Scala Assignment 15.2

For this assignment, I've used Intellij Idea scala.

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

Steps followed :

1) Created a function, which takes 3 Integers input and adds it, as below.

```
def adder(i: Int,j: Int,k:Int):Int = {
  return i + j + k
}
```

2) Then created a partial function, add2, by calling the adder function in which one number is a constant and another 2 numbers are passed as input.

```
val add2=adder(_ :Int,_ :Int,20);
```

3) Input can be passed to add2 as add2(20,30) , which will return the output of 70.

```
println("Calling add2 directly: "+add2(20,30));
```

   Ouput printed in console:

```
Calling add2 directly: 70
```

4) Now Created another method as below, which takes, a function with 2 Integer arguments as input, and the argument function is executed internally. The ouput of the function is stored as 'abc'

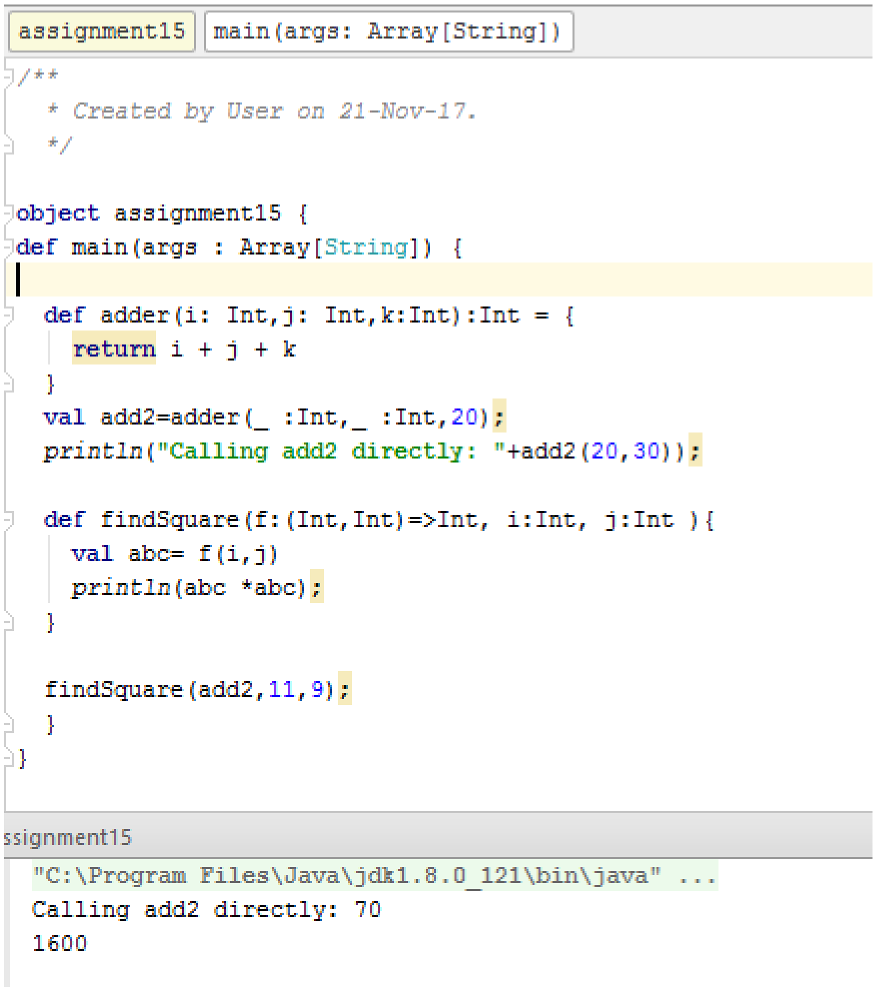
   Finally squaring the output and printing the abc * abc as below.

```
def findSquare(f:(Int,Int)=>Int, i:Int, j:Int ){
  val abc= f(i,j)
  println(abc *abc);
}
```

5) We can call the findsquare by passing partialargument method name, input1, input2 as arguments. Tested the program with various inputs as below.

Input Passed	***Output printed in the console***
`findSquare(add2,3,2);`	625
`findSquare(add2,6,3);`	841
`findSquare(add2,11,9);`	1600

Screenshot of IntelliJ and output:



```
/**
  * Created by User on 21-Nov-17.
  */

object assignment15 {
def main(args : Array[String]) {

  def adder(i: Int,j: Int,k:Int):Int = {
    return i + j + k
  }
  val add2=adder(_ :Int,_ :Int,20);
  println("Calling add2 directly: "+add2(20,30));

  def findSquare(f:(Int,Int)=>Int, i:Int, j:Int ){
    val abc= f(i,j)
    println(abc *abc);
  }

  findSquare(add2,11,9);
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Calling add2 directly: 70
1600
```

Source code for course Enquiry case match:

```
trait Drawable {
  def draw
  def paint
}
trait Cartoonist extends Drawable {
  override def draw() { println("Bang!") }
  override def paint(){ println("I need this to be painted")}
}
```

```
trait Artist extends Drawable {
  override def draw() { println("A pretty painting") }
  override def paint(){ println("I don't want this to be painted")}
}
//child class extending 2 parent traits
class CartoonistArtist extends Cartoonist with Artist{
  override def paint = super[Cartoonist].paint
}
object assignment_15_12 extends App {
   (new CartoonistArtist()).draw()
   (new CartoonistArtist()).paint()
}
```

4. 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..

Steps followed :

1) Created a case class as below

```
case class Acadgild(course: String)
```

When we declare a case class the Scala compiler does the following for you:

6) Creates a class and its companion object.
7) Implements the apply method that we can use as a factory. This lets us to create instances of the class without the new keyword. E.g.:
   Val p = new Acadgild("Spark");

2) Created the below method, which takes, Instance of acadgild course as input, and search for matching case. If matching case is found it will return the price.

```
def courseName(course: Acadgild) = course match {
  case Acadgild("Android") => 12999
  case Acadgild("Big Data Development") => 17999
  case Acadgild("Big Data Development") => 17999
  case Acadgild("Spark") => 19999
  case _ => "The specified course is not available"
}
```

3) Tested the program with various inputs as below.

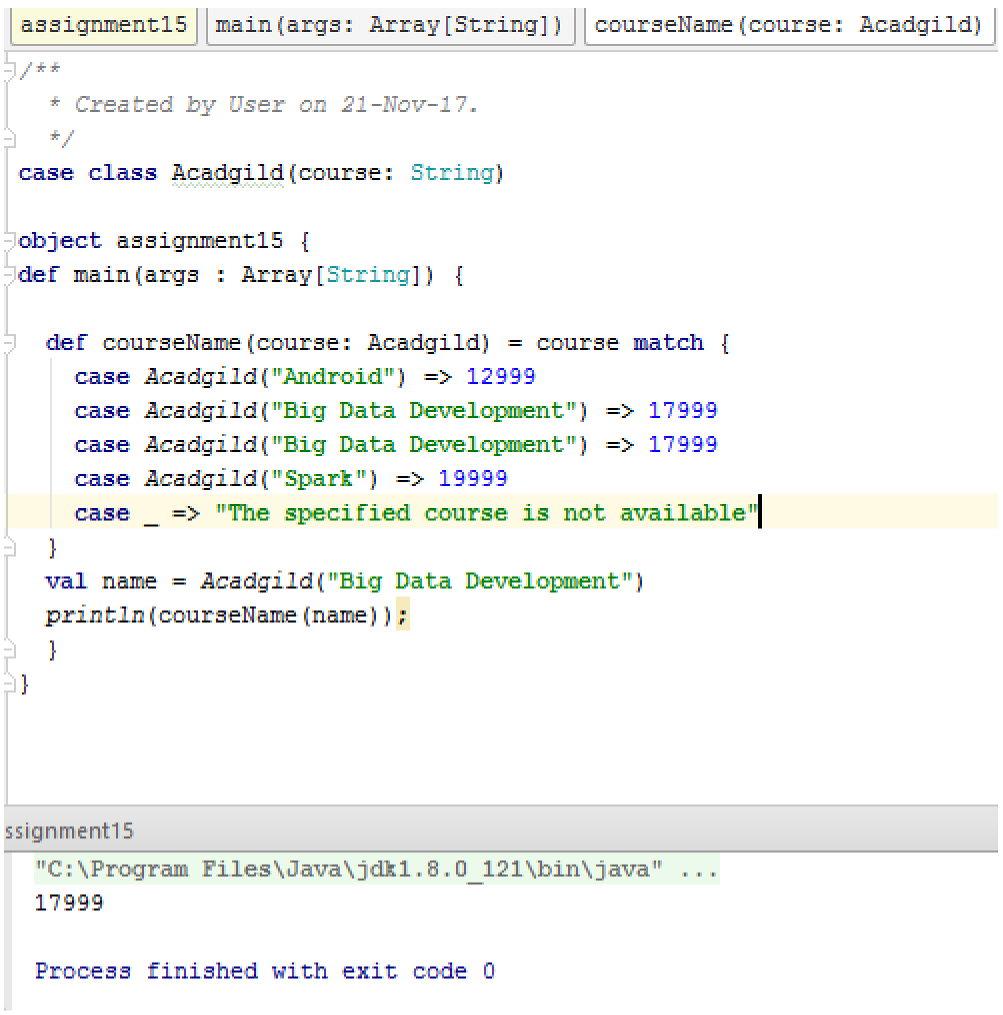
Input Passed	***Output printed in the console***

`val name = Acadgild("Big Data Development")`
`println(courseName(name));`
	17999

`val name = Acadgild("abc")`
`println(courseName(name));`
	The specified course is not available

`val name = Acadgild("Spark")`
`println(courseName(name));`
	19999

Screenshot of IntelliJ and output:



```
assignment15 | main(args: Array[String]) | courseName(course: Acadgild)
/**
  * Created by User on 21-Nov-17.
  */
case class Acadgild(course: String)

object assignment15 {
def main(args : Array[String]) {

  def courseName(course: Acadgild) = course match {
    case Acadgild("Android") => 12999
    case Acadgild("Big Data Development") => 17999
    case Acadgild("Big Data Development") => 17999
    case Acadgild("Spark") => 19999
    case _ => "The specified course is not available"
  }
  val name = Acadgild("Big Data Development")
  println(courseName(name));
  }
}
```

```
ssignment15
 "C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
 17999

 Process finished with exit code 0
```

Source code for course Enquiry case match:

```
trait Drawable {
  def draw
  def paint
}
trait Cartoonist extends Drawable {
  override def draw() { println("Bang!") }
  override def paint(){ println("I need this to be painted")}
}
trait Artist extends Drawable {
  override def draw() { println("A pretty painting") }
  override def paint(){ println("I don't want this to be painted")}
}
//child class extending 2 parent traits
class CartoonistArtist extends Cartoonist with Artist{
  override def paint = super[Cartoonist].paint
}
object assignment_15_12 extends App {
  (new CartoonistArtist()).draw()
  (new CartoonistArtist()).paint()
}
```