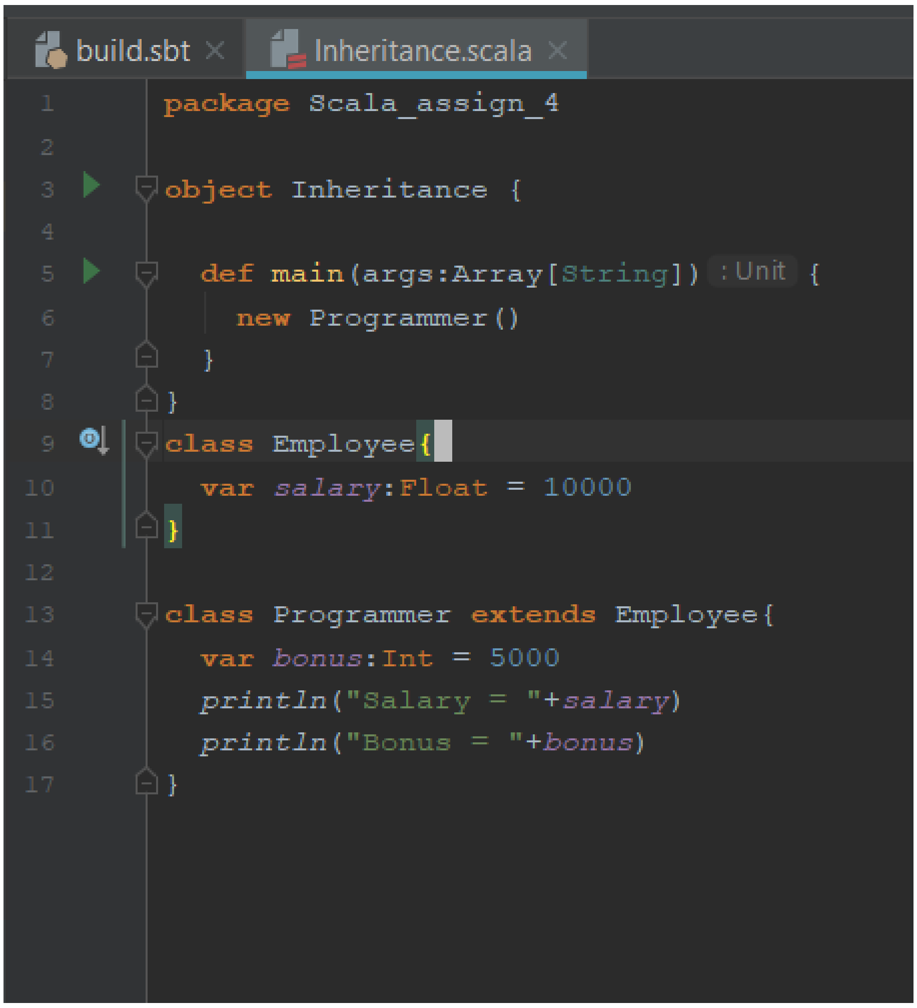
Assignment -17

Task 1:

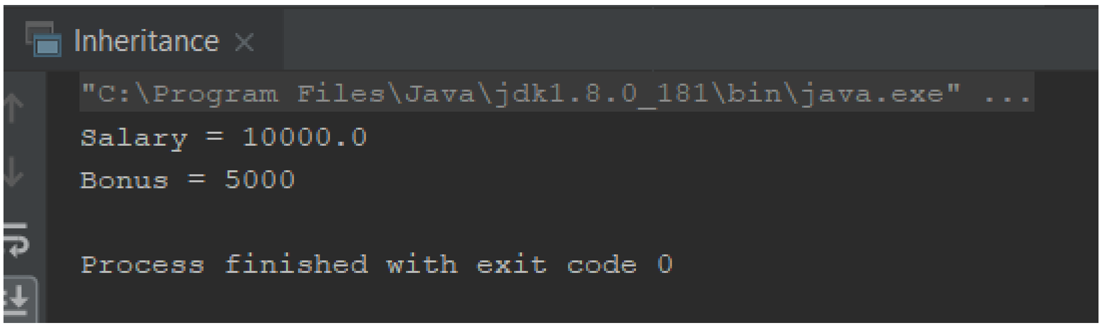
Write a simple program to show inheritance in scala.

Scala code to show inheritance is shown the below screenshot.



```scala
package Scala_assign_4

object Inheritance {

  def main(args:Array[String]) :Unit {
    new Programmer()
  }
}
class Employee{
  var salary:Float = 10000
}

class Programmer extends Employee{
  var bonus:Int = 5000
  println("Salary = "+salary)
  println("Bonus = "+bonus)
}
```

Output of the above code is as shown in the below screenshot.

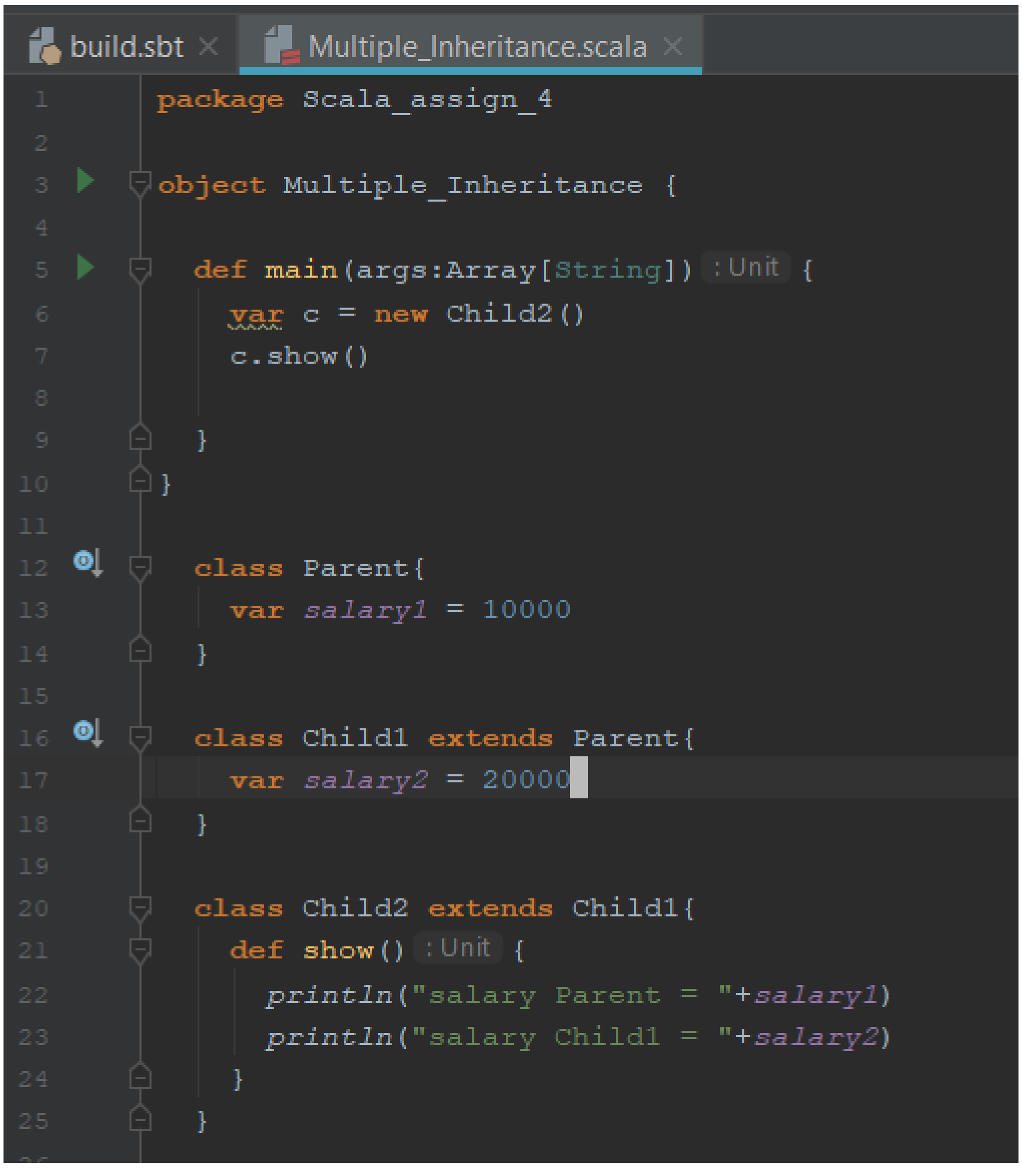


```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Salary = 10000.0
Bonus = 5000

Process finished with exit code 0
```

Task 2

Write a simple program to show multiple inheritance in scala

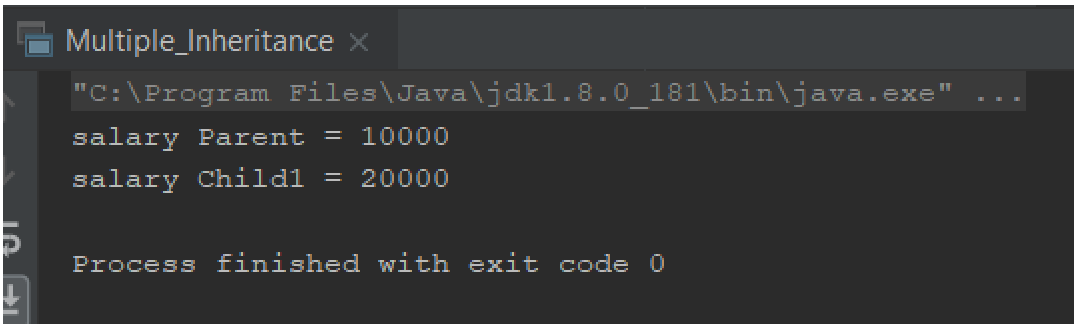
Scala code to implement multiple inheritance is shown in the screenshot.



build.sbt × Multiple_Inheritance.scala ×

```scala
package Scala_assign_4

object Multiple_Inheritance {

  def main(args:Array[String]) : Unit {
    var c = new Child2()
    c.show()

  }
}

  class Parent{
    var salary1 = 10000
  }

  class Child1 extends Parent{
    var salary2 = 20000
  }

  class Child2 extends Child1{
    def show() : Unit {
      println("salary Parent = "+salary1)
      println("salary Child1 = "+salary2)
    }
  }
```

Output of the above code is shown in the below screenshot.



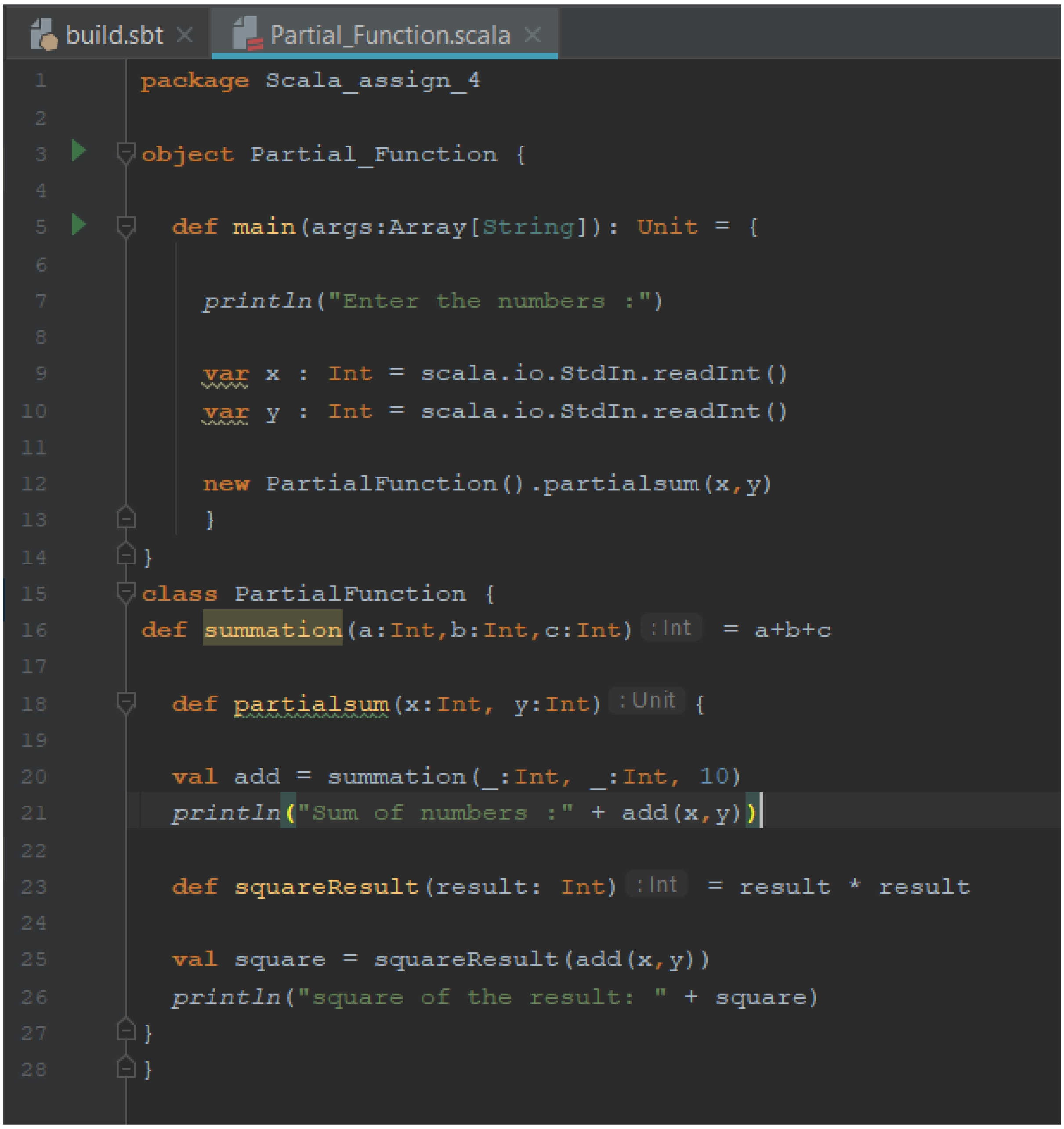
Multiple_Inheritance ×

```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
salary Parent = 10000
salary Child1 = 20000

Process finished with exit code 0
```

Task 3:

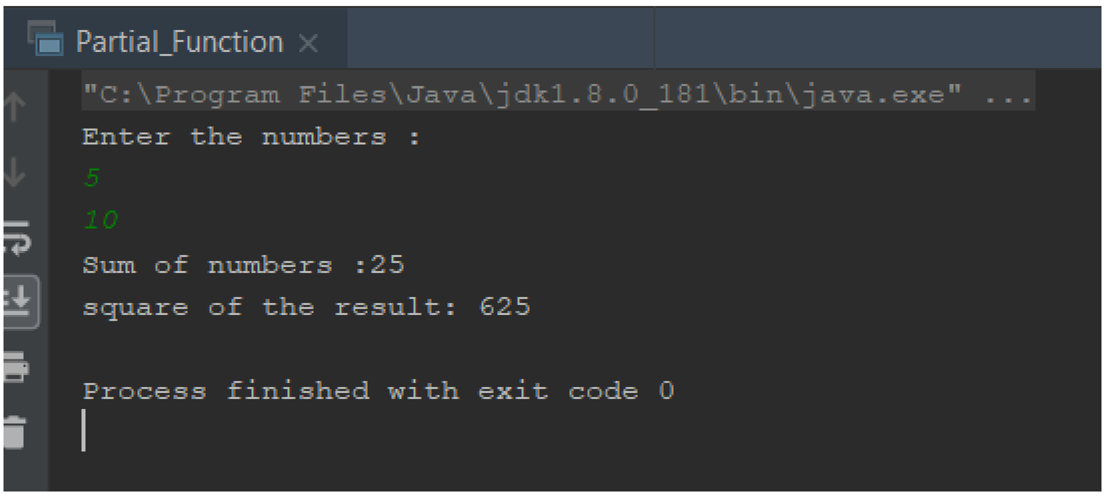
Write a partial function to add three numbers in which one number is constant and two numbers can be passed as inputs and define another method which can take the partial function as input and squares the result.

Partial function to add three numbers and squaring the result of partial function is implemented using the below scala code.



```scala
package Scala_assign_4

object Partial_Function {

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

    println("Enter the numbers :")

    var x : Int = scala.io.StdIn.readInt()
    var y : Int = scala.io.StdIn.readInt()

    new PartialFunction().partialsum(x,y)
    }
}
class PartialFunction {
def summation(a:Int,b:Int,c:Int) : Int  = a+b+c

  def partialsum(x:Int, y:Int) : Unit {

  val add = summation(_:Int, _:Int, 10)
  println("Sum of numbers :" + add(x,y))

  def squareResult(result: Int) : Int  = result * result

  val square = squareResult(add(x,y))
  println("square of the result: " + square)
}
}
```

The output of the above code is shown in the below screenshot.



```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Enter the numbers :
5
10
Sum of numbers :25
square of the result: 625

Process finished with exit code 0
```

Task 4

Write a program to print the prices of 4 courses of Acadgild:

Android App Development -14,999 INR

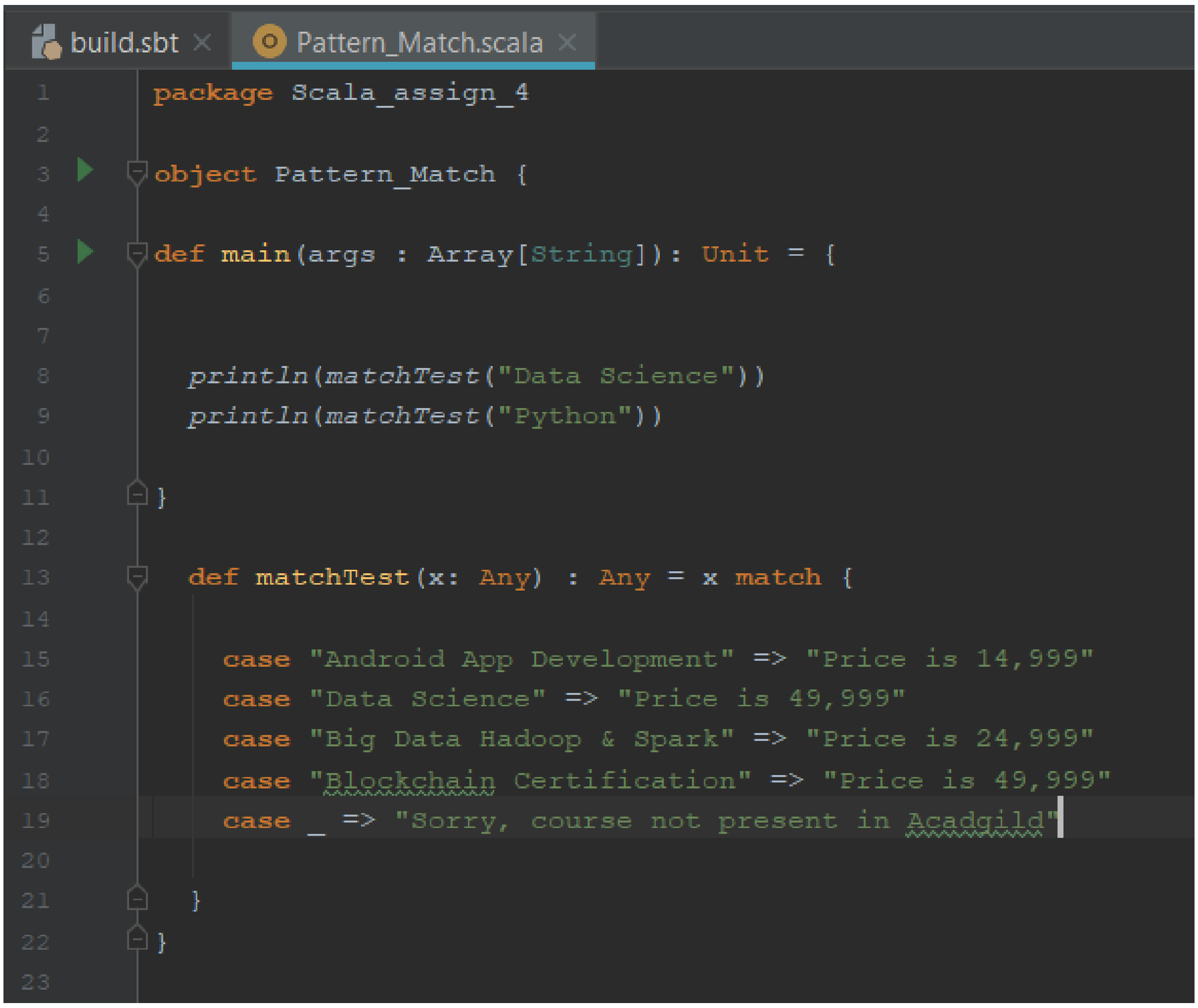
Data Science - 49,999 INR

Big Data Hadoop & Spark Developer – 24,999 INR

Blockchain Certification – 49,999 INR

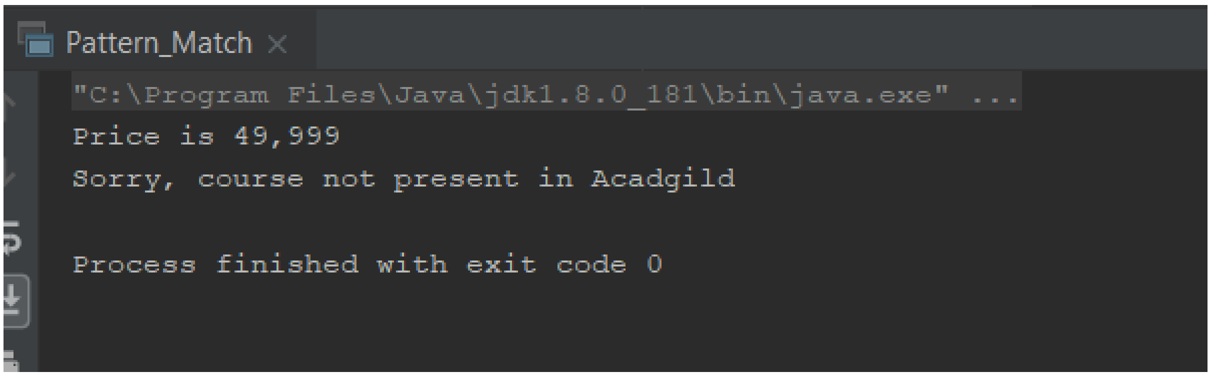
using match and add a default condition if the user enters any other course.

Scala code to implement the above pattern match conditions has been shown in the below screenshot.



```scala
package Scala_assign_4

object Pattern_Match {

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


  println(matchTest("Data Science"))
  println(matchTest("Python"))

}

  def matchTest(x: Any) : Any = x match {

    case "Android App Development" => "Price is 14,999"
    case "Data Science" => "Price is 49,999"
    case "Big Data Hadoop & Spark" => "Price is 24,999"
    case "Blockchain Certification" => "Price is 49,999"
    case _ => "Sorry, course not present in Acadgild"

  }
}
```

The output of the above code is shown in the below screenshot.



```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Price is 49,999
Sorry, course not present in Acadgild

Process finished with exit code 0
```