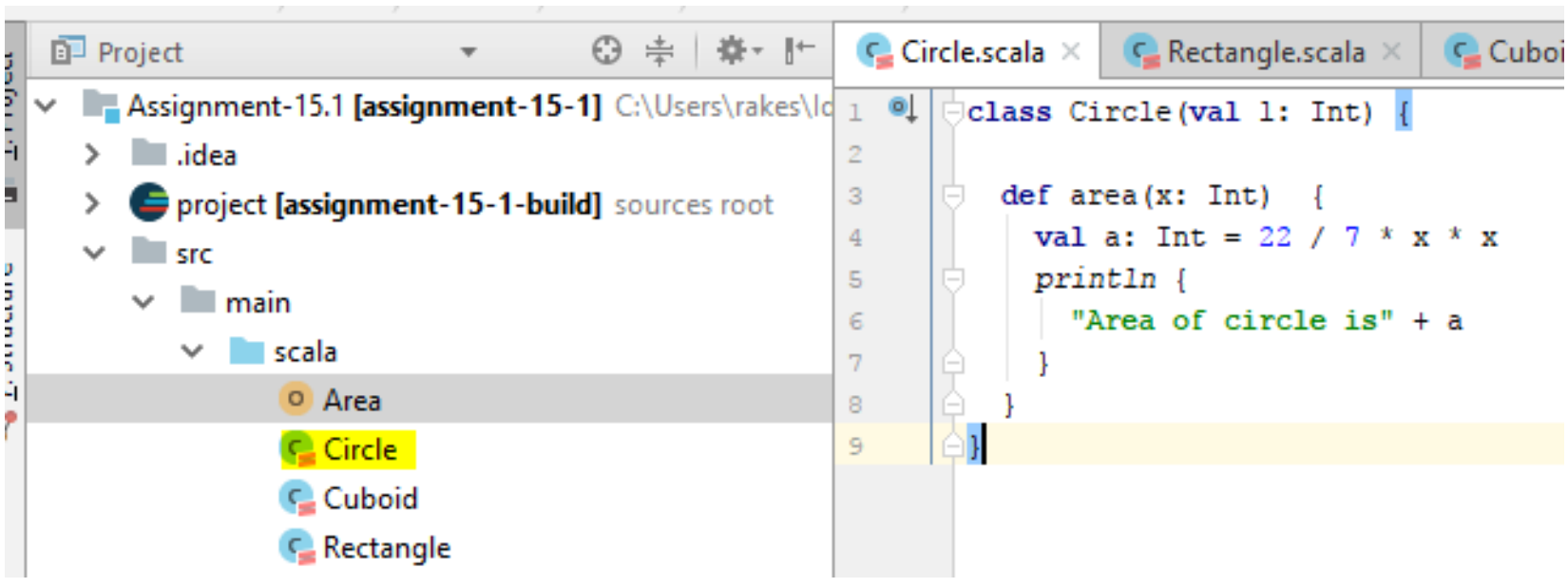
Problem Statement-

1. Write a simple program to show inheritance in scala.

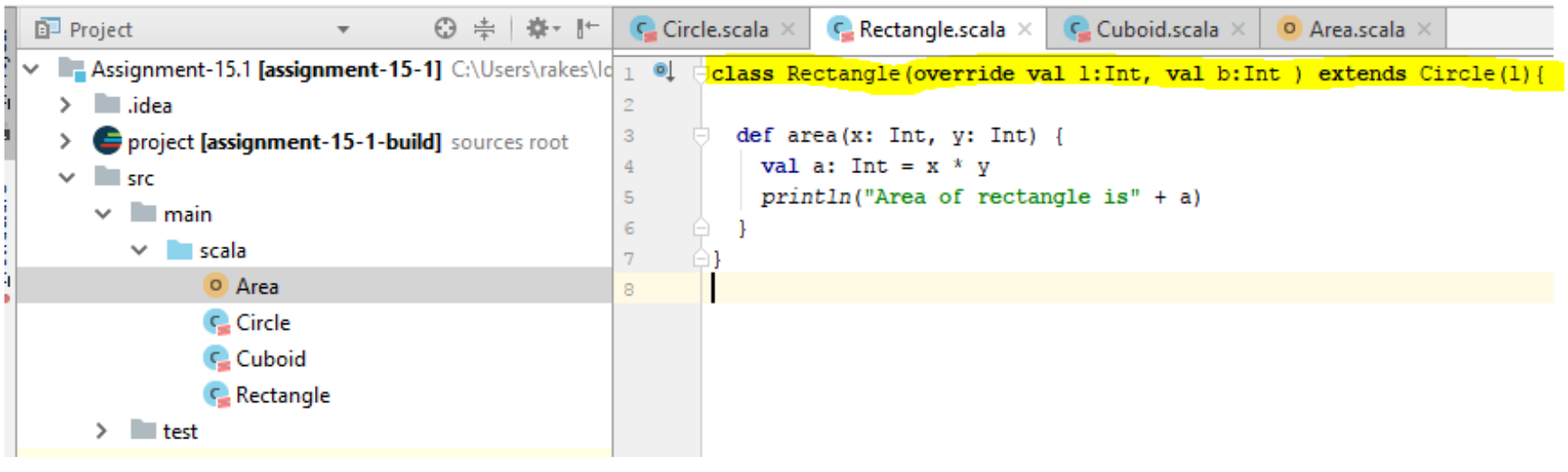
Solution-

To demonstrate simple inheritance in Scala we have used below 3 classes namely Circle, Rectangle and Cuboid to calculate area and volume of cuboid-

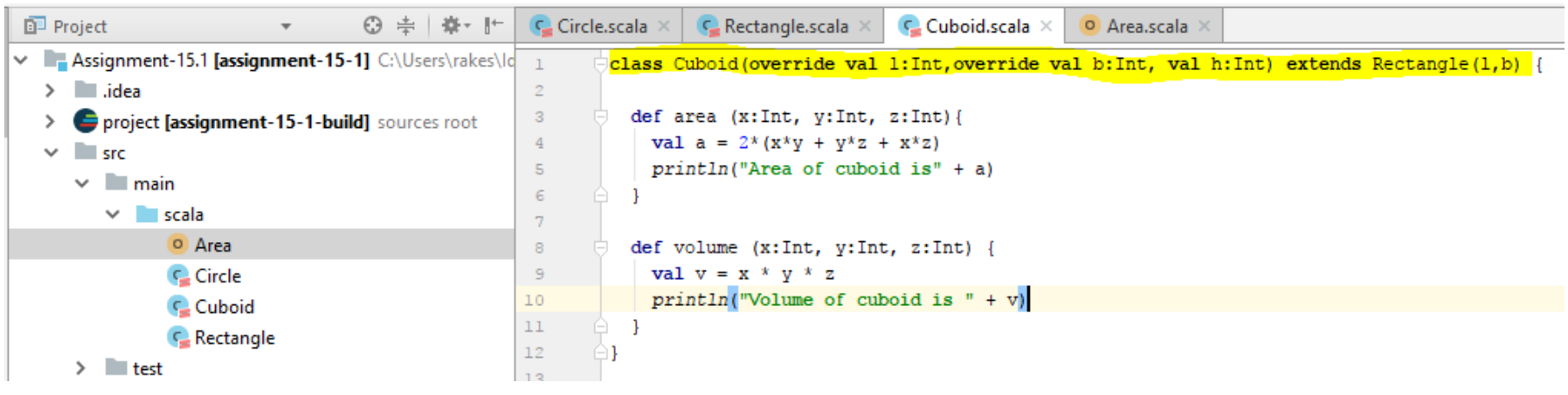
Below screenshot is for class Circle which takes 1 argument as input and is defining a method named area which calculates the area of the Circle-



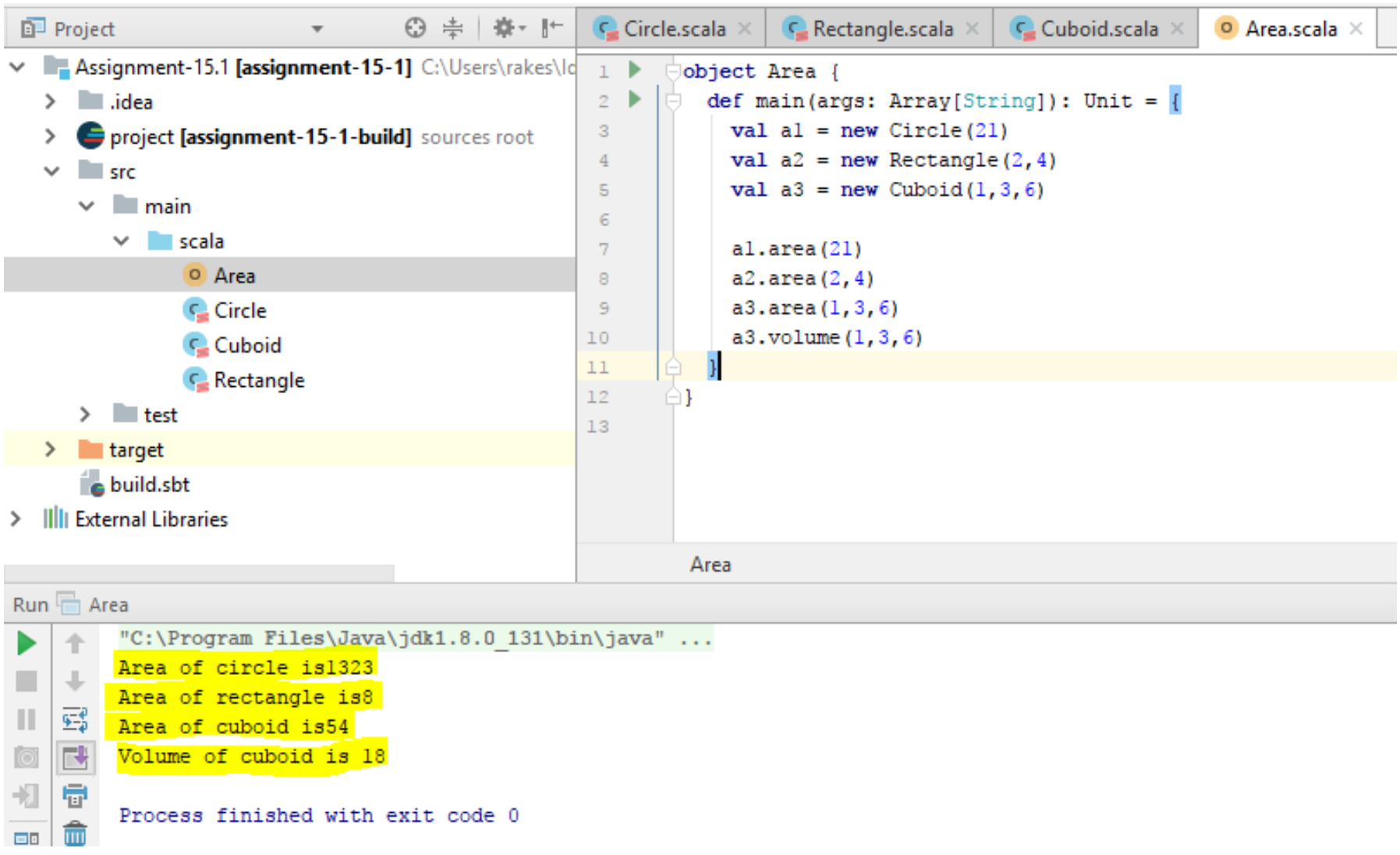
Below screenshot is for class Rectangle which is extending the class Circle and also calculating area of rectangle based on different definition. This takes 2 integers as input-



Below screenshot shows the class definition of Cuboid which takes 3 variables as input and is extending Rectangle class. It contains 2 methods area which calculates surface area of Cuboid and volume which calculates volume of cuboid-



Below screenshot shows the definition of main object Area in which we are passing values to the object of the above classes and calculating the area and volume. Bottom part shows the results for the same-



Problem Statement

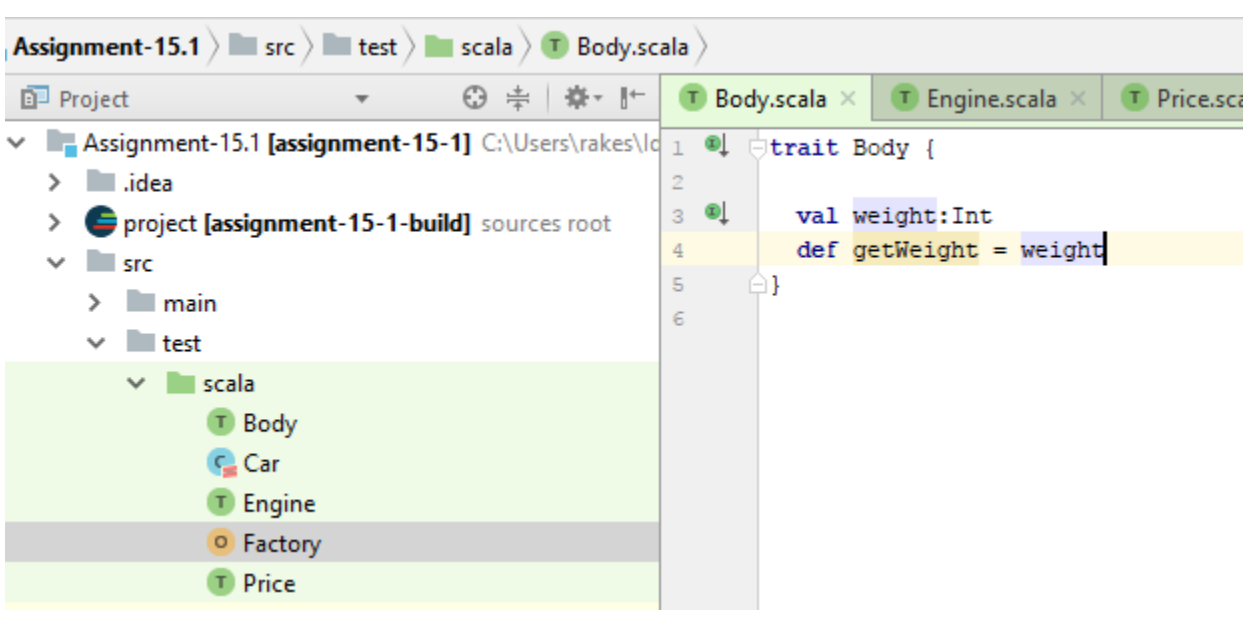
2. Write a simple program to show multiple inheritances in scala.

Solution

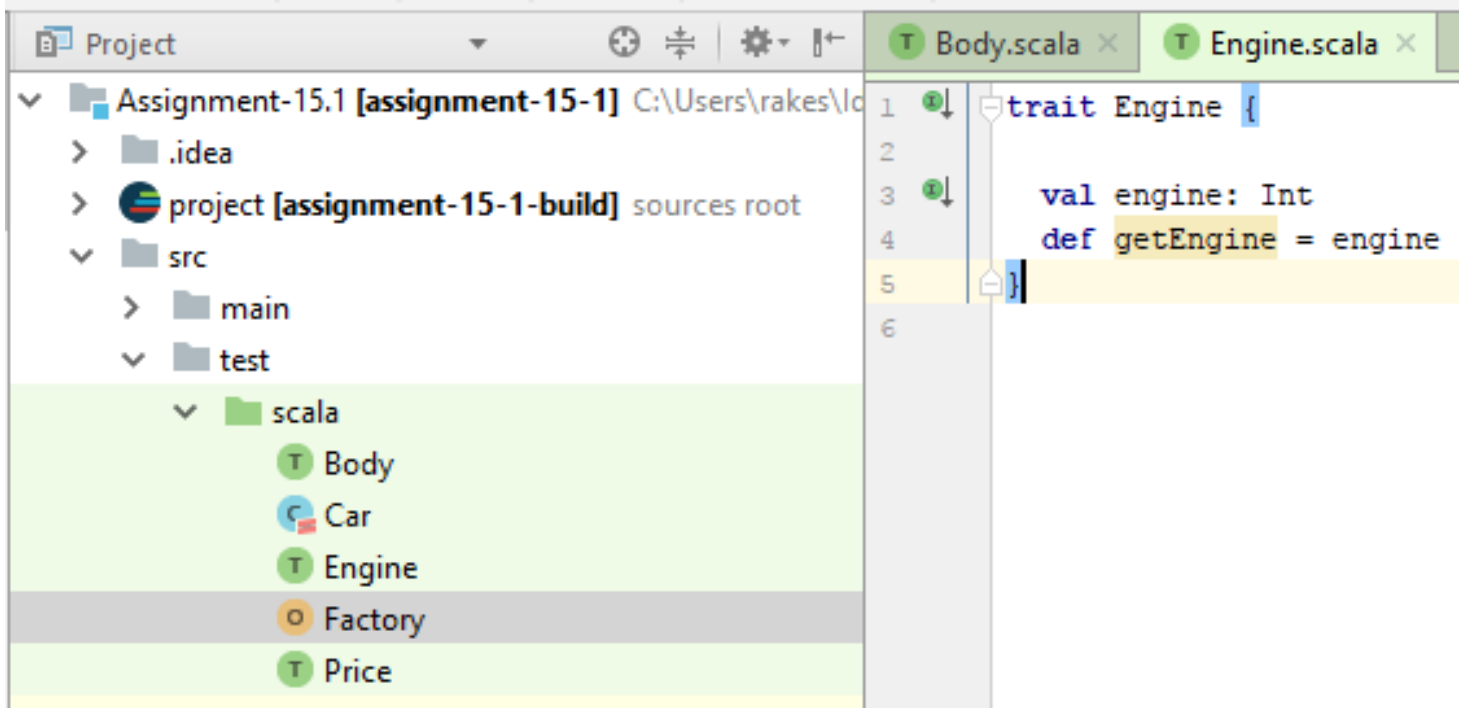
In Scala multiple inheritances is implemented via **traits.** A trait encapsulates method and field definitions, which can then be reused by mixing them into classes. Unlike class inheritance, in which each class must inherit from just one superclass, a class can mix in any number of traits.

Here we are using below Traits to show multiple inheritance in Scala

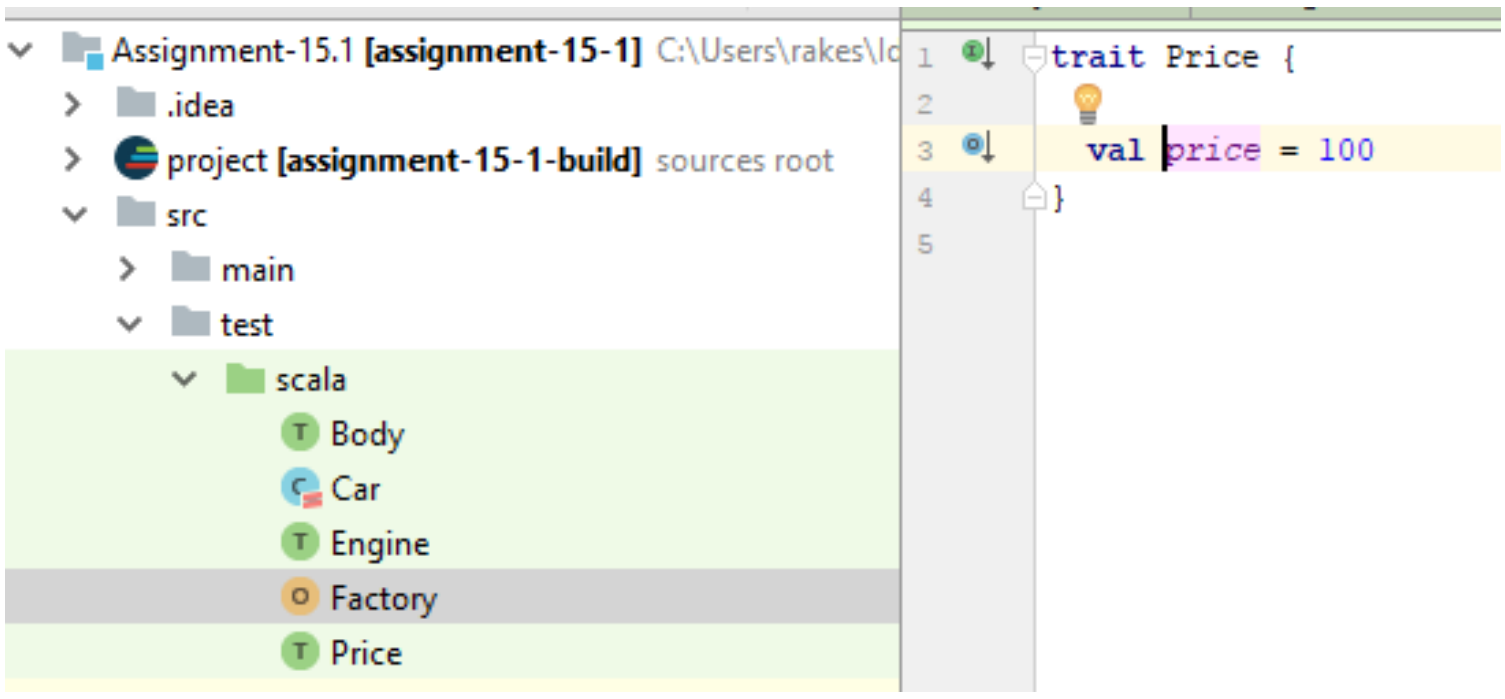
Below is the Trait written named as body which define one val named as weight and one function named as getWeight.



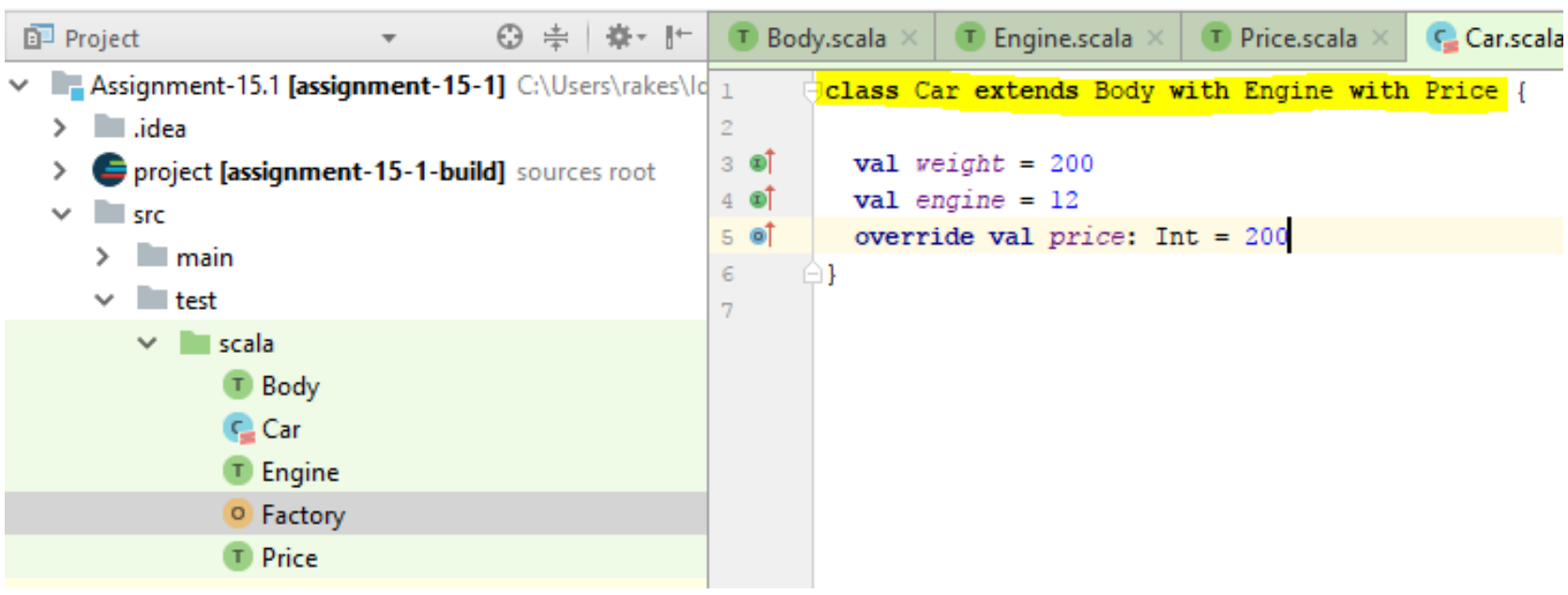
Below is the Trait written named as Engine which define one val named as engine and one function named as getEngine.



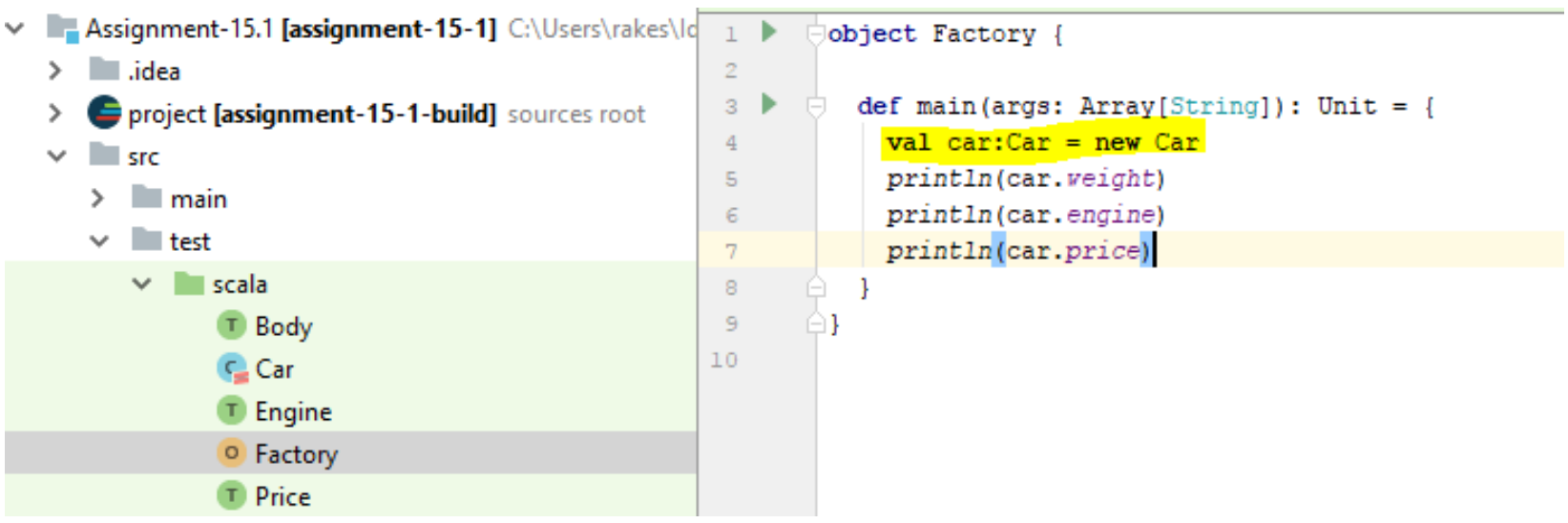
Below is the trait named as Price where we are assigning a val price as 100



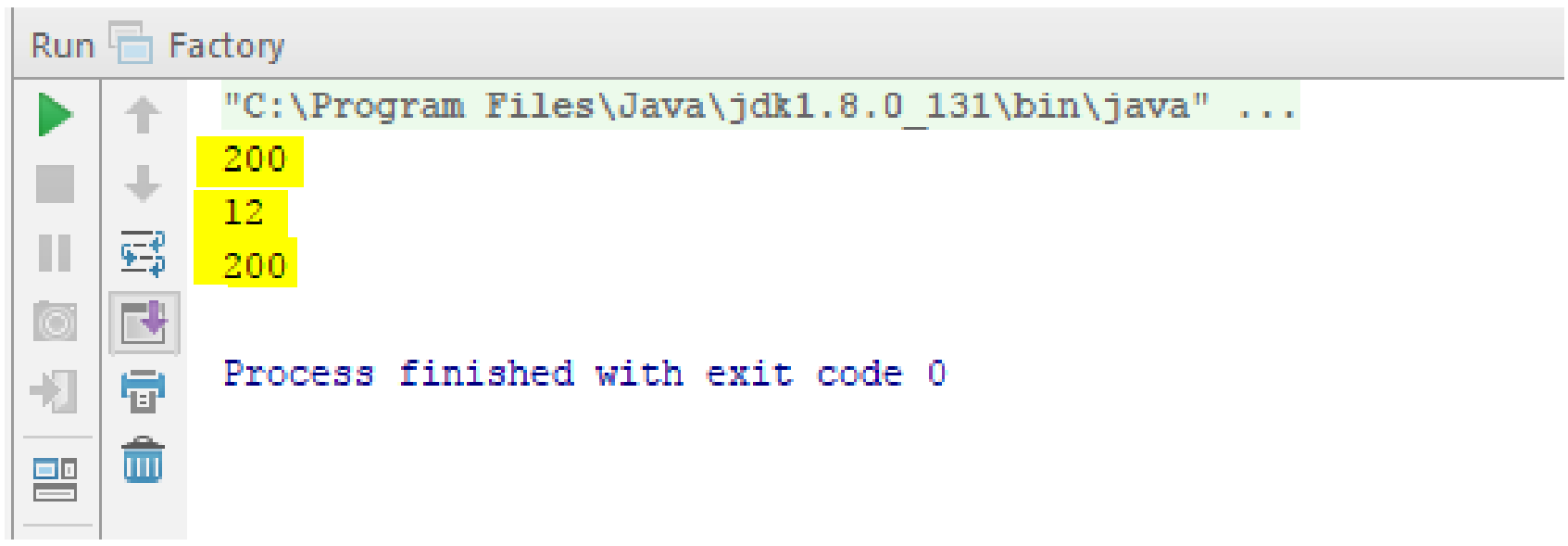
Below is the class written which is inheriting all the properties and parameters defined in above traits. It is extending trait Body with Engine with trait Price



Below is the object defined to print values of all vals and methods defined in above class. Here we are instantiating object of Car and calling the different methods defined in separate traits-



Below is the final screenshot of output-

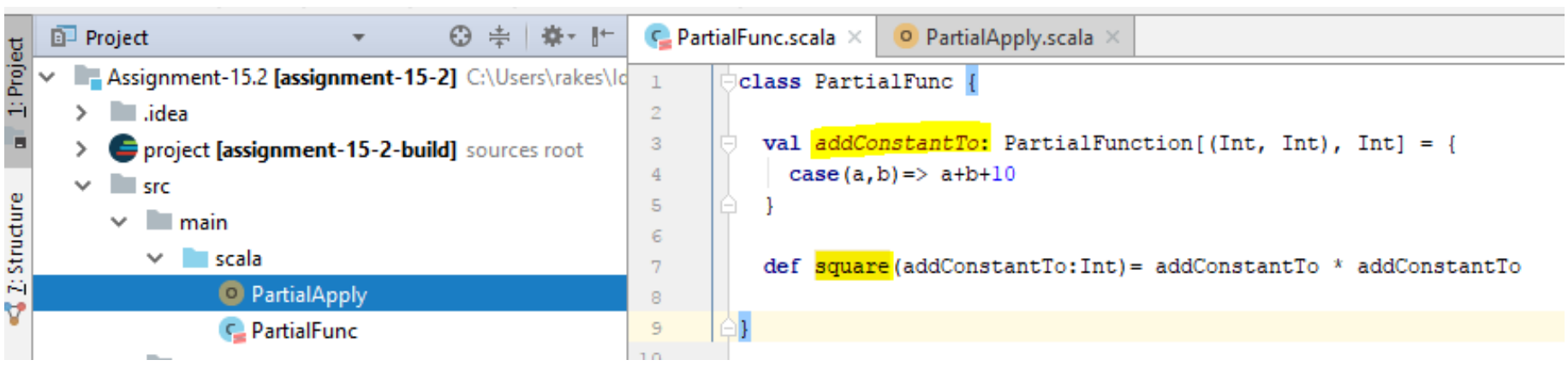


Problem Statement

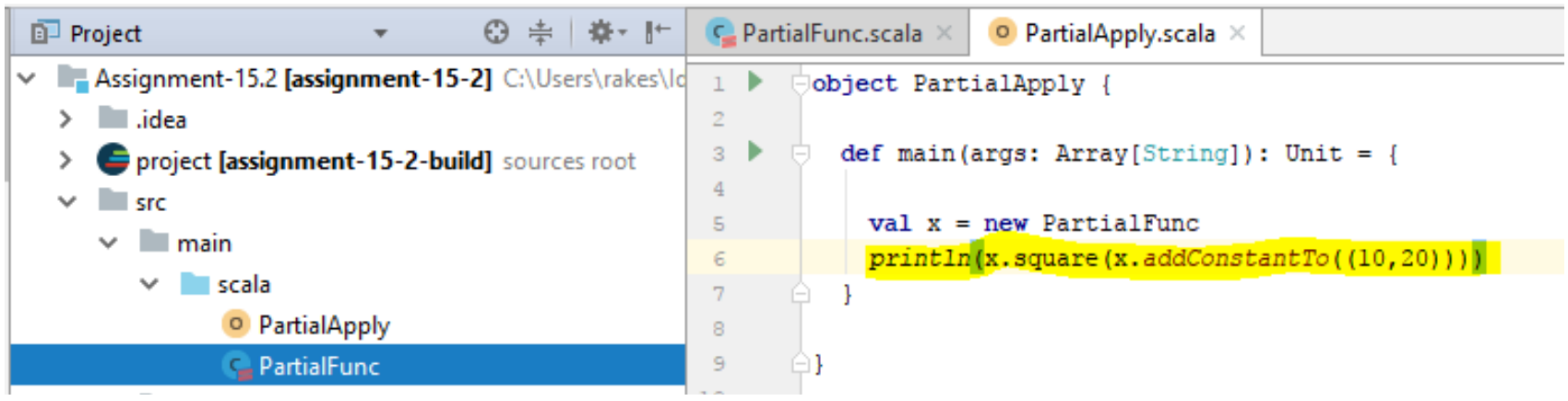
1. Write a partial function to add three numbers in which one number is constant and two numbers can be passed as inputs and define another method which can take the partial function as input and squares the result.

Solution-

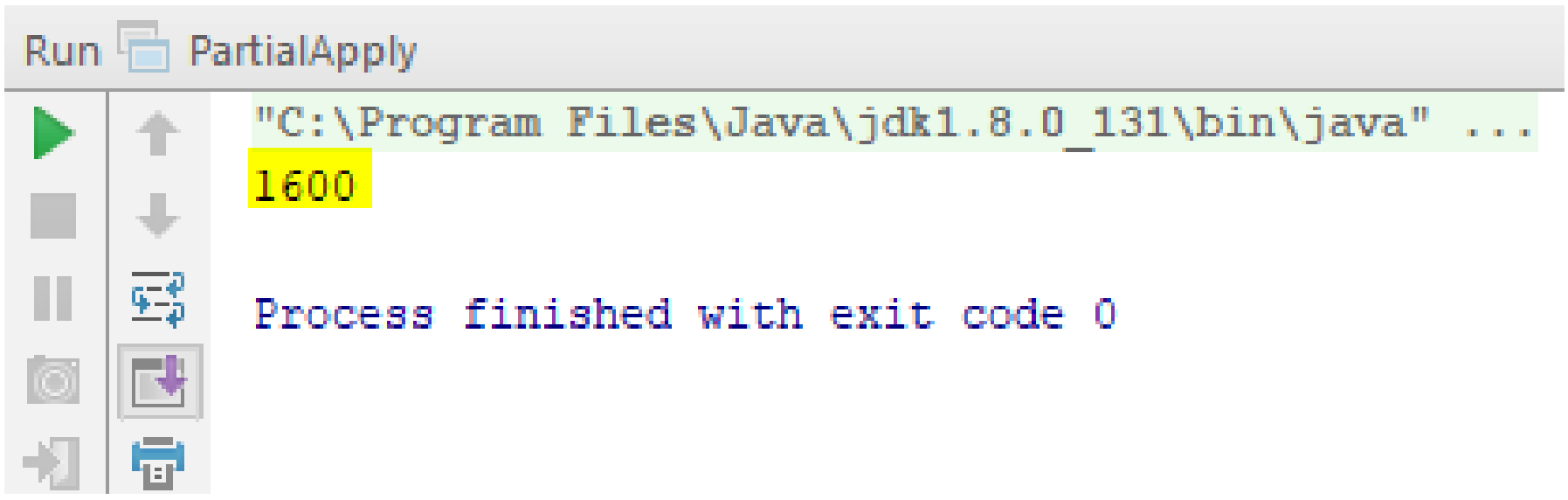
We have defined below class **PartialFunc** which contains partial function which takes 2 integers from user and 1 constant as input and adds the result.
We have 1 more method in same class named as square which takes the above function and squares the result-



Now in the main method we are calling same function. We are passing 10, 20 as input and constant 10 is being added.



Below is the output-



Problem Statement

1. Write a program to print the prices of 4 courses of Acadgild: Android-12999,Big Data Development-17999,Big Data Development-17999,Spark-19999 using match and add a default condition if the user enters any other course.

Solution-

We have defined a class which contains a method for pattern matching which will take string as input which will be course name in our case and will give its price(Int) as output-

Below is the main method in which we are calling this method using course name and its corresponding price is getting generated-

```scala
object TestMatch {

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

    val x = new Matchtest
    println(x.matchCourse("Android App Development"))
    println(x.matchCourse("Data Science"))
    println(x.matchCourse("Big Data Hadoop & Spark Developer"))
    println(x.matchCourse("Blockchain Certification"))
    println(x.matchCourse("Any other course"))
  }

}
```

Below is the output (course price) for the same-

```
Run: TestMatch
"C:\Program Files\Java\jdk1.8.0_131\bin\java.exe" ...
14999
49999
24999
49999
5000

Process finished with exit code 0
```