SCALA - SESSION IV

Assignment

Student Name: Subham Vishal

Course: Big Data Hadoop & Spark Training

Assignment
Write a simple program in Scala to show partial function and match and add.

Contents

Introduction

In this assignment, we are going to write a simple SCALA code to show partial function and match and add

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.
2. 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.

Task 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.

What is Partial Function?

A **partial function** is a function that does not provide an answer for every possible input value it can be given. It provides an answer only for a subset of possible data, and defines the data it can handle. In Scala, a **partial function** can also be queried to determine if it can handle a particular value

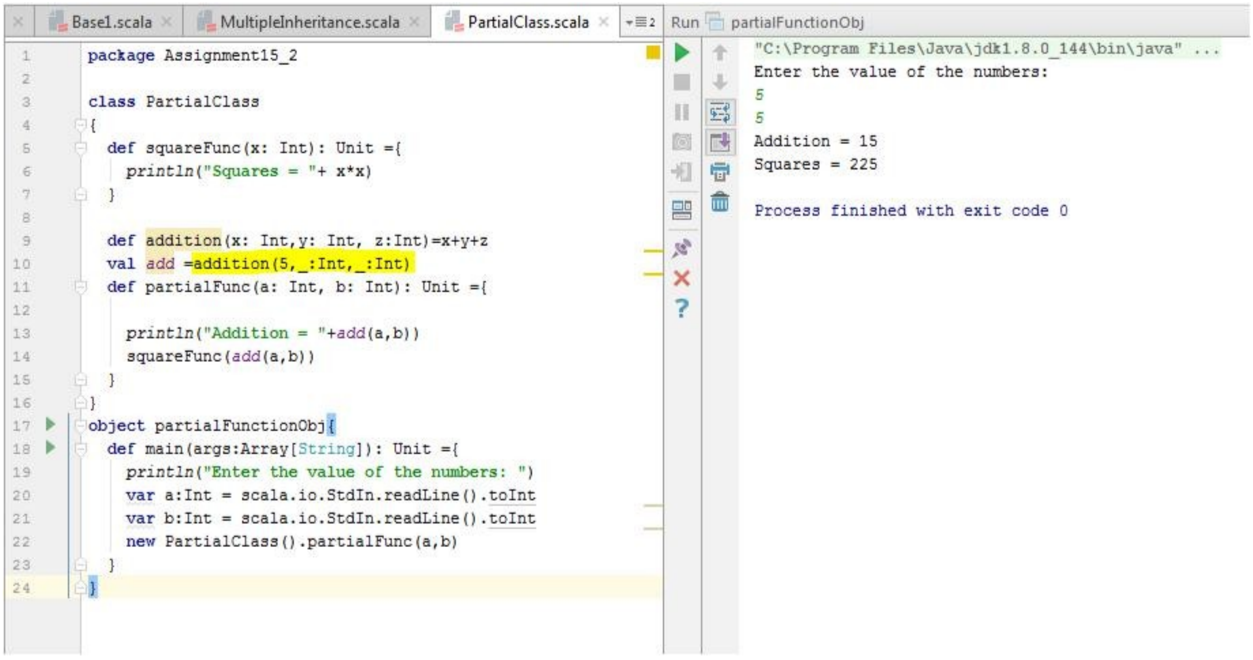
Scala Code

```
package Assignment15_2

class PartialClass
{
  def squareFunc(x: Int): Unit ={
    println("Squares = "+ x*x) // defined a function to square the input's
  }

  def addition(x: Int,y: Int, z:Int)=x+y+z//a function to add
  constant+value1+value2 val add =addition(5,_:Int,_:Int) // the constant value = 5
  def partialFunc(a: Int, b: Int): Unit ={ // another method to define a value
for constant


    println("Addition = "+add(a,b))
    squareFunc(add(a,b))
  }
}
object partialFunctionObj{ // singleton object to call the
  functions def main(args:Array[String]): Unit ={
    println("Enter the value of the numbers: ")
    var a:Int = scala.io.StdIn.readLine().toInt // reading the input
    value var b:Int = scala.io.StdIn.readLine().toInt new
    PartialClass().partialFunc(a,b) //
  }
}
```

Here the constant is x and we defined the value of **x as 5**, we have two variables a and b, we pass **a=y=5** and **b=z=5**, we get the **x+y+z** = **5+5+5 = 15**.

15 is the output of the partial function is squared **15*15** in the **squareFunc** which is **225**.

Output



```
package Assignment15_2

class PartialClass
{
  def squareFunc(x: Int): Unit ={
    println("Squares = "+ x*x)
  }

  def addition(x: Int,y: Int, z:Int)=x+y+z
  val add =addition(5,_:Int,_:Int)
  def partialFunc(a: Int, b: Int): Unit ={

    println("Addition = "+add(a,b))
    squareFunc(add(a,b))
  }
}
object partialFunctionObj{
  def main(args:Array[String]): Unit ={
    println("Enter the value of the numbers: ")
    var a:Int = scala.io.StdIn.readLine().toInt
    var b:Int = scala.io.StdIn.readLine().toInt
    new PartialClass().partialFunc(a,b)
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Enter the value of the numbers:
5
5
Addition = 15
Squares = 225

Process finished with exit code 0
```

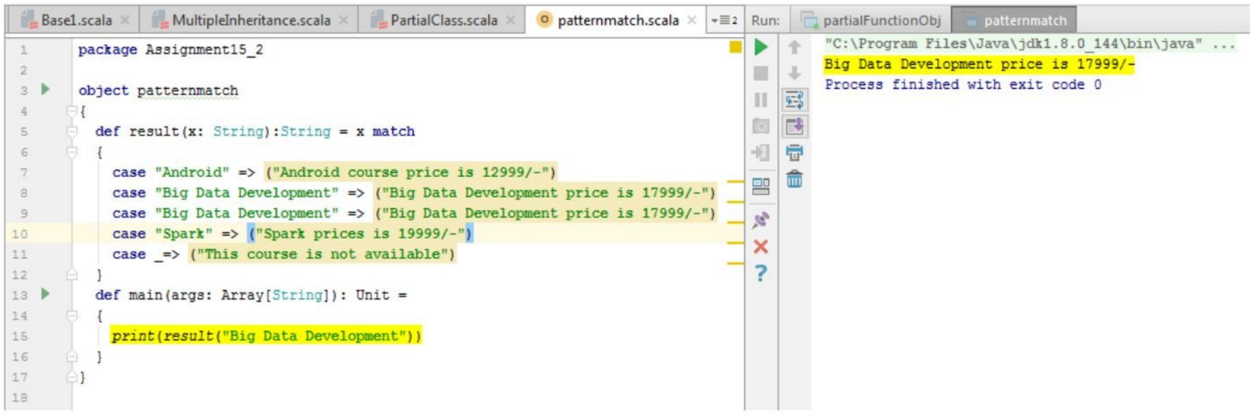
Task2

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.

Scala Code

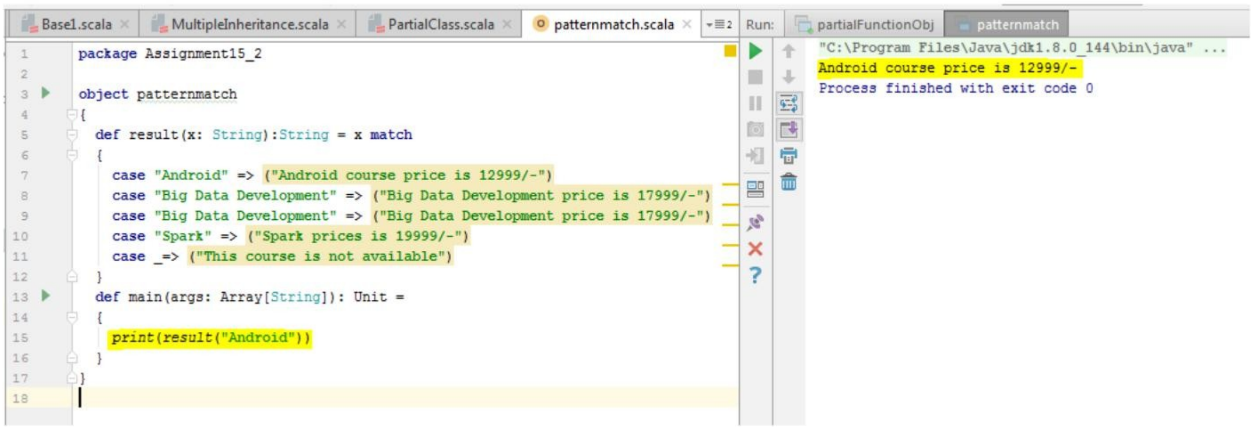
```
package Assignment15_2

object patternmatch
{
  def result(x: String):String = x match
    {
    case "Android" => ("Android course price is 12999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Spark" => ("Spark prices is 19999/-")
    case _=> ("This course is not available")
    }
  def main(args: Array[String]): Unit =
  {
    print(result("Big Data Development"))
  }
}
```

Output



```
package Assignment15_2

object patternmatch
{
  def result(x: String):String = x match
  {
    case "Android" => ("Android course price is 12999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Spark" => ("Spark prices is 19999/-")
    case _=> ("This course is not available")
  }
  def main(args: Array[String]): Unit =
  {
    print(result("Big Data Development"))
  }
}
```

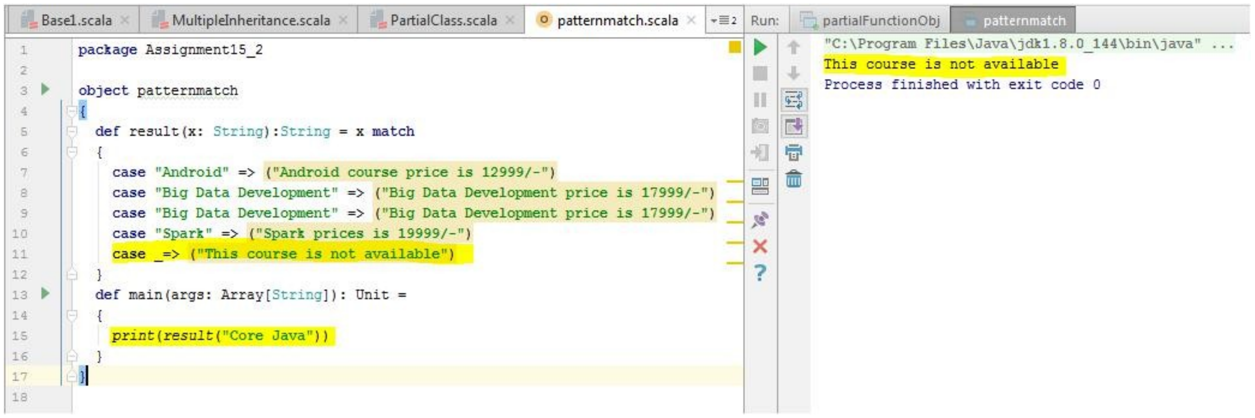
```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Big Data Development price is 17999/-
Process finished with exit code 0
```



```
package Assignment15_2

object patternmatch
{
  def result(x: String):String = x match
  {
    case "Android" => ("Android course price is 12999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Spark" => ("Spark prices is 19999/-")
    case _=> ("This course is not available")
  }
  def main(args: Array[String]): Unit =
  {
    print(result("Android"))
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Android course price is 12999/-
Process finished with exit code 0
```

When we provide any other course, example Core Java we will get the default value as result like below.



```
package Assignment15_2

object patternmatch
{
  def result(x: String):String = x match
  {
    case "Android" => ("Android course price is 12999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Big Data Development" => ("Big Data Development price is 17999/-")
    case "Spark" => ("Spark prices is 19999/-")
    case _=> ("This course is not available")
  }
  def main(args: Array[String]): Unit =
  {
    print(result("Core Java"))
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
This course is not available
Process finished with exit code 0
```