

## EASY

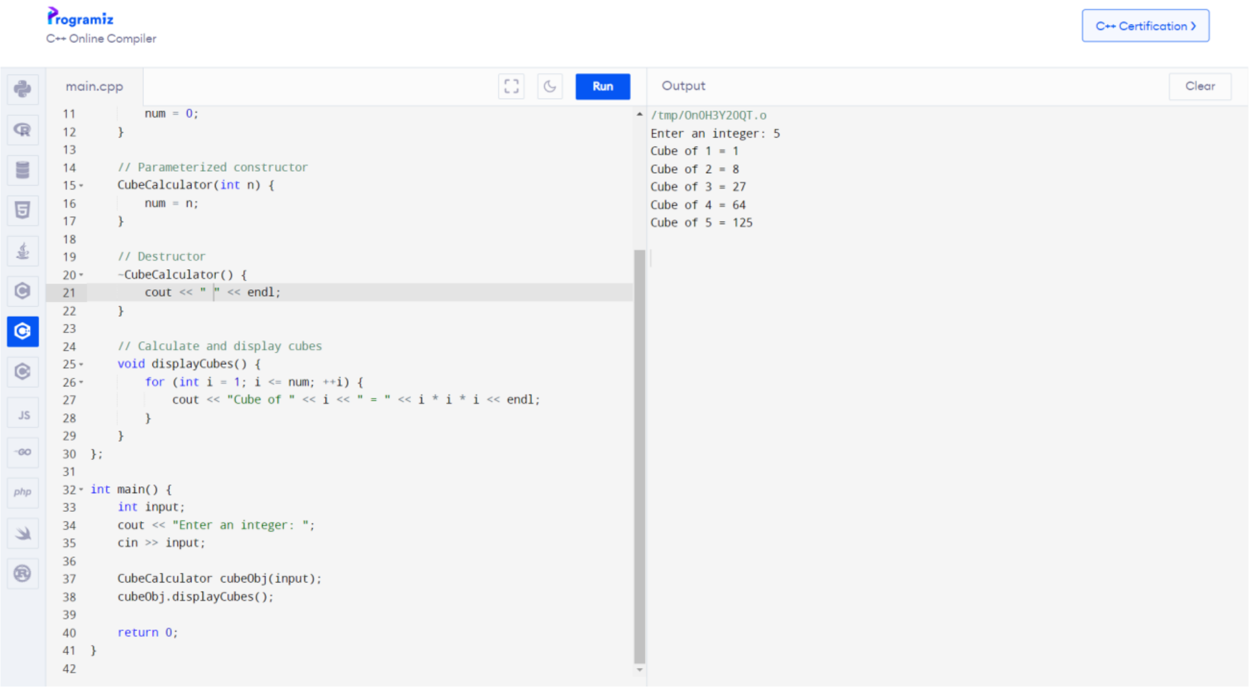
1. Write C++ Program to display the cube of the number up to a given integer using Destructor



```cpp
class CubeCalculator
{
private:
    int num;
public:
    CubeCalculator(int n)
    {
        num = n;
    }
    ~CubeCalculator()
    {
        cout << "." << endl;
    }
    void displayCubes()
    {
        for (int i = 1; i <= num; ++i) {
            cout << "Cube of " << i << " = " << i * i * i << endl;
        }
    }
};

int main() {
    int input;
    cout << "Enter an integer: ";
    cin >> input;

    CubeCalculator cubeObj(input);
    cubeObj.displayCubes();

    return 0;
}
```

```
/tmp/QvoBfIsuQV.o
Enter an integer: 10
Cube of 1 = 1
Cube of 2 = 8
Cube of 3 = 27
Cube of 4 = 64
Cube of 5 = 125
Cube of 6 = 216
Cube of 7 = 343
Cube of 8 = 512
Cube of 9 = 729
Cube of 10 = 1000
.
```

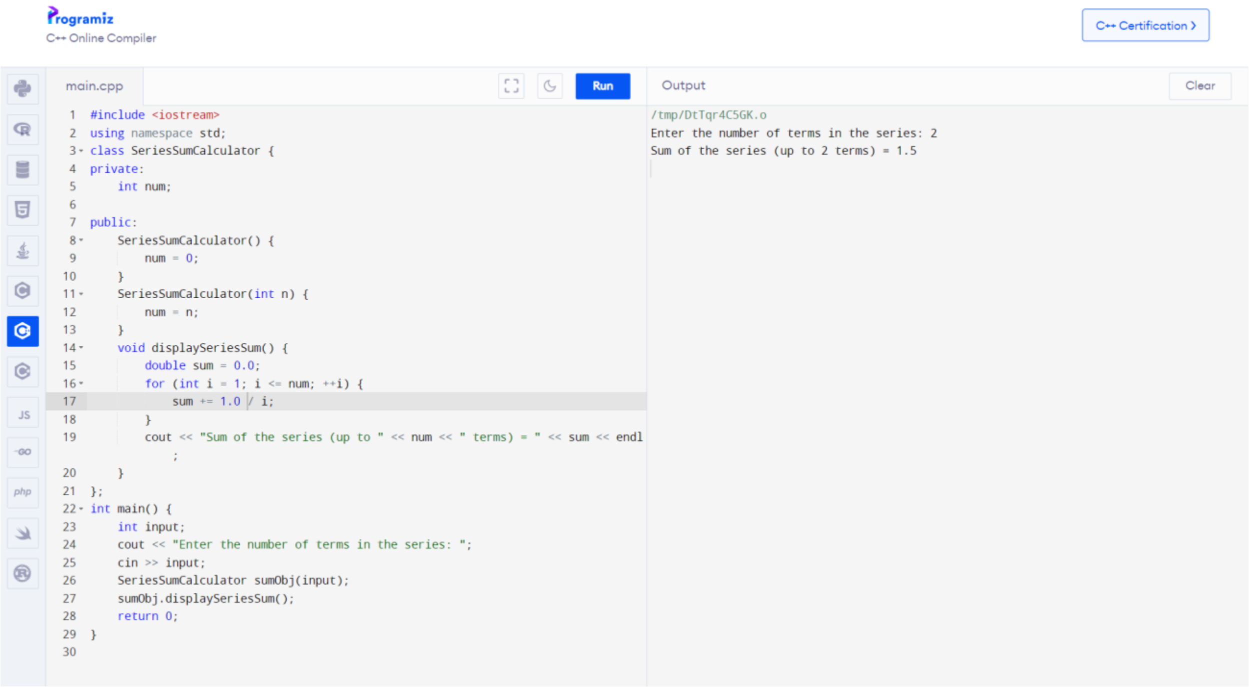
2. Write C++ Program to display the cube of the number up to a given integer using constructor Overloading



```cpp
        num = 0;
    }

    // Parameterized constructor
    CubeCalculator(int n) {
        num = n;
    }

    // Destructor
    ~CubeCalculator() {
        cout << " " << endl;
    }

    // Calculate and display cubes
    void displayCubes() {
        for (int i = 1; i <= num; ++i) {
            cout << "Cube of " << i << " = " << i * i * i << endl;
        }
    }
};

int main() {
    int input;
    cout << "Enter an integer: ";
    cin >> input;

    CubeCalculator cubeObj(input);
    cubeObj.displayCubes();

    return 0;
}
```

```
/tmp/On0H3Y20QT.o
Enter an integer: 5
Cube of 1 = 1
Cube of 2 = 8
Cube of 3 = 27
Cube of 4 = 64
Cube of 5 = 125
```

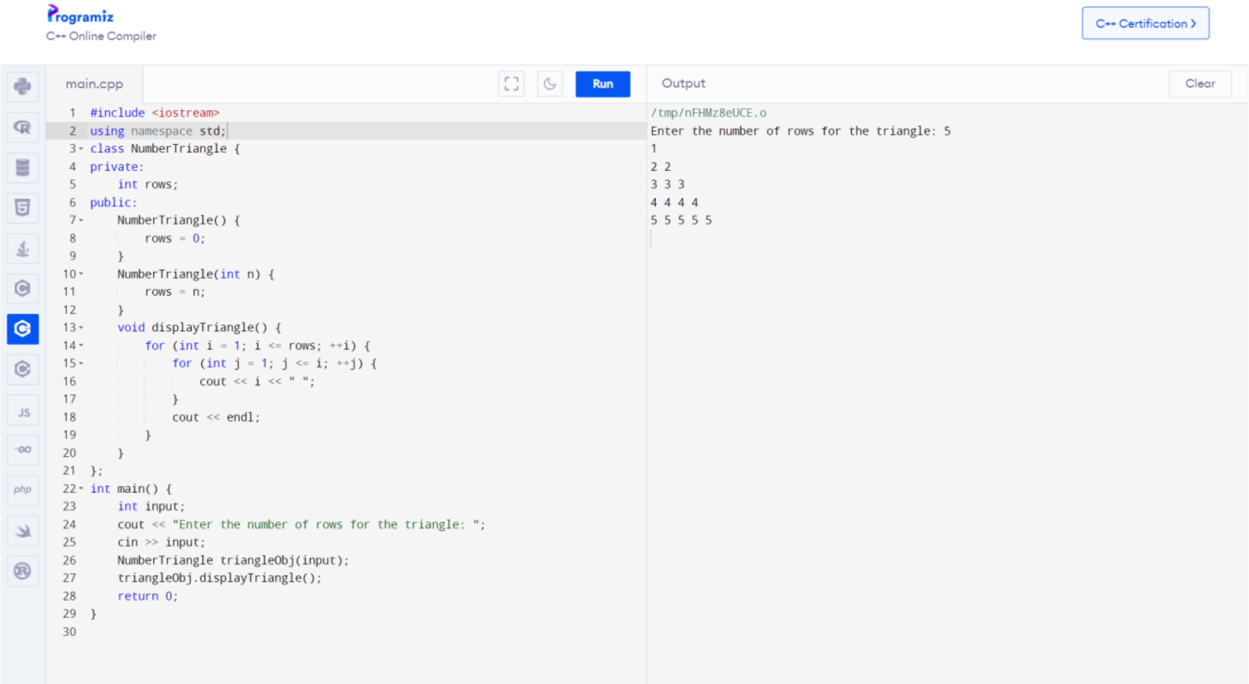
3. Write a program in C++ to find the sum of the series using the constructor overloading.



```cpp
#include <iostream>
using namespace std;
class SeriesSumCalculator {
private:
    int num;

public:
    SeriesSumCalculator() {
        num = 0;
    }
    SeriesSumCalculator(int n) {
        num = n;
    }
    void displaySeriesSum() {
        double sum = 0.0;
        for (int i = 1; i <= num; ++i) {
            sum += 1.0 / i;
        }
        cout << "Sum of the series (up to " << num << " terms) = " << sum << endl
            ;
    }
};
int main() {
    int input;
    cout << "Enter the number of terms in the series: ";
    cin >> input;
    SeriesSumCalculator sumObj(input);
    sumObj.displaySeriesSum();
    return 0;
}
```

```
/tmp/DtTqr4C5GK.o
Enter the number of terms in the series: 2
Sum of the series (up to 2 terms) = 1.5
```

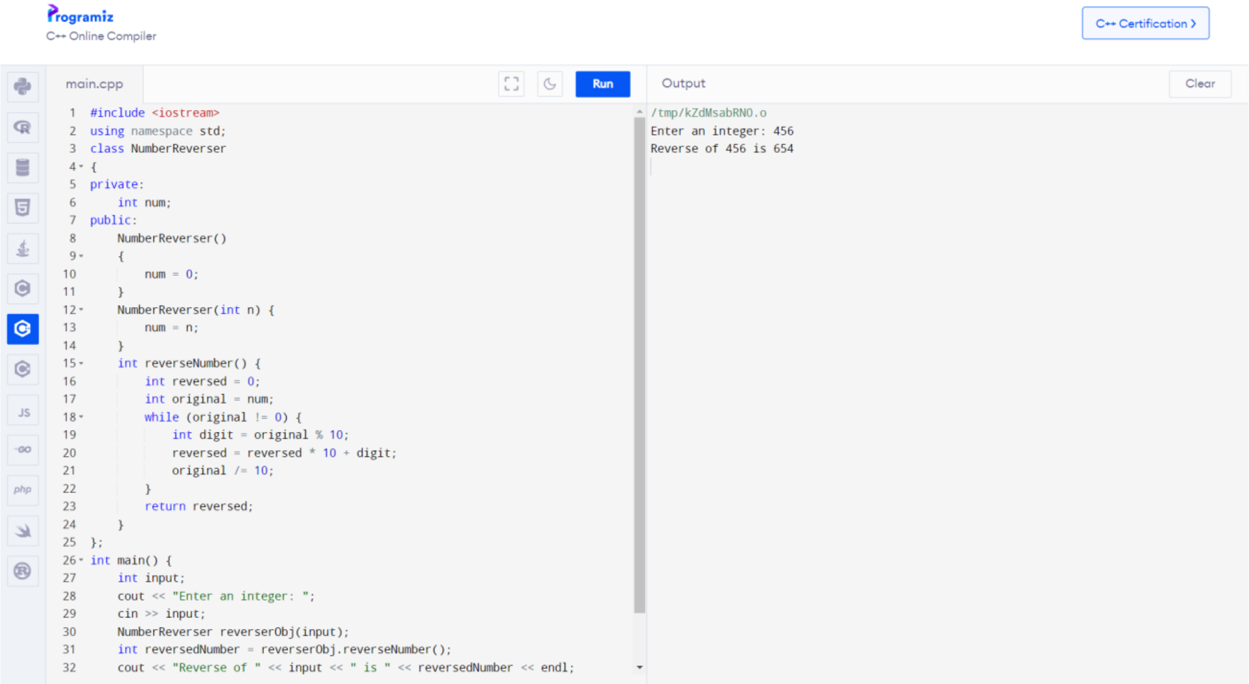
4. Write a program in C++ to print a pattern of right-angle triangle with a number that will repeat a number in the row by using the constructor overloading.



```cpp
#include <iostream>
using namespace std;
class NumberTriangle {
private:
    int rows;
public:
    NumberTriangle() {
        rows = 0;
    }
    NumberTriangle(int n) {
        rows = n;
    }
    void displayTriangle() {
        for (int i = 1; i <= rows; ++i) {
            for (int j = 1; j <= i; ++j) {
                cout << i << " ";
            }
            cout << endl;
        }
    }
};
int main() {
    int input;
    cout << "Enter the number of rows for the triangle: ";
    cin >> input;
    NumberTriangle triangleObj(input);
    triangleObj.displayTriangle();
    return 0;
}
```

```
/tmp/nFHMz8eUCE.o
Enter the number of rows for the triangle: 5
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

5. Write a C++ Program to display the reverse of a number using the constructor overloading

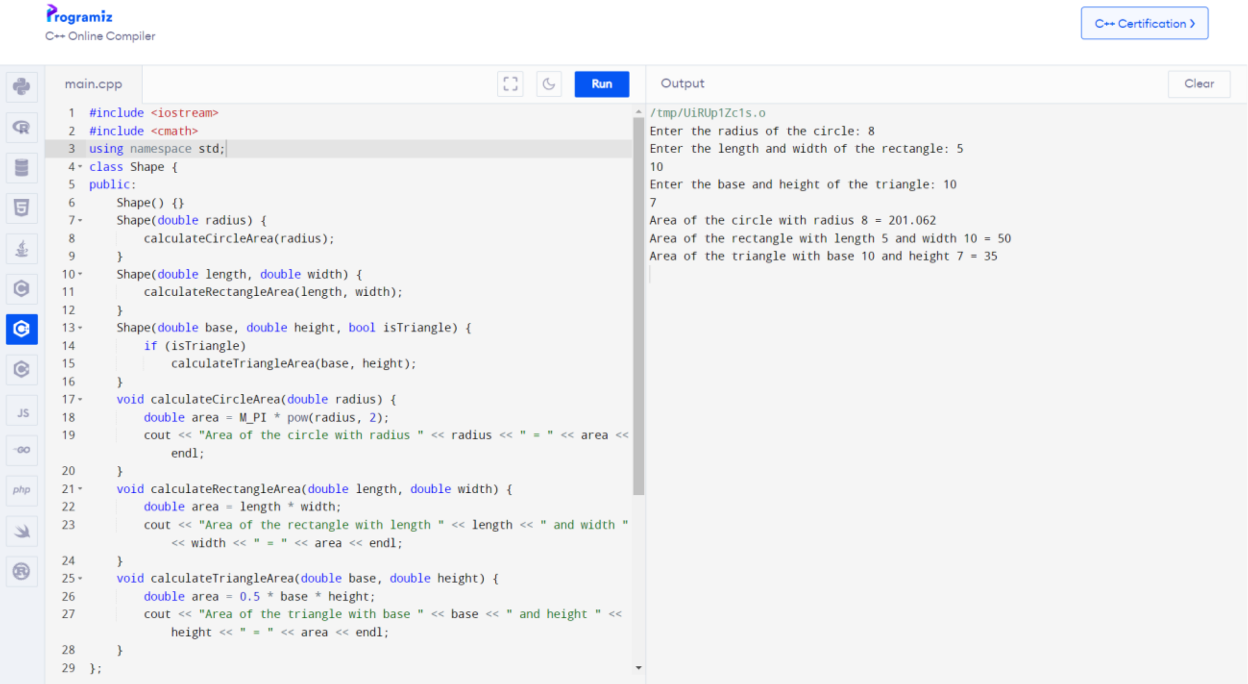


```cpp
#include <iostream>
using namespace std;
class NumberReverser
{
private:
    int num;
public:
    NumberReverser()
    {
        num = 0;
    }
    NumberReverser(int n) {
        num = n;
    }
    int reverseNumber() {
        int reversed = 0;
        int original = num;
        while (original != 0) {
            int digit = original % 10;
            reversed = reversed * 10 + digit;
            original /= 10;
        }
        return reversed;
    }
};
int main() {
    int input;
    cout << "Enter an integer: ";
    cin >> input;
    NumberReverser reverserObj(input);
    int reversedNumber = reverserObj.reverseNumber();
    cout << "Reverse of " << input << " is " << reversedNumber << endl;
```

```
/tmp/kZdMsabRN0.o
Enter an integer: 456
Reverse of 456 is 654
```

# MEDIUM

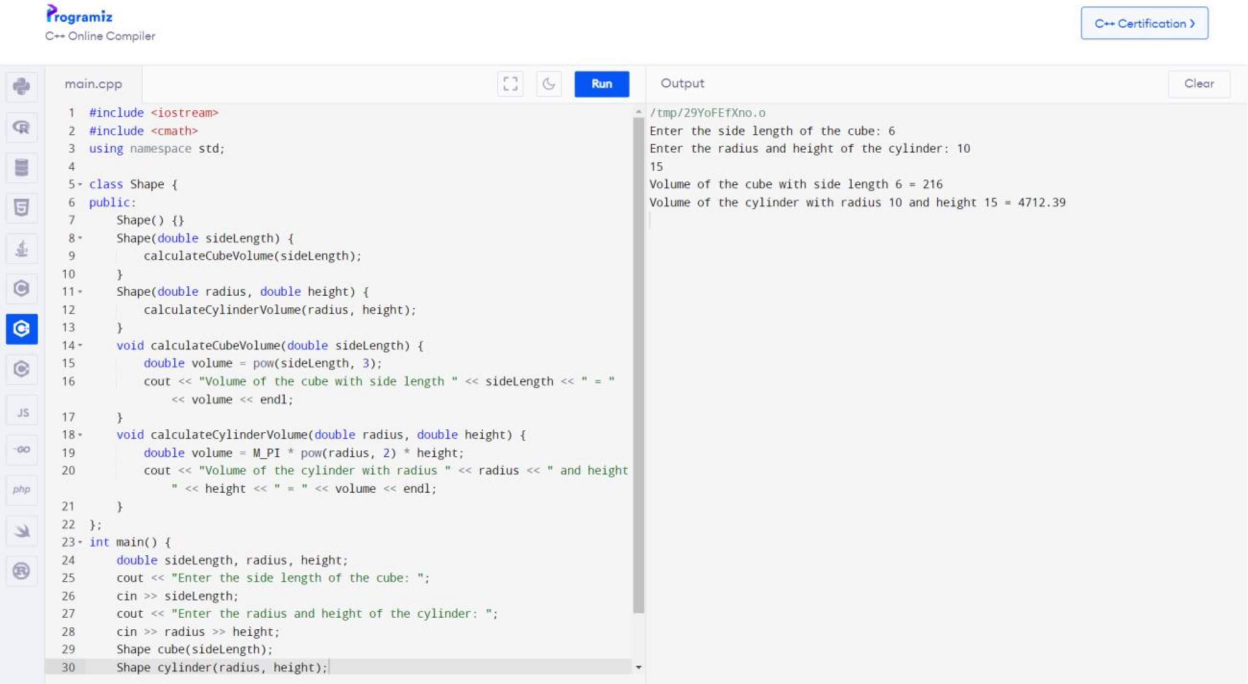
1. Write a program to find area of Circle, Rectangle and Triangle using constructor overloading.



```cpp
#include <iostream>
#include <cmath>
using namespace std;
class Shape {
public:
    Shape() {}
    Shape(double radius) {
        calculateCircleArea(radius);
    }
    Shape(double length, double width) {
        calculateRectangleArea(length, width);
    }
    Shape(double base, double height, bool isTriangle) {
        if (isTriangle)
            calculateTriangleArea(base, height);
    }
    void calculateCircleArea(double radius) {
        double area = M_PI * pow(radius, 2);
        cout << "Area of the circle with radius " << radius << " = " << area <<
            endl;
    }
    void calculateRectangleArea(double length, double width) {
        double area = length * width;
        cout << "Area of the rectangle with length " << length << " and width "
            << width << " = " << area << endl;
    }
    void calculateTriangleArea(double base, double height) {
        double area = 0.5 * base * height;
        cout << "Area of the triangle with base " << base << " and height " <<
            height << " = " << area << endl;
    }
};
```

```
/tmp/UiRUp1Zc1s.o
Enter the radius of the circle: 8
Enter the length and width of the rectangle: 5
10
Enter the base and height of the triangle: 10
7
Area of the circle with radius 8 = 201.062
Area of the rectangle with length 5 and width 10 = 50
Area of the triangle with base 10 and height 7 = 35
```

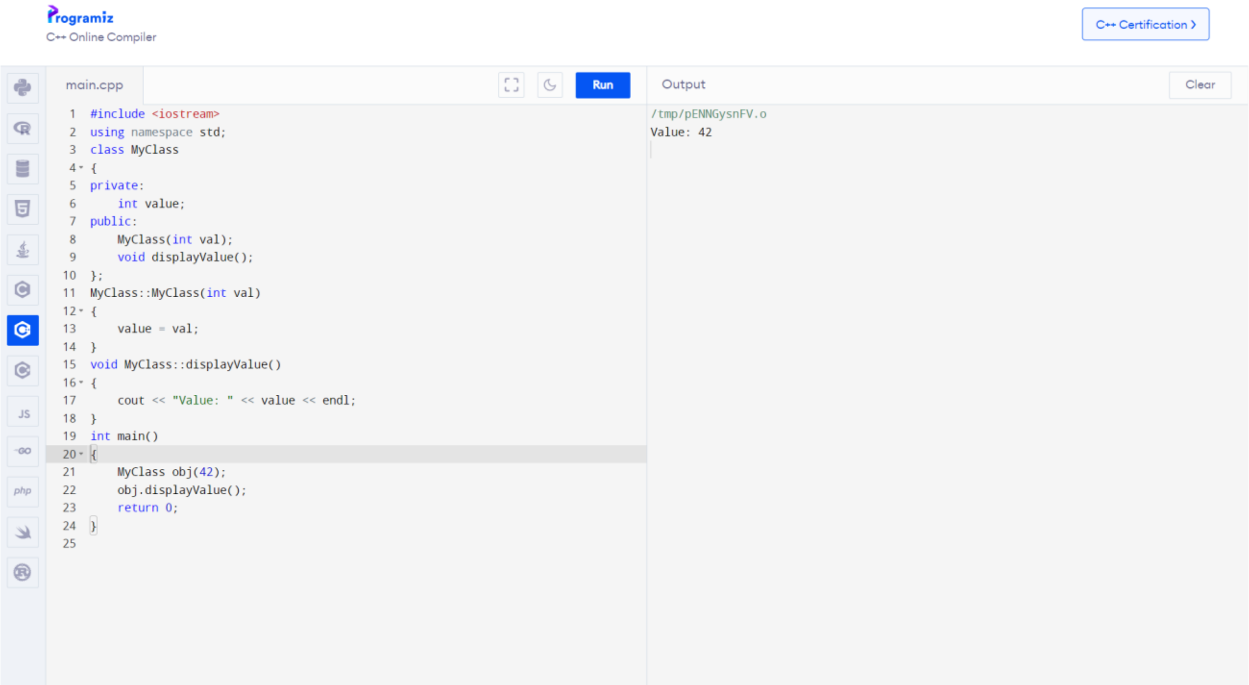
2. Write a program to find Cube, Cylinder using constructor overloading



```cpp
#include <iostream>
#include <cmath>
using namespace std;

class Shape {
public:
    Shape() {}
    Shape(double sideLength) {
        calculateCubeVolume(sideLength);
    }
    Shape(double radius, double height) {
        calculateCylinderVolume(radius, height);
    }
    void calculateCubeVolume(double sideLength) {
        double volume = pow(sideLength, 3);
        cout << "Volume of the cube with side length " << sideLength << " = "
            << volume << endl;
    }
    void calculateCylinderVolume(double radius, double height) {
        double volume = M_PI * pow(radius, 2) * height;
        cout << "Volume of the cylinder with radius " << radius << " and height
            " << height << " = " << volume << endl;
    }
};
int main() {
    double sideLength, radius, height;
    cout << "Enter the side length of the cube: ";
    cin >> sideLength;
    cout << "Enter the radius and height of the cylinder: ";
    cin >> radius >> height;
    Shape cube(sideLength);
    Shape cylinder(radius, height);
```

```
/tmp/29YoFEfXno.o
Enter the side length of the cube: 6
Enter the radius and height of the cylinder: 10
15
Volume of the cube with side length 6 = 216
Volume of the cylinder with radius 10 and height 15 = 4712.39
```

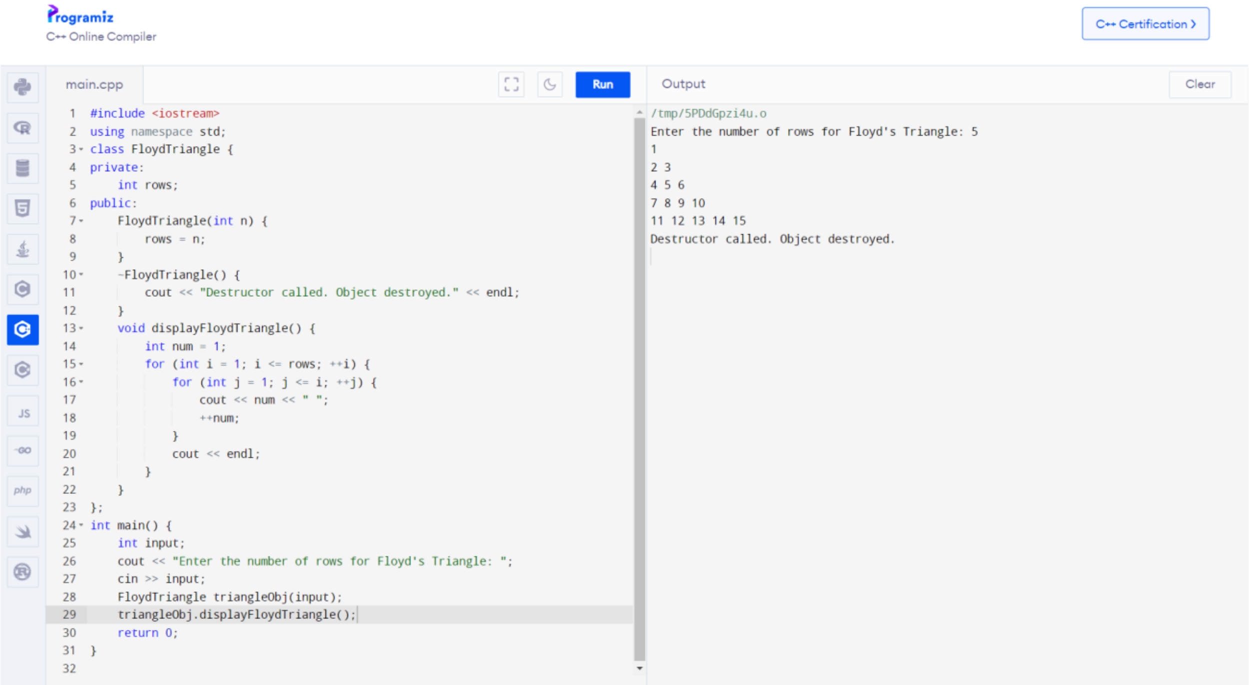
3. Write a program to declare the constructor inside the class, and then define it outside of the class by specifying the name of the class.



```cpp
#include <iostream>
using namespace std;
class MyClass
{
private:
    int value;
public:
    MyClass(int val);
    void displayValue();
};
MyClass::MyClass(int val)
{
    value = val;
}
void MyClass::displayValue()
{
    cout << "Value: " << value << endl;
}
int main()
{
    MyClass obj(42);
    obj.displayValue();
    return 0;
}
```

```
/tmp/pENNGysnFV.o
Value: 42
```

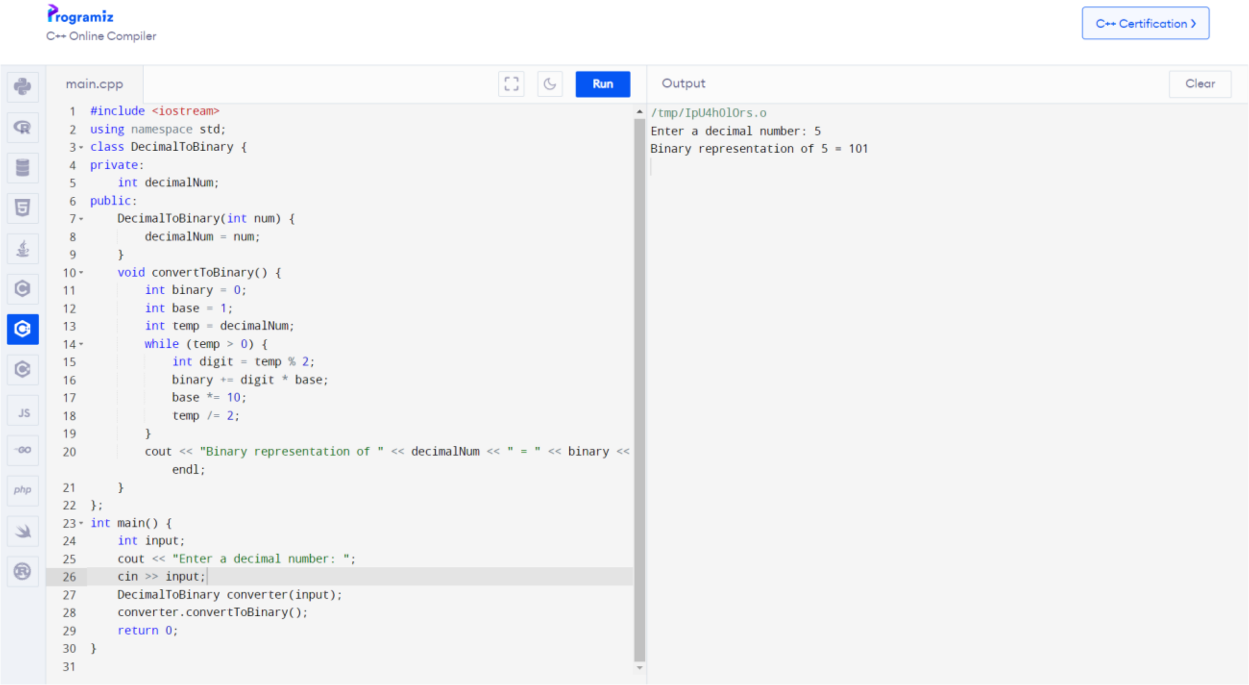
4. Write a program in C++ to print Floyd's Triangle by using the constructor destructor



```cpp
#include <iostream>
using namespace std;
class FloydTriangle {
private:
    int rows;
public:
    FloydTriangle(int n) {
        rows = n;
    }
    ~FloydTriangle() {
        cout << "Destructor called. Object destroyed." << endl;
    }
    void displayFloydTriangle() {
        int num = 1;
        for (int i = 1; i <= rows; ++i) {
            for (int j = 1; j <= i; ++j) {
                cout << num << " ";
                ++num;
            }
            cout << endl;
        }
    }
};
int main() {
    int input;
    cout << "Enter the number of rows for Floyd's Triangle: ";
    cin >> input;
    FloydTriangle triangleObj(input);
    triangleObj.displayFloydTriangle();
    return 0;
}
```

```
/tmp/5PDdGpzi4u.o
Enter the number of rows for Floyd's Triangle: 5
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
Destructor called. Object destroyed.
```

5. Write a program in C++ to convert a decimal number into binary without using an array by using the constructor overloading.

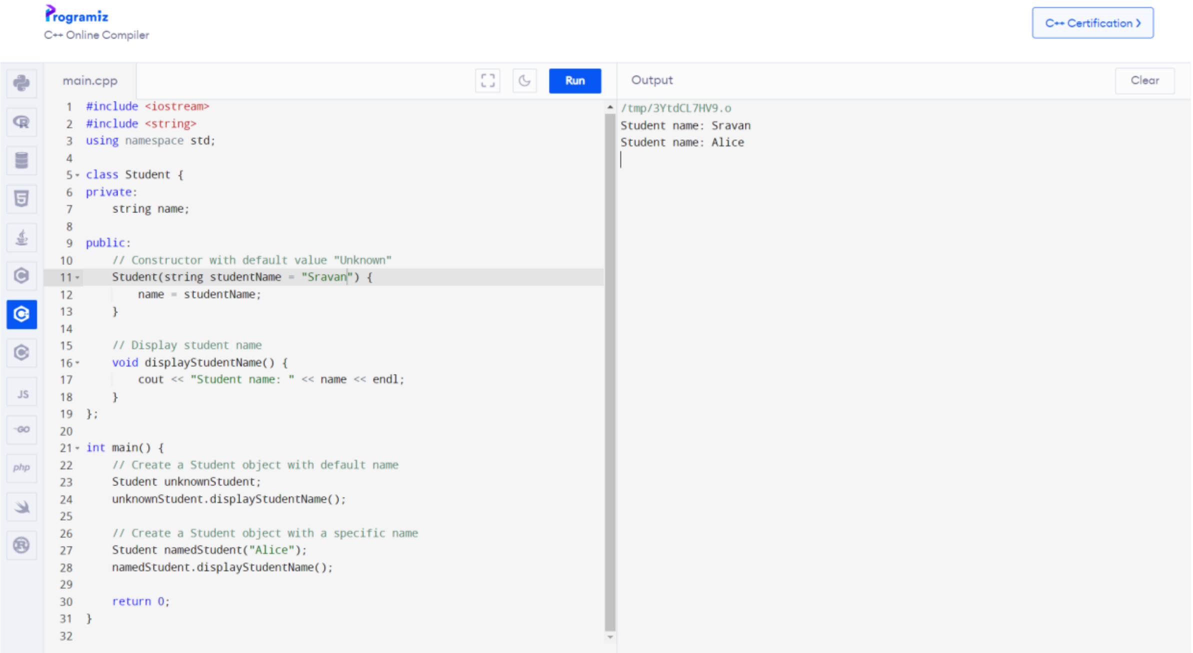


```cpp
#include <iostream>
using namespace std;
class DecimalToBinary {
private:
    int decimalNum;
public:
    DecimalToBinary(int num) {
        decimalNum = num;
    }
    void convertToBinary() {
        int binary = 0;
        int base = 1;
        int temp = decimalNum;
        while (temp > 0) {
            int digit = temp % 2;
            binary += digit * base;
            base *= 10;
            temp /= 2;
        }
        cout << "Binary representation of " << decimalNum << " = " << binary << endl;
    }
};
int main() {
    int input;
    cout << "Enter a decimal number: ";
    cin >> input;
    DecimalToBinary converter(input);
    converter.convertToBinary();
    return 0;
}
```

```
/tmp/IpU4h0lOrs.o
Enter a decimal number: 5
Binary representation of 5 = 101
```

## HARD

1. Write a program to print the names of students by creating a Student class. If no name is passed while creating an object of the Student class, then the name should be "Unknown", otherwise the name should be equal to the String value passed while creating the object of the Student class.



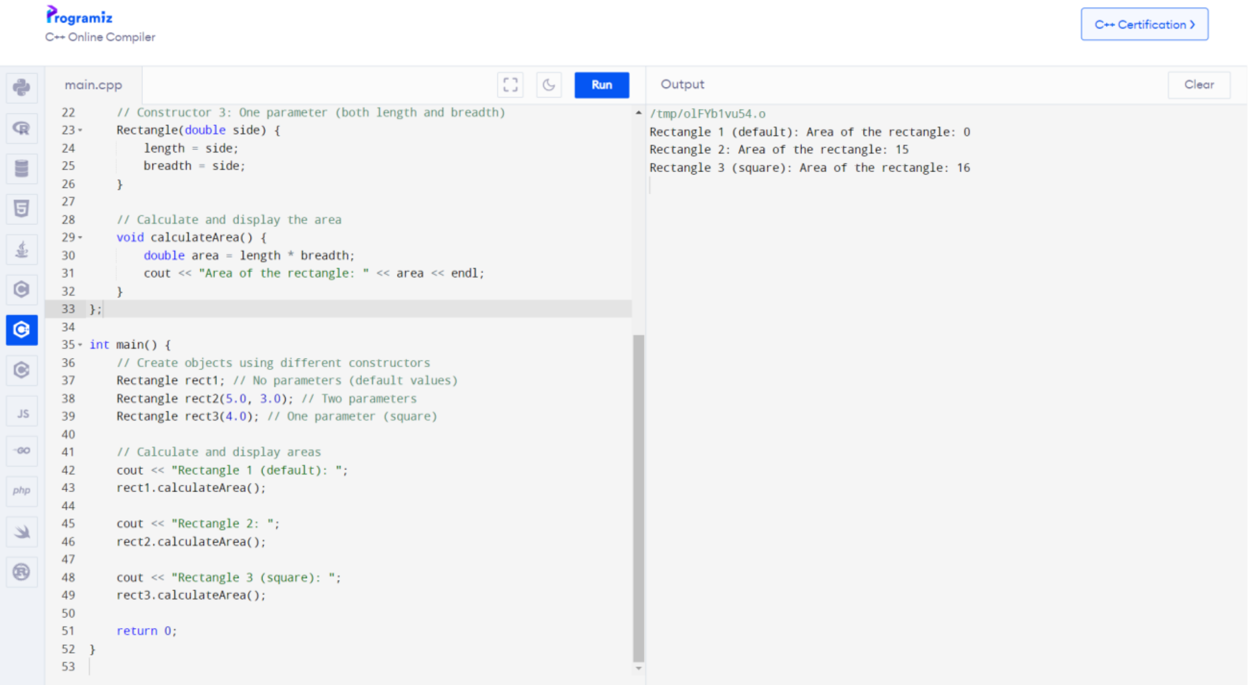
```cpp
#include <iostream>
#include <string>
using namespace std;

class Student {
private:
    string name;

public:
    // Constructor with default value "Unknown"
    Student(string studentName = "Sravan") {
        name = studentName;
    }

    // Display student name
    void displayStudentName() {
        cout << "Student name: " << name << endl;
    }
};

int main() {
    // Create a Student object with default name
    Student unknownStudent;
    unknownStudent.displayStudentName();

    // Create a Student object with a specific name
    Student namedStudent("Alice");
    namedStudent.displayStudentName();

    return 0;
}
```

```
/tmp/3YtdCL7HV9.o
Student name: Sravan
Student name: Alice
```

2. Create a class named 'Rectangle' with two data members- length and breadth and a function to calculate the area which is 'length*breadth'. The class has three constructors which are:

a) 1 - Having no parameter - values of both length and breadth are assigned zero.
b) 2 - Having two numbers as parameters - the two numbers are assigned as length and breadth respectively.
c) 3 - Having one number as parameter - both length and breadth are assigned that number.

Now, create objects of the 'Rectangle' class having none, one and two parameters and print their areas.



```
    // Constructor 3: One parameter (both length and breadth)
    Rectangle(double side) {
        length = side;
        breadth = side;
    }

    // Calculate and display the area
    void calculateArea() {
        double area = length * breadth;
        cout << "Area of the rectangle: " << area << endl;
    }
};

int main() {
    // Create objects using different constructors
    Rectangle rect1; // No parameters (default values)
    Rectangle rect2(5.0, 3.0); // Two parameters
    Rectangle rect3(4.0); // One parameter (square)

    // Calculate and display areas
    cout << "Rectangle 1 (default): ";
    rect1.calculateArea();

    cout << "Rectangle 2: ";
    rect2.calculateArea();

    cout << "Rectangle 3 (square): ";
    rect3.calculateArea();

    return 0;
}
```

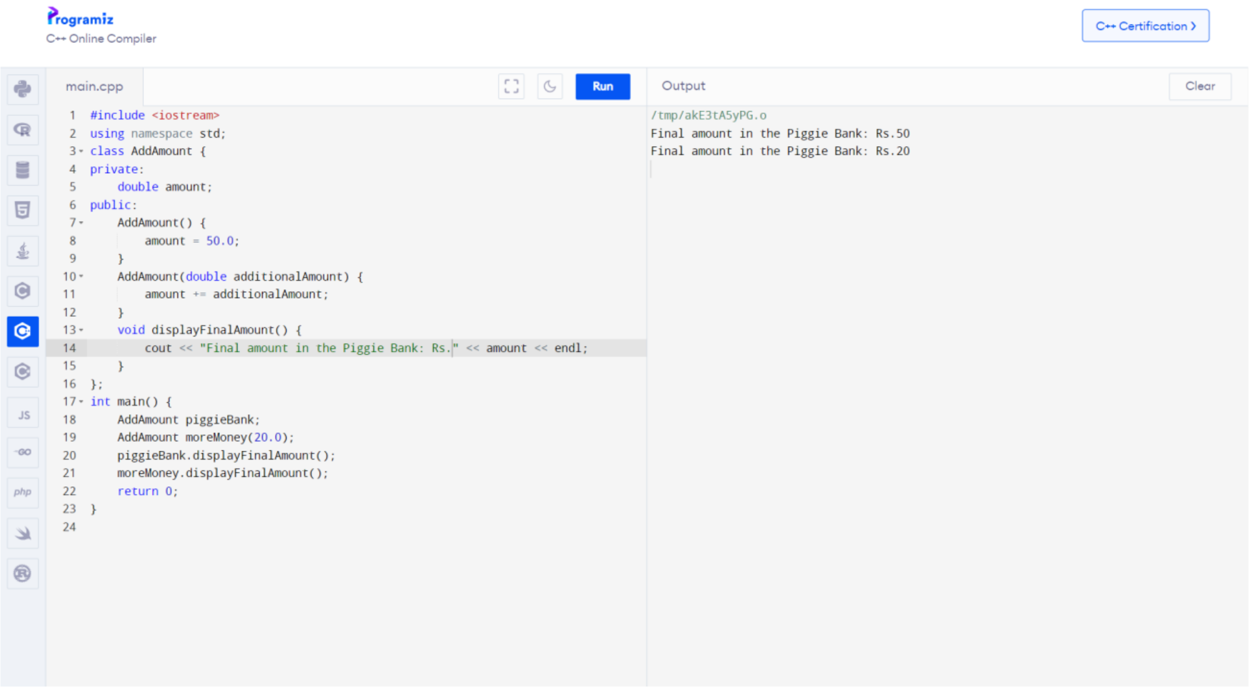
```
/tmp/olFYb1vu54.o
Rectangle 1 (default): Area of the rectangle: 0
Rectangle 2: Area of the rectangle: 15
Rectangle 3 (square): Area of the rectangle: 16
```

3.Suppose you have a Piggie Bank with an initial amount of $50 and you have to add some more amounts to it. Create a class 'AddAmount' with a data member named 'amount' with an initial value of $50. Now make two constructors of this class as follows:

1 - without any parameter - no amount will be added to the Piggie Bank

2 - having a parameter which is the amount that will be added to the Piggie Bank

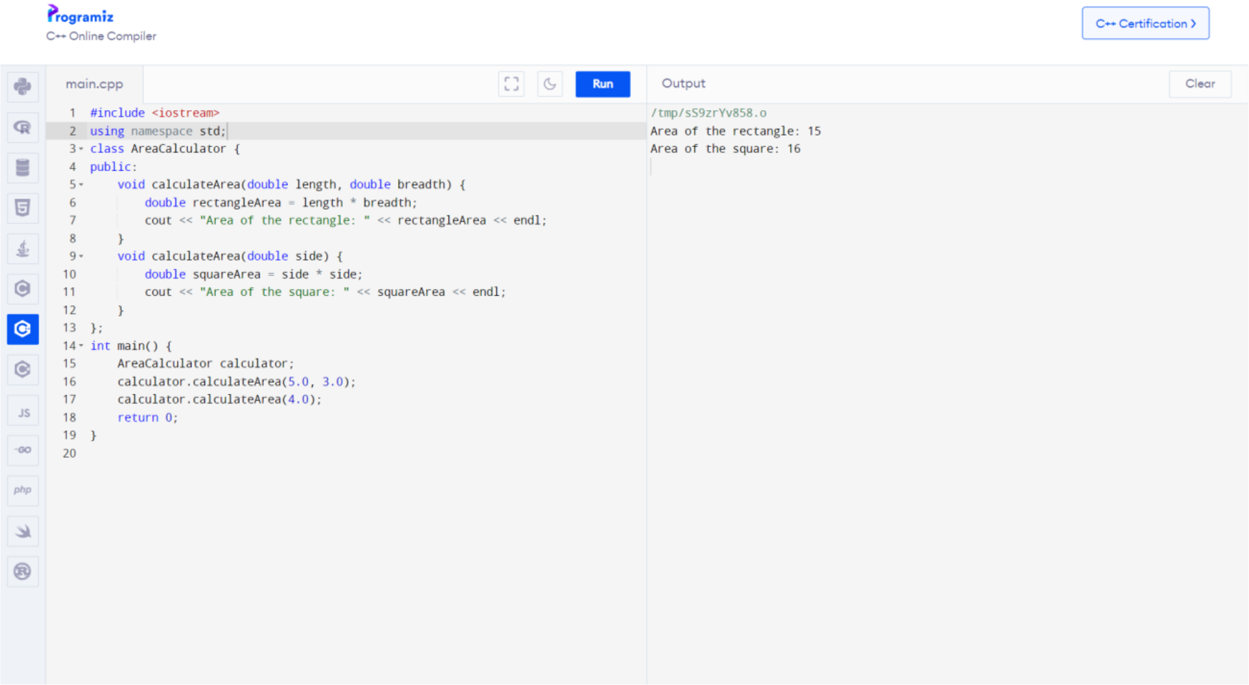
Create an object of the 'AddAmount' class and display the final amount in the Piggie Bank.



```cpp
#include <iostream>
using namespace std;
class AddAmount {
private:
    double amount;
public:
    AddAmount() {
        amount = 50.0;
    }
    AddAmount(double additionalAmount) {
        amount += additionalAmount;
    }
    void displayFinalAmount() {
        cout << "Final amount in the Piggie Bank: Rs." << amount << endl;
    }
};
int main() {
    AddAmount piggieBank;
    AddAmount moreMoney(20.0);
    piggieBank.displayFinalAmount();
    moreMoney.displayFinalAmount();
    return 0;
}
```

```
/tmp/akE3tA5yPG.o
Final amount in the Piggie Bank: Rs.50
Final amount in the Piggie Bank: Rs.20
```

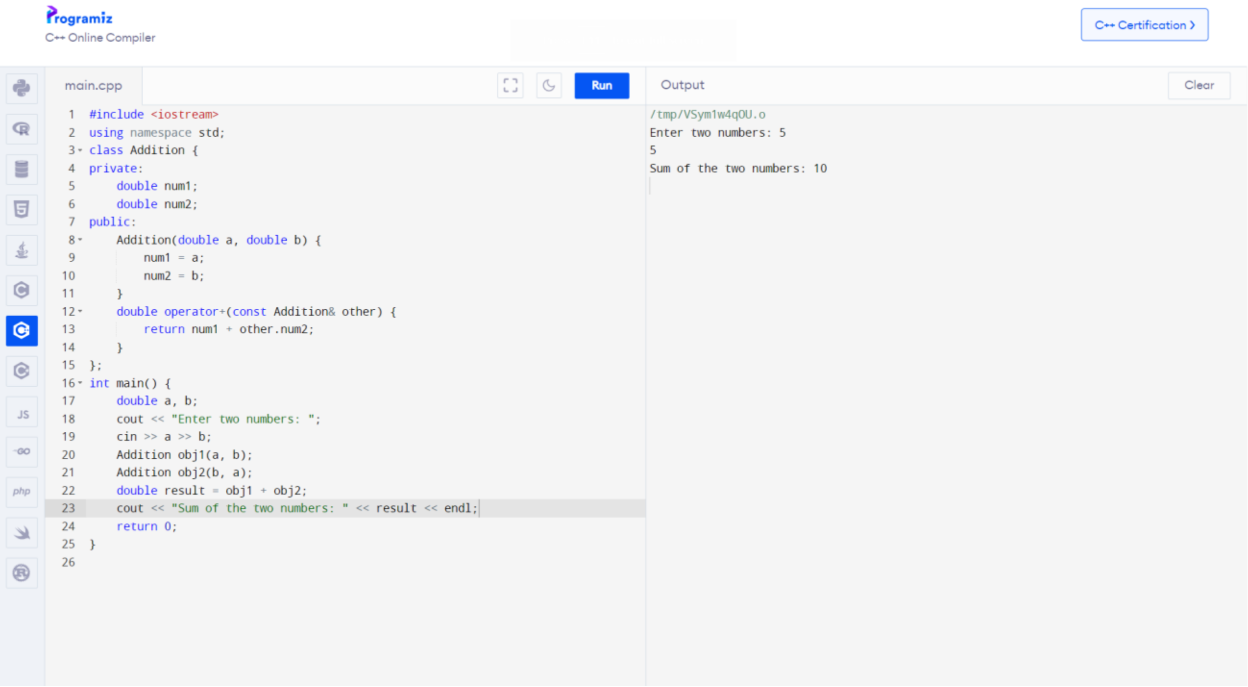
4. Create a class to print the area of a square and a rectangle. The class has two functions with the same name but different number of parameters. The function for printing the area of rectangle has two parameters which are its length and breadth respectively while the other function for printing the area of square has one parameter which is the side of the square.



```cpp
#include <iostream>
using namespace std;
class AreaCalculator {
public:
    void calculateArea(double length, double breadth) {
        double rectangleArea = length * breadth;
        cout << "Area of the rectangle: " << rectangleArea << endl;
    }
    void calculateArea(double side) {
        double squareArea = side * side;
        cout << "Area of the square: " << squareArea << endl;
    }
};
int main() {
    AreaCalculator calculator;
    calculator.calculateArea(5.0, 3.0);
    calculator.calculateArea(4.0);
    return 0;
}
```

```
/tmp/sS9zrYv858.o
Area of the rectangle: 15
Area of the square: 16
```

5. Create a class called add for addition of two numbers using operator overloading



```cpp
#include <iostream>
using namespace std;
class Addition {
private:
    double num1;
    double num2;
public:
    Addition(double a, double b) {
        num1 = a;
        num2 = b;
    }
    double operator+(const Addition& other) {
        return num1 + other.num2;
    }
};
int main() {
    double a, b;
    cout << "Enter two numbers: ";
    cin >> a >> b;
    Addition obj1(a, b);
    Addition obj2(b, a);
    double result = obj1 + obj2;
    cout << "Sum of the two numbers: " << result << endl;
    return 0;
}
```

```
/tmp/VSym1w4qOU.o
Enter two numbers: 5
5
Sum of the two numbers: 10
```