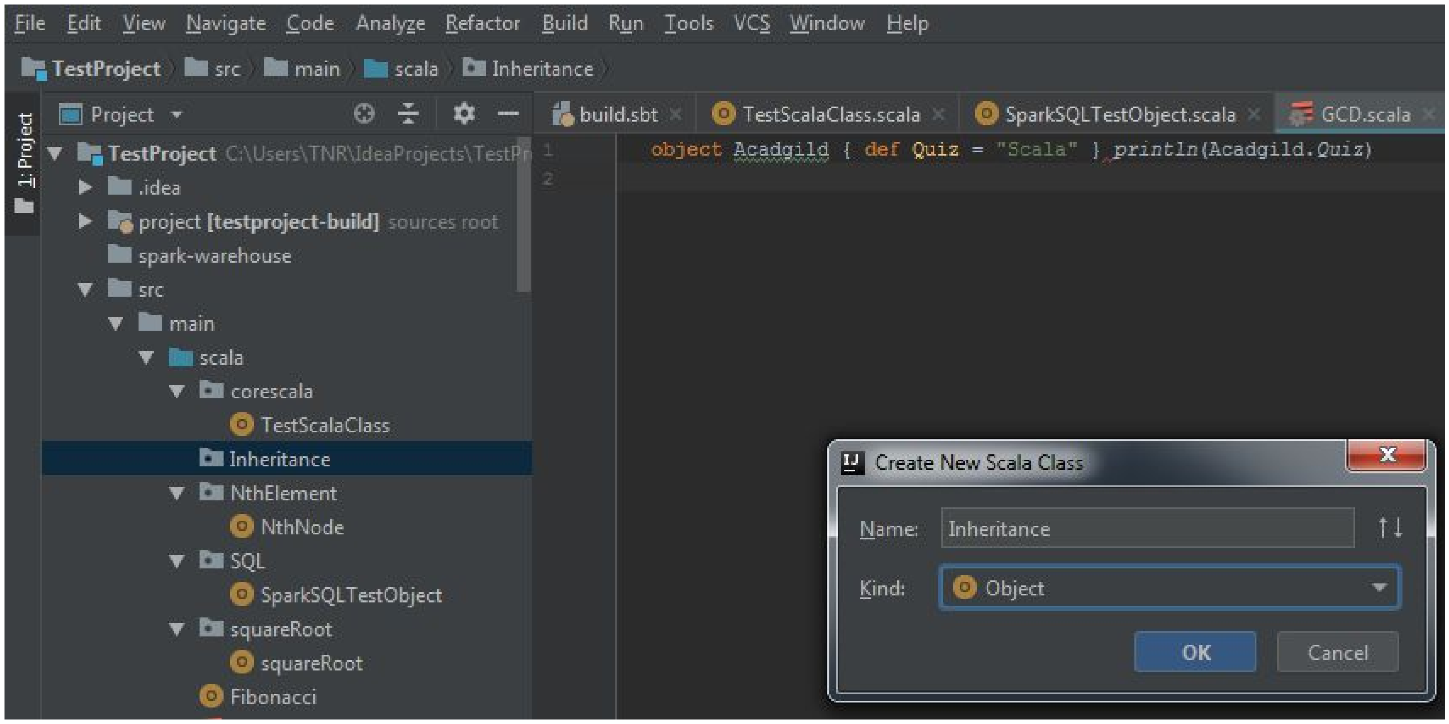
Session 17: SCALA BASICS 4
Assignment 1

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

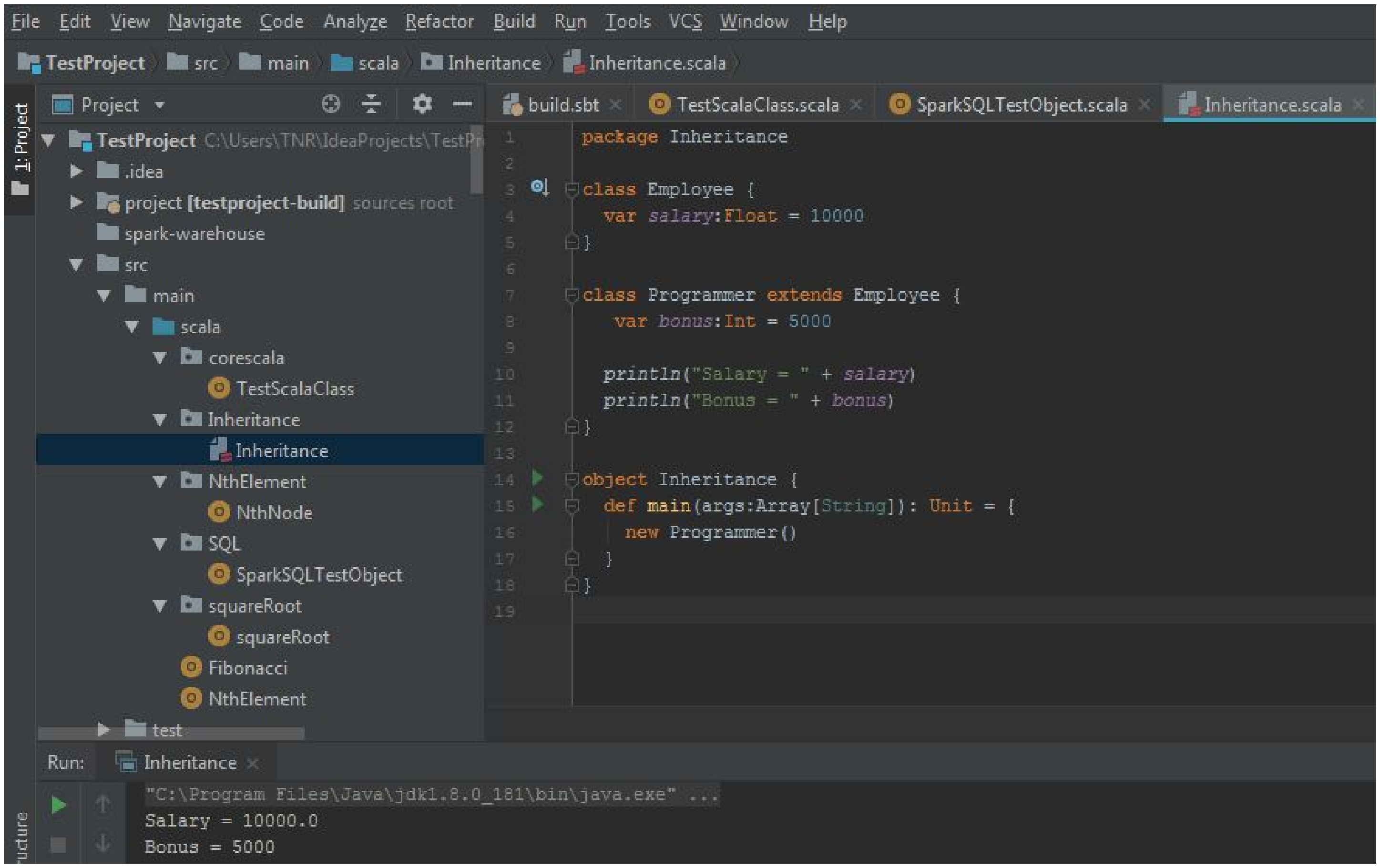
1. Open Intellij IDEA create a project called "TestProject", and create a Scala package with name "inheritance", as follows:



```
object Acadgild { def Quiz = "Scala" } println(Acadgild.Quiz)
```

2. Now write the following simple program which shows inheritance in scala.

```
    Package Inheritance

    class Employee {
        var salary:Float = 10000
    }

    class Programmer extends Employee {
        var bonus:Int = 5000

    println("Salary =" + salary)
    println("Bonus =" + bonus)
}
object Inheritance {
    def main(args:Array[String]): Unit = {
        new Programmer()
    }
}
```

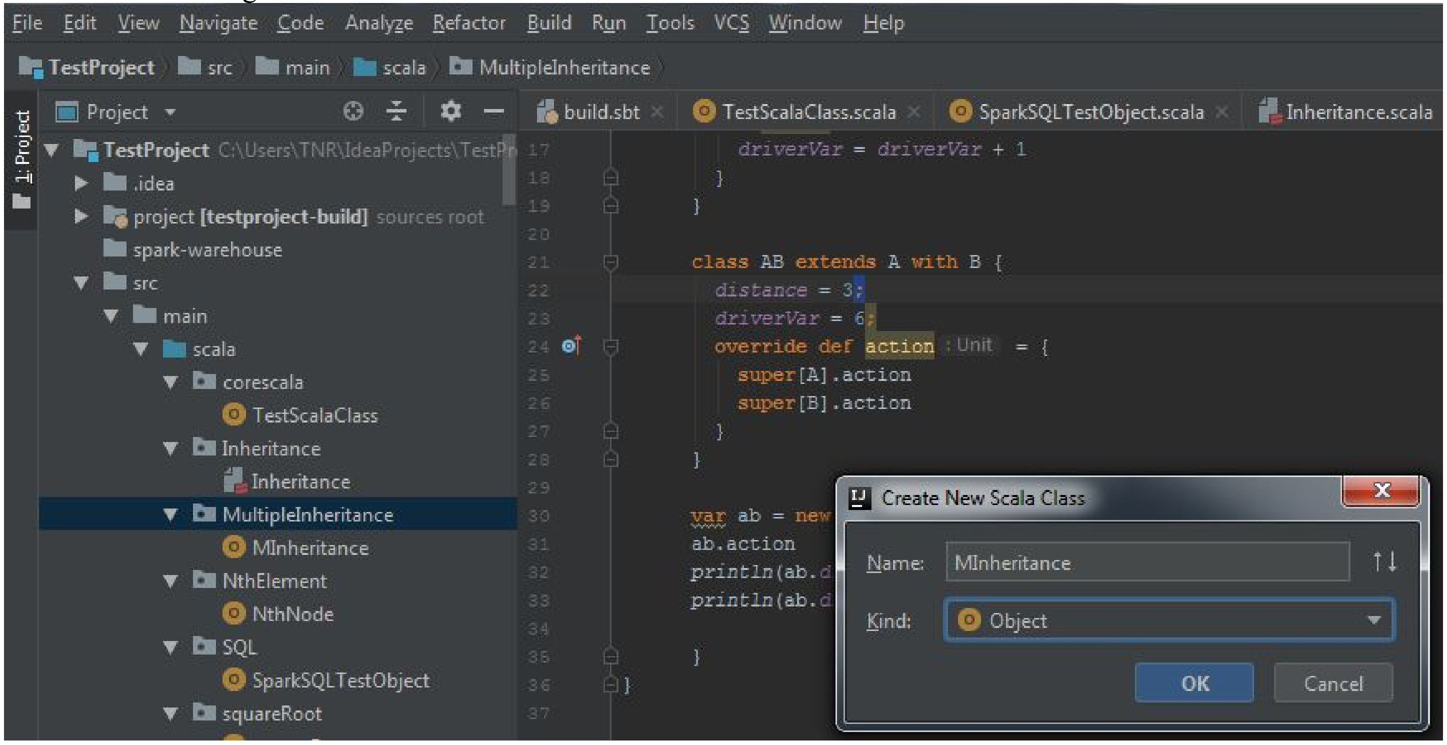


```
package Inheritance

class Employee {
  var salary:Float = 10000
}

class Programmer extends Employee {
   var bonus:Int = 5000

  println("Salary = " + salary)
  println("Bonus = " + bonus)
}

object Inheritance {
  def main(args:Array[String]): Unit = {
    new Programmer()
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Salary = 10000.0
Bonus = 5000
```

Task 2: Write a simple program to show multiple inheritance in scala.

1. Create a new scala package "Multiple Inheritance" and within that create a scala class "MInheritance" and write the following code:



```
        driverVar = driverVar + 1
      }
    }

    class AB extends A with B {
      distance = 3;
      driverVar = 6;
      override def action : Unit = {
        super[A].action
        super[B].action
      }
    }

    var ab = new
    ab.action
    println(ab.d
    println(ab.d

    }
}
```

Create New Scala Class

Name: MInheritance

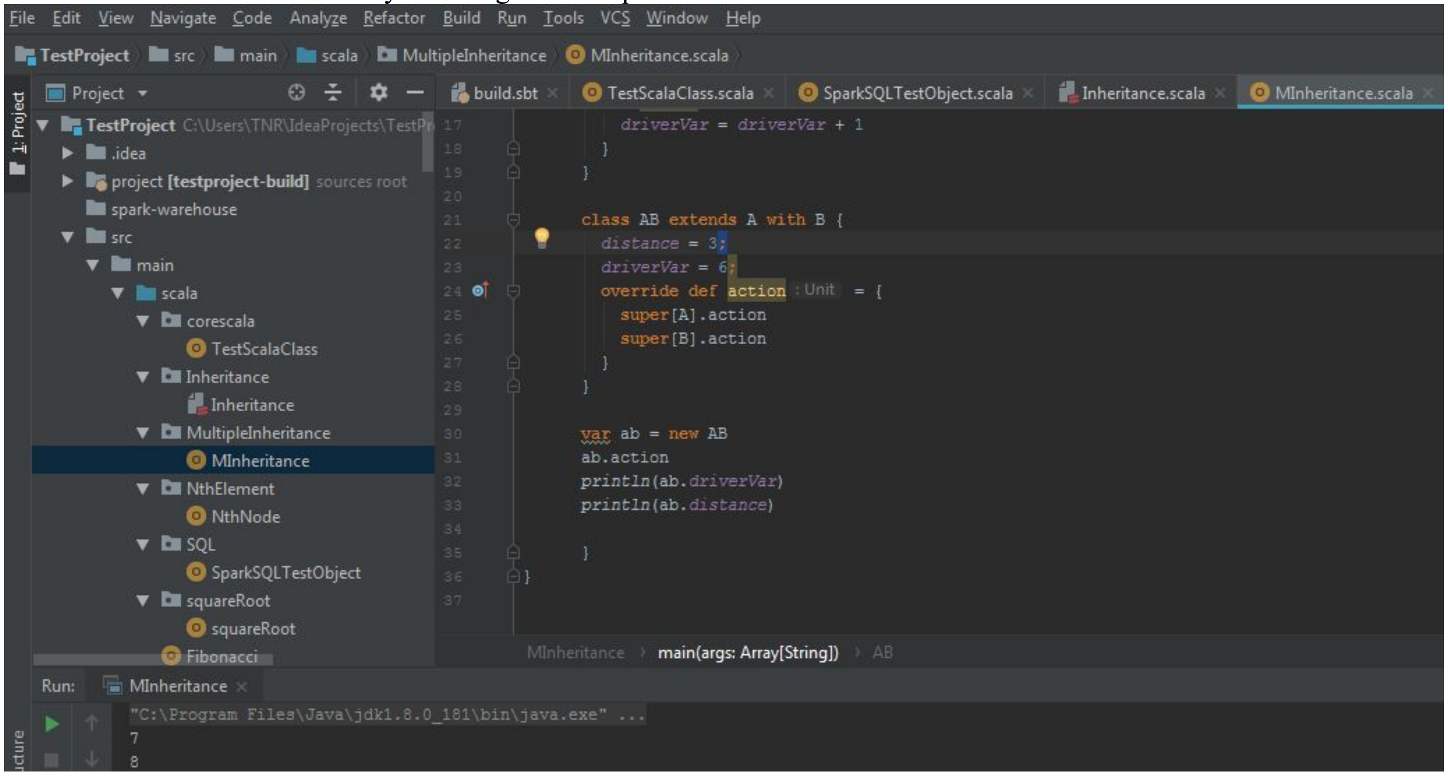
Kind: Object

package MultipleInheritance

object MInheritance {

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

        trait A {
          var distance: Int = _
          def action = {
            distance = distance + 5
          }
        }

        trait B {
          var driverVar: Int = _
          def action = {
            driverVar = driverVar + 1
          }
        }

        class AB extends A with B {
          distance = 3;
          driverVar = 6;
          override def action = {
            super[A].action
            super[B].action
          }
        }

        var ab = new AB
        ab.action
        println(ab.driverVar)
        println(ab.distance)

        }
}
```

2. Now run the above code by selecting the run option from the menu.

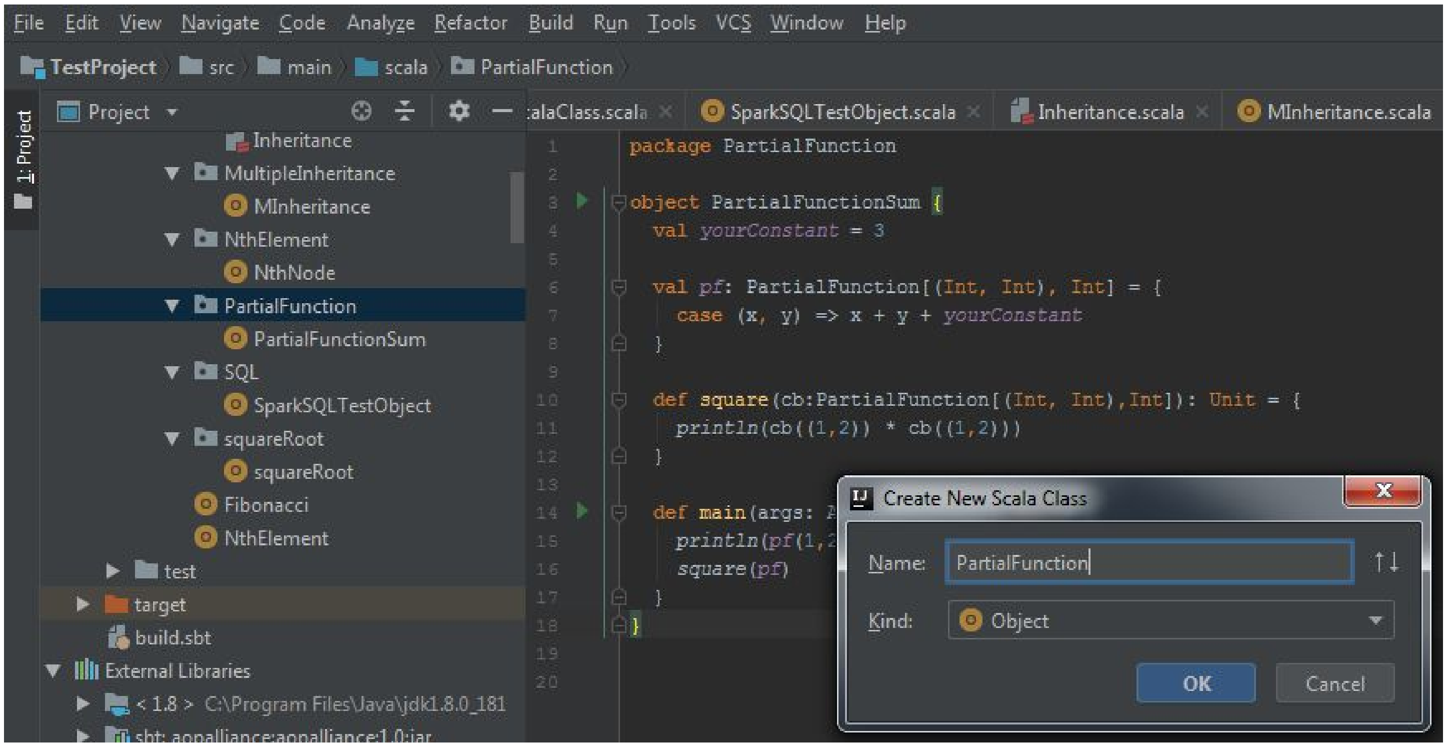


```
      driverVar = driverVar + 1
    }
  }

  class AB extends A with B {
    distance = 3;
    driverVar = 6;
    override def action : Unit = {
      super[A].action
      super[B].action
    }
  }

  var ab = new AB
  ab.action
  println(ab.driverVar)
  println(ab.distance)

  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
7
8
```

Task 3: 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.

1. Create a package with name "Partial Function" and create a scala class with name "PartialFunction" and write the below code into that.



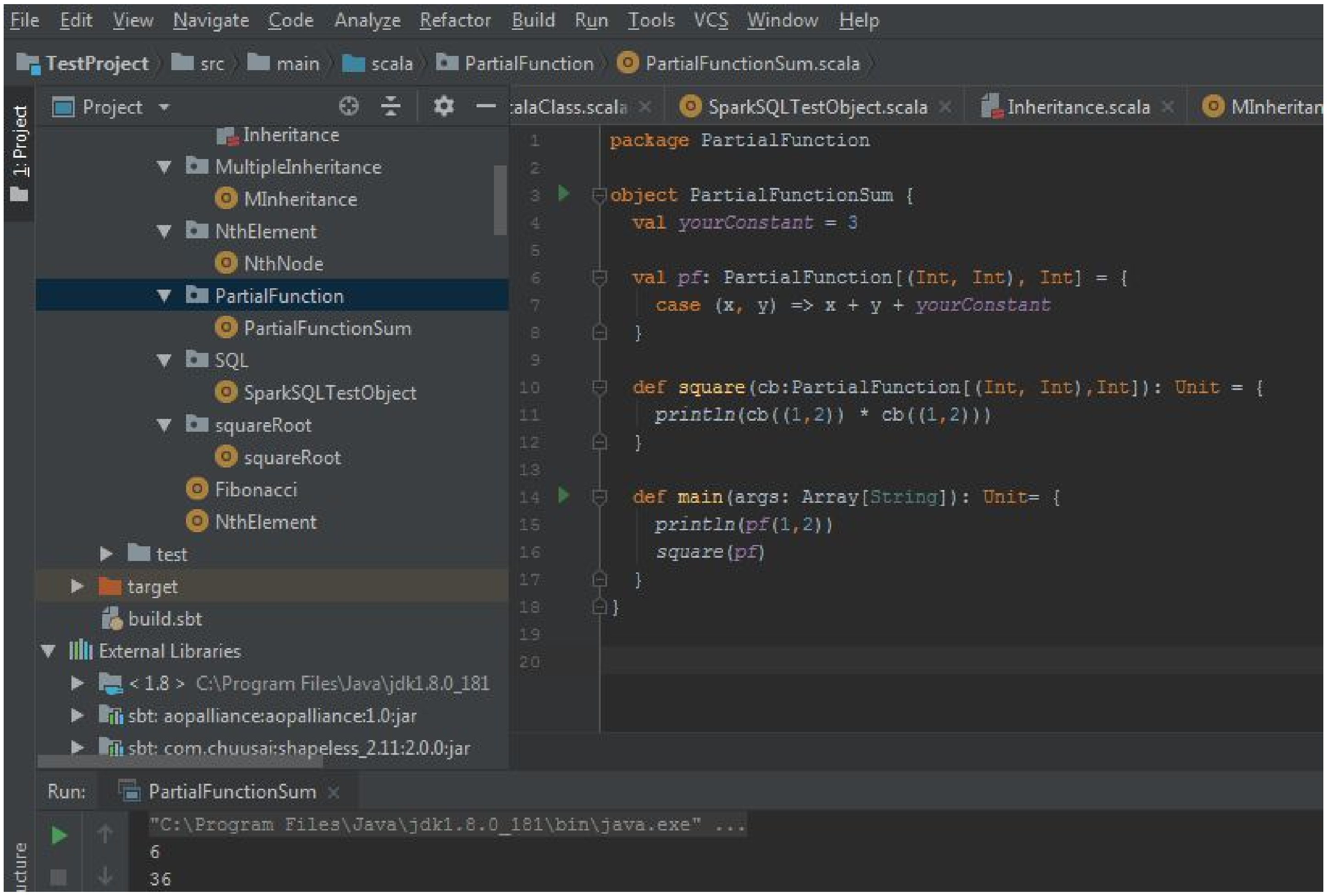
```
package PartialFunction

object PartialFunctionSum {
  val yourConstant = 3

  val pf: PartialFunction[(Int, Int), Int] = {
    case (x, y) => x + y + yourConstant
  }

  def square(cb:PartialFunction[(Int, Int),Int]): Unit = {
    println(cb((1,2)) * cb((1,2)))
  }

  def main(args: A
    println(pf(1,2
    square(pf)
  }
}
```

Name: PartialFunction
Kind: Object

2. Now write the below code into the scala class:

```
package PartialFunction

object PartialFunctionSum {
  val yourConstant = 3

  val pf: PartialFunction[(Int, Int), Int] = {
    case (x, y) => x + y + yourConstant
  }

  def square(cb:PartialFunction[(Int, Int),Int]): Unit = {
    println(cb((1,2)) * cb((1,2)))
  }

  def main(args: Array[String]): Unit= {
    println(pf(1,2))
    square(pf)
  }
}
```

3. Now run the code by selecting the run option on the scala class.

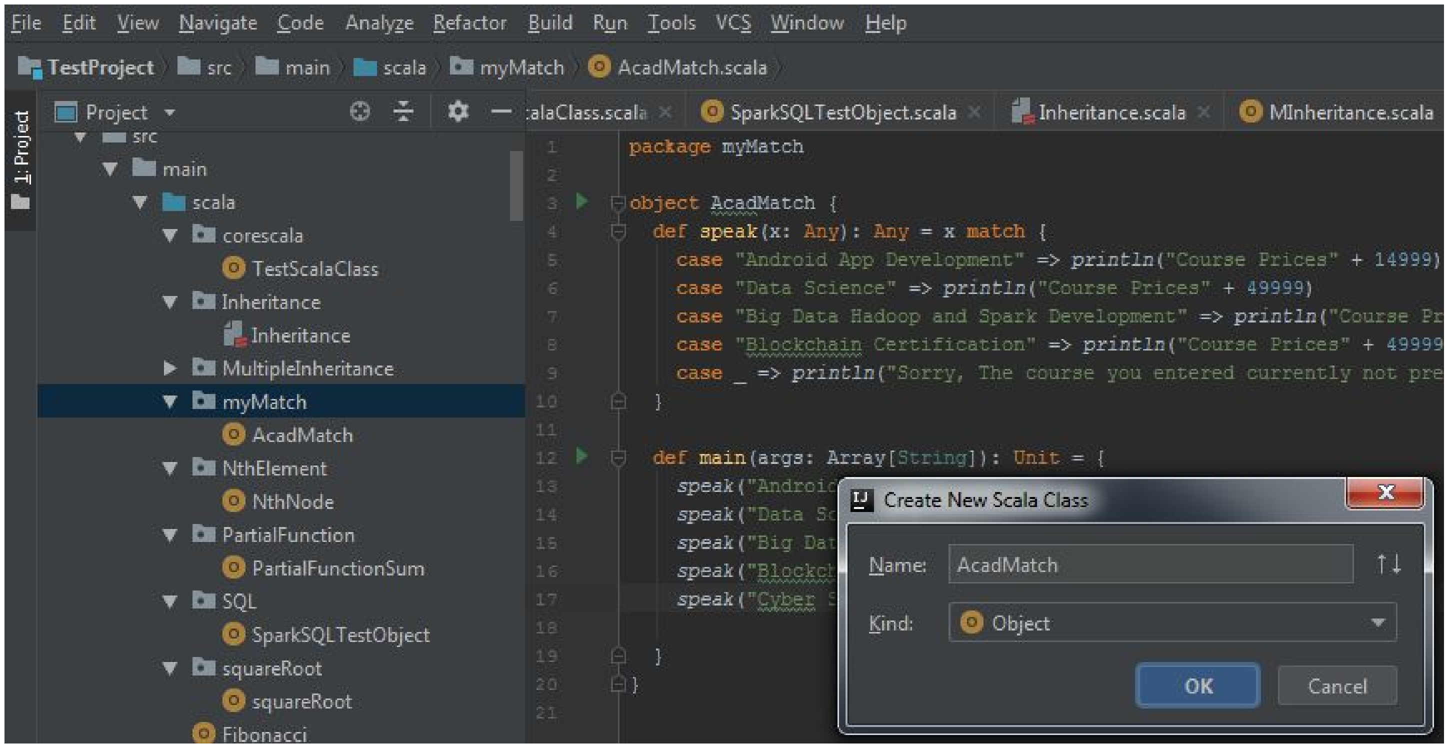

```
package PartialFunction

object PartialFunctionSum {
  val yourConstant = 3

  val pf: PartialFunction[(Int, Int), Int] = {
    case (x, y) => x + y + yourConstant
  }

  def square(cb:PartialFunction[(Int, Int),Int]): Unit = {
    println(cb((1,2)) * cb((1,2)))
  }

  def main(args: Array[String]): Unit= {
    println(pf(1,2))
    square(pf)
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
6
36
```

We can observe that the partial function prints the value 6, and the square function prints the 36.

Task 4:
Write a program to print the prices of 4 courses of Acadgild:
Android App Development – 14,999 INR
Data Science – 49,999 INR
Big Data Hadoop & Spark Development – 24,999 INR
Blockchain Certification – 49,999 INR.
Using match and add a default condition if the user enters any other course.

1. Create a Scala package with name “myMatch” and create a scala package with name “AcadMatch” and write the following code:



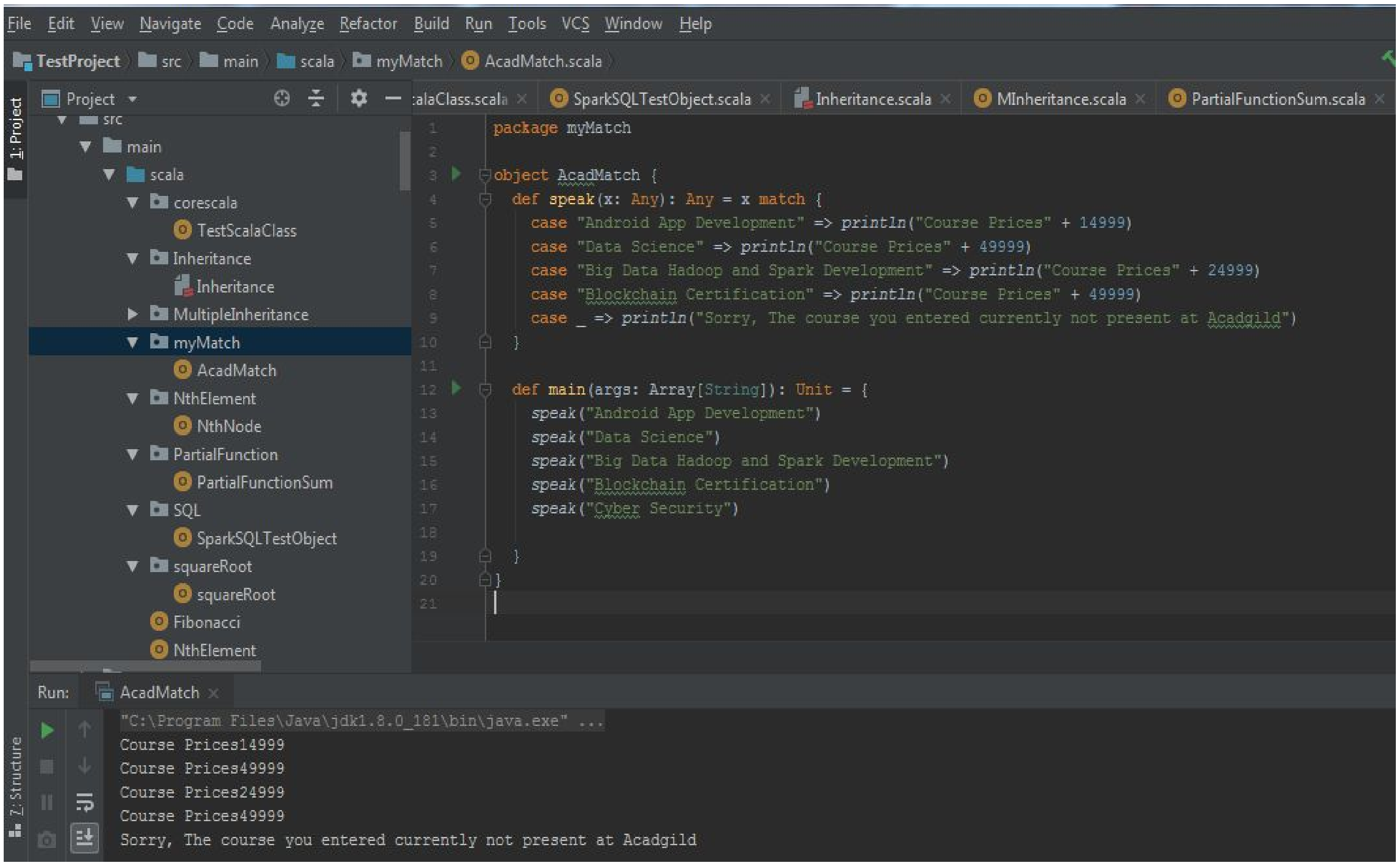
```
package myMatch

object AcadMatch {
  def speak(x: Any): Any = x match {
    case "Android App Development" => println("Course Prices" + 14999)
    case "Data Science" => println("Course Prices" + 49999)
    case "Big Data Hadoop and Spark Development" => println("Course Pr
    case "Blockchain Certification" => println("Course Prices" + 49999
    case _ => println("Sorry, The course you entered currently not pre
  }

  def main(args: Array[String]): Unit = {
    speak("Androic
    speak("Data Sc
    speak("Big Dat
    speak("Blockch
    speak("Cyber S

  }
}
```

Name: AcadMatch
Kind: Object

package myMatch

```
object AcadMatch {
  def speak(x: Any): Any = x match {
    case "Android App Development" => println("Course Prices" + 14999)
    case "Data Science" => println("Course Prices" + 49999)
    case "Big Data Hadoop and Spark Development" => println("Course Prices" + 24999)
    case "Blockchain Certification" => println("Course Prices" + 49999)
    case _ => println("Sorry, The course you entered currently not present at Acadgild")
  }

  def main(args: Array[String]): Unit = {
    speak("Android App Development")
    speak("Data Science")
    speak("Big Data Hadoop and Spark Development")
    speak("Blockchain Certification")
    speak("Cyber Security")

  }
}
```

2. Run the above program we’ll see the output as follows:



```scala
package myMatch

object AcadMatch {
  def speak(x: Any): Any = x match {
    case "Android App Development" => println("Course Prices" + 14999)
    case "Data Science" => println("Course Prices" + 49999)
    case "Big Data Hadoop and Spark Development" => println("Course Prices" + 24999)
    case "Blockchain Certification" => println("Course Prices" + 49999)
    case _ => println("Sorry, The course you entered currently not present at Acadgild")
  }

  def main(args: Array[String]): Unit = {
    speak("Android App Development")
    speak("Data Science")
    speak("Big Data Hadoop and Spark Development")
    speak("Blockchain Certification")
    speak("Cyber Security")

  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Course Prices14999
Course Prices49999
Course Prices24999
Course Prices49999
Sorry, The course you entered currently not present at Acadgild
```