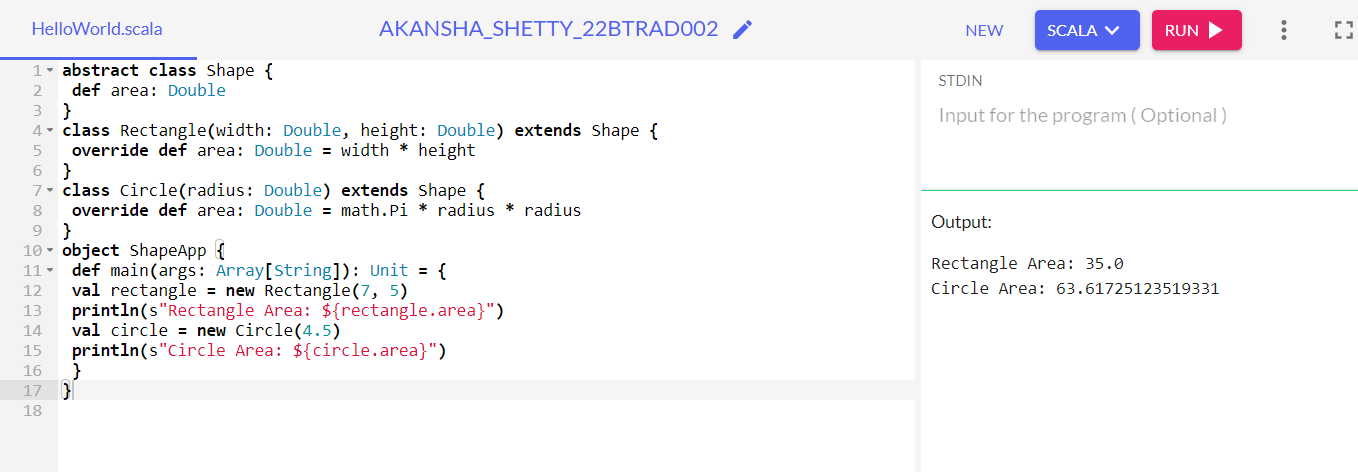
**NAME: AKANSHA SHETTY USN: 22BTRAD002 COURSE: AI/DE SECTION: A**

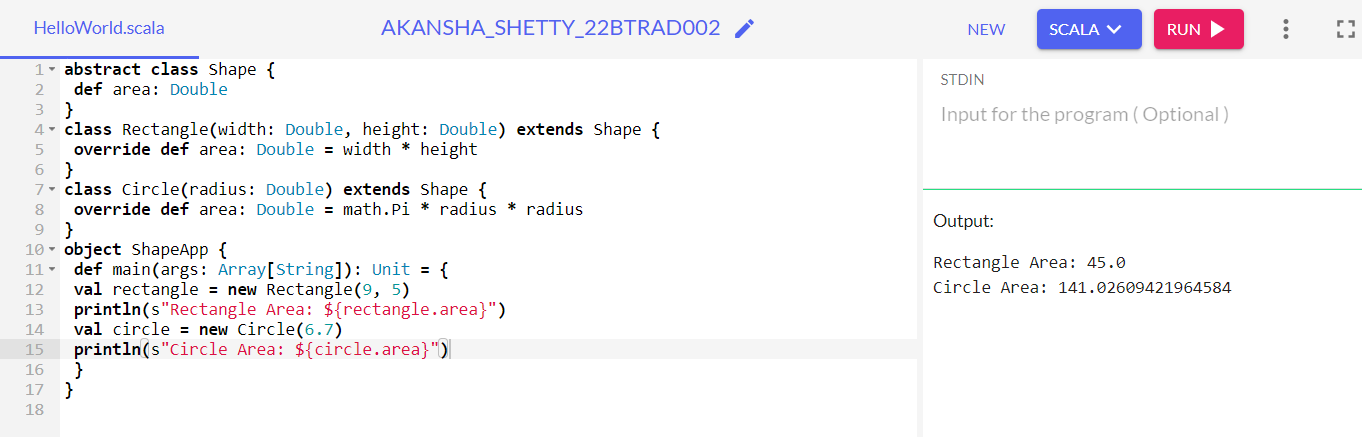
**DATE: 17/09/2023**

**GITHUB:** <https://github.com/Akansha-S1/SCALA_PROGRAMMING>

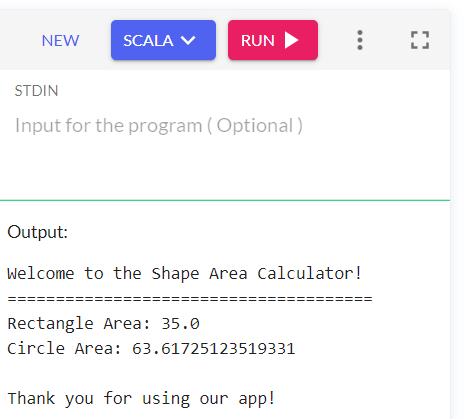
**Problem statement : 14 : Write a Scala program that creates an abstract class Shape with an abstract method area. Implement subclasses Rectangle and Circle that override the area method.**

**TEST CASE 1:** Use given data:



**TEST CASE 2:** Update with your own data:

**TEST CASE 3:** Make it more user friendly:

**OUTPUT:**

**WHAT ARE THE CHANGES:**

* **Have added a welcoming message to greet the user.**
* **Included a header to introduce the application.**
* **Changed the method name to make it more understandable: method name calculateArea in the Shape class.**
* **Provided a "Thank you" message at the end to indicate the end of the program.**

**CODE:**

abstract class Shape {

def calculateArea: Double

}

class Rectangle(width: Double, height: Double) extends Shape {

override def calculateArea: Double = width \* height

}

class Circle(radius: Double) extends Shape {

override def calculateArea: Double = math.Pi \* radius \* radius

}

object ShapeApp {

def main(args: Array[String]): Unit = {

println("Welcome to the Shape Area Calculator!")

println("======================================")

val rectangle = new Rectangle(7, 5)

println(s"Rectangle Area: ${rectangle.calculateArea}")

val circle = new Circle(4.5)

println(s"Circle Area: ${circle.calculateArea}")

println("\nThank you for using our app!")

}

}

**Explanation:**

* Abstract Class Shape:
* Shape is like a blueprint for various shapes.
* It defines a crucial requirement: any shape must be able to calculate its own area.
* The area method, declared in Shape, is a placeholder for this calculation and it returns a Double representing the shape's area.
* Class Rectangle (Extending Shape):
* Rectangle is a specific type of shape that represents, well, rectangles!
* Since it's a kind of shape, it extends the Shape abstract class, which means it must follow the rule and implement the area method.
* In the Rectangle class, we specify how to calculate the area for a rectangle: multiply its width and height.
* Class Circle (Extending Shape):
* Circle is another specific type of shape, this time for circles.
* Like the Rectangle class, it also extends the Shape abstract class and adheres to the rule by implementing the area method.
* In the Circle class, we calculate the area of a circle using the formula π \* radius^2 (Pi times the square of the radius).
* ShapeApp Object and main Method:
* The ShapeApp object serves as the starting point of our program, where everything begins.
* Inside its main method, we bring our shapes to life.
* We create a specific Rectangle with a width of 7 and a height of 5, then calculate its area.
* We then created a Circle with a radius of 4.5 and find its area.
* And then we calculated both the areas.