Problem Statement: Shape Area Calculator Using Method Overloading

You are required to implement a program that calculates the area of different shapes using compile-time polymorphism (method overloading) in C++. The program should support calculation of areas for the following shapes:

Rectangle

Circle

Triangle

Requirements:

Shape Class: Implement a Shape class as a base class with virtual functions to calculate and display the area of each shape.

Derived Classes: Implement derived classes Rectangle, Circle, and Triangle, inheriting from Shape, each with overridden functions to calculate and display their respective areas.

Method Overloading: Use method overloading in the Shape class to define multiple calculateArea functions, each specific to one shape.

Input and Output: Implement a main() function to test the implemented classes by creating instances of each shape, inputting dimensions, and displaying their calculated areas.

#include <iostream>

#include <cmath>

// Base class Shape

class Shape {

public:

virtual double calculateArea() const = 0; // Pure virtual function to calculate area

virtual void displayArea() const = 0; // Pure virtual function to display area

virtual ~Shape() {} // Virtual destructor

};

// Derived class Rectangle

class Rectangle : public Shape {

private:

double length;

double width;

public:

Rectangle(double l, double w) : length(l), width(w) {}

double calculateArea() const override {

return length \* width;

}

void displayArea() const override {

std::cout << "Rectangle Area: " << calculateArea() << std::endl;

}

};

// Derived class Circle

class Circle : public Shape {

private:

double radius;

public:

Circle(double r) : radius(r) {}

double calculateArea() const override {

return M\_PI \* radius \* radius;

}

void displayArea() const override {

std::cout << "Circle Area: " << calculateArea() << std::endl;

}

};

// Derived class Triangle

class Triangle : public Shape {

private:

double base;

double height;

public:

Triangle(double b, double h) : base(b), height(h) {}

double calculateArea() const override {

return 0.5 \* base \* height;

}

void displayArea() const override {

std::cout << "Triangle Area: " << calculateArea() << std::endl;

}

};

int main() {

// Creating instances of each shape and inputting dimensions

Rectangle rect(10.0, 5.0);

Circle circ(7.0);

Triangle tri(6.0, 8.0);

// Displaying the calculated areas

rect.displayArea();

circ.displayArea();

tri.displayArea();

return 0;

}

Output:

