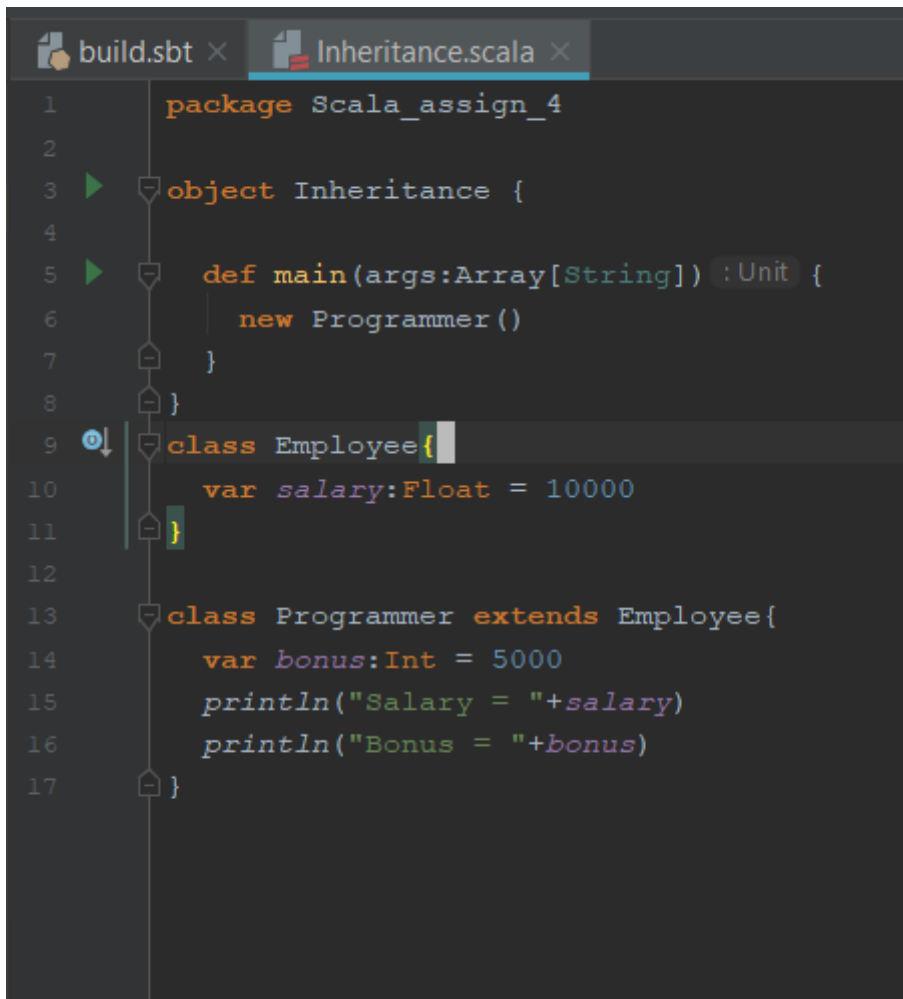


Assignment -17

Task 1:

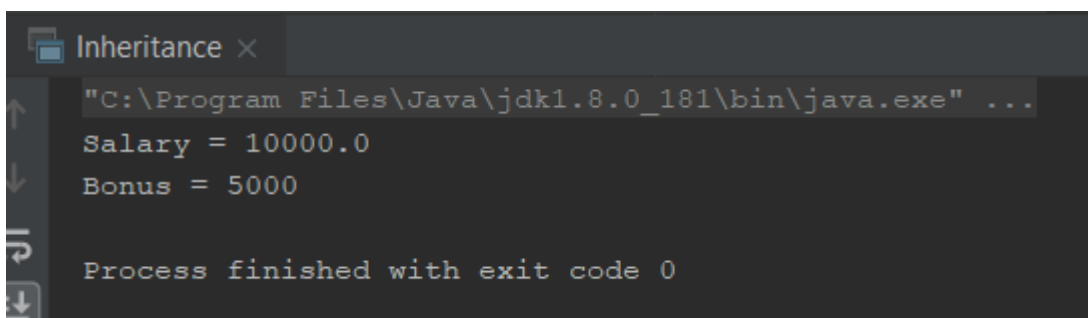
Write a simple program to show inheritance in scala.

Scala code to show inheritance is shown the below screenshot.



```
1 package Scala_assign_4
2
3 object Inheritance {
4
5     def main(args:Array[String]) :Unit {
6         new Programmer()
7     }
8 }
9 class Employee{
10     var salary:Float = 10000
11 }
12
13 class Programmer extends Employee{
14     var bonus:Int = 5000
15     println("Salary = "+salary)
16     println("Bonus = "+bonus)
17 }
```

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



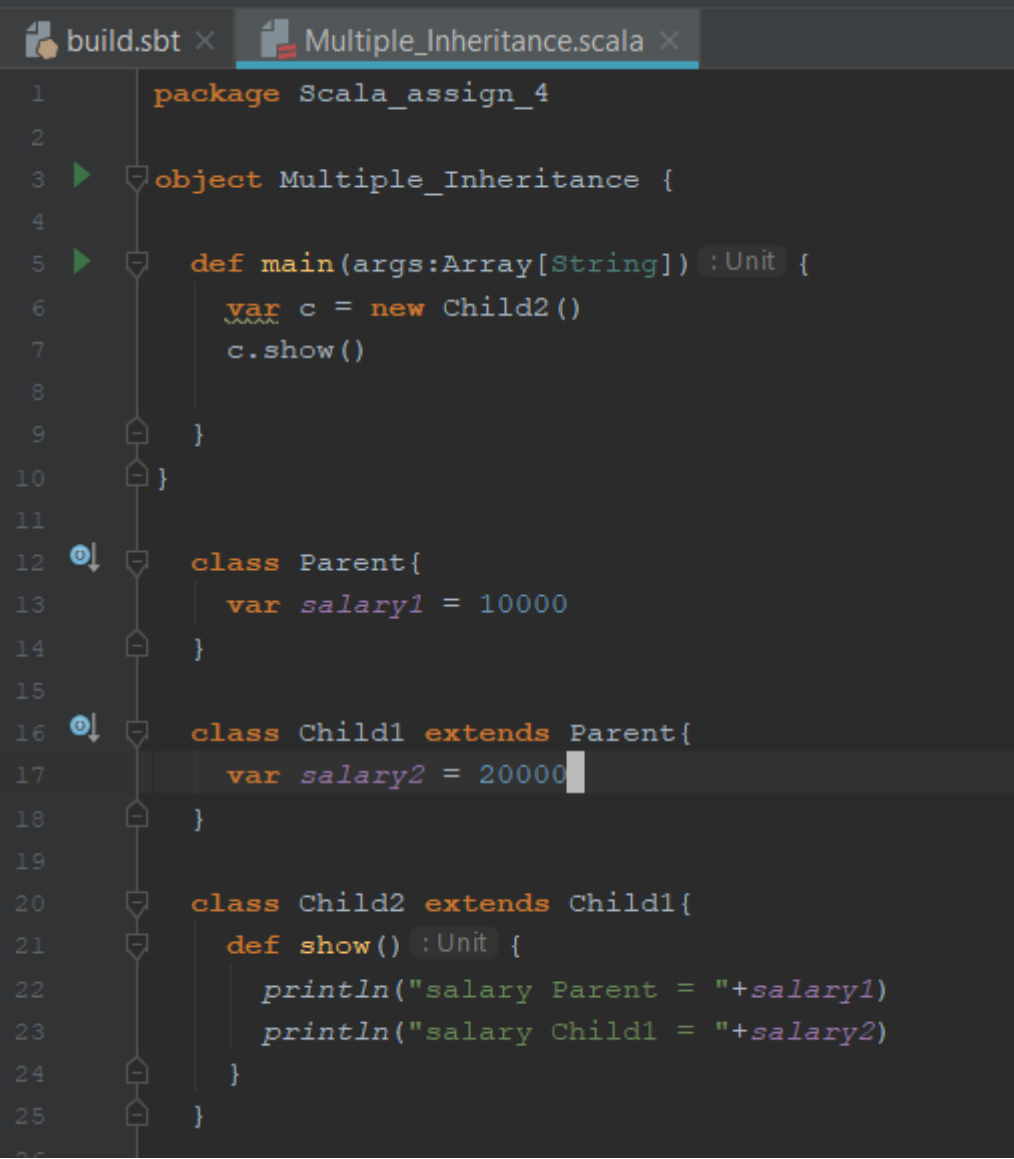
```
Inheritance x
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Salary = 10000.0
Bonus = 5000

Process finished with exit code 0
```

Task 2

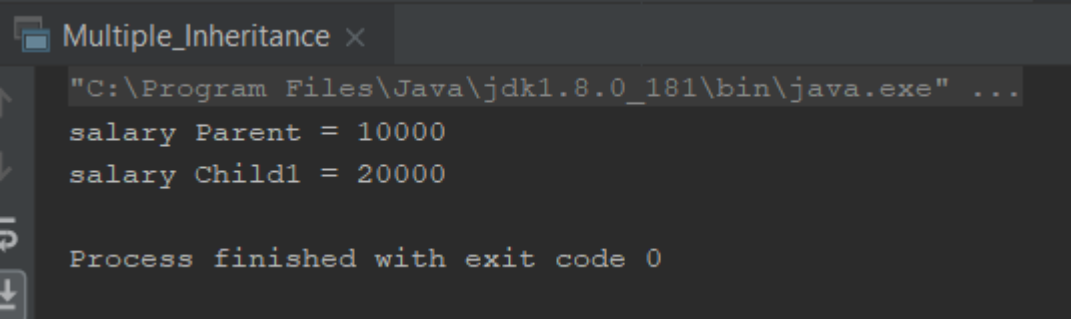
Write a simple program to show multiple inheritance in scala

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



```
1 package Scala_assign_4
2
3 object Multiple_Inheritance {
4
5     def main(args:Array[String]) :Unit {
6         var c = new Child2()
7         c.show()
8     }
9 }
10
11
12 class Parent{
13     var salary1 = 10000
14 }
15
16 class Child1 extends Parent{
17     var salary2 = 20000
18 }
19
20 class Child2 extends Child1{
21     def show() :Unit {
22         println("salary Parent = "+salary1)
23         println("salary Child1 = "+salary2)
24     }
25 }
```

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



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

Process finished with exit code 0
```

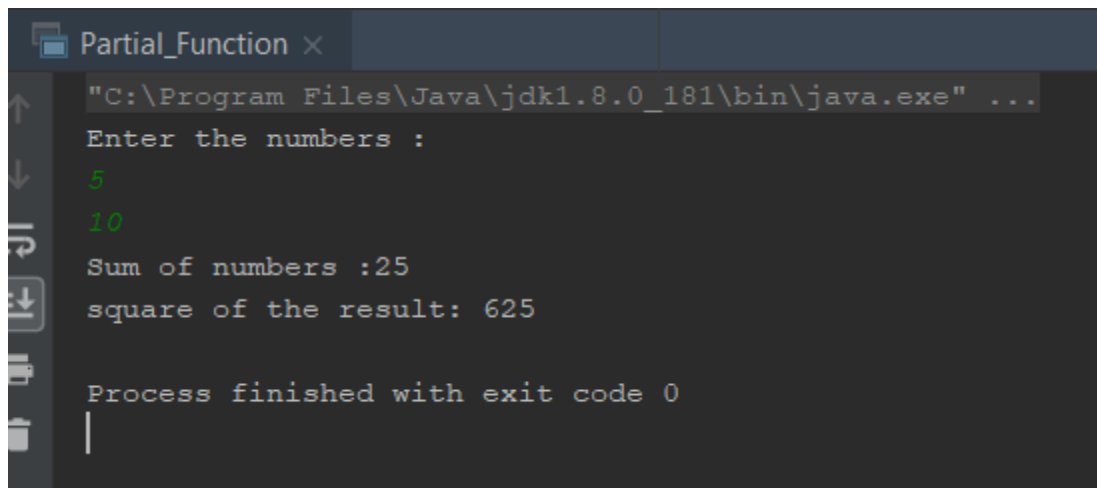
Task 3:

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.

```
build.sbt x Partial_Function.scala x
1 package Scala_assign_4
2
3 object Partial_Function {
4
5     def main(args:Array[String]): Unit = {
6
7         println("Enter the numbers :")
8
9         var x : Int = scala.io.StdIn.readInt()
10        var y : Int = scala.io.StdIn.readInt()
11
12        new PartialFunction().partialsum(x,y)
13    }
14 }
15 class PartialFunction {
16     def summation(a:Int,b:Int,c:Int) :Int = a+b+c
17
18     def partialsum(x:Int, y:Int) :Unit {
19
20         val add = summation(_:Int, _:Int, 10)
21         println("Sum of numbers : " + add(x,y))
22
23         def squareResult(result: Int) :Int = result * result
24
25         val square = squareResult(add(x,y))
26         println("square of the result: " + square)
27     }
28 }
```

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



```
Partial_Function x
"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
|
```

The screenshot shows a terminal window titled "Partial_Function x". The command prompt shows the Java executable path: "C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ... The program prompts the user to "Enter the numbers :". The user enters "5" and "10" on separate lines. The program then outputs "Sum of numbers :25" and "square of the result: 625". Finally, it displays "Process finished with exit code 0" and a cursor is visible on the next line.

Task 4

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

Android App Development -14,999 INR

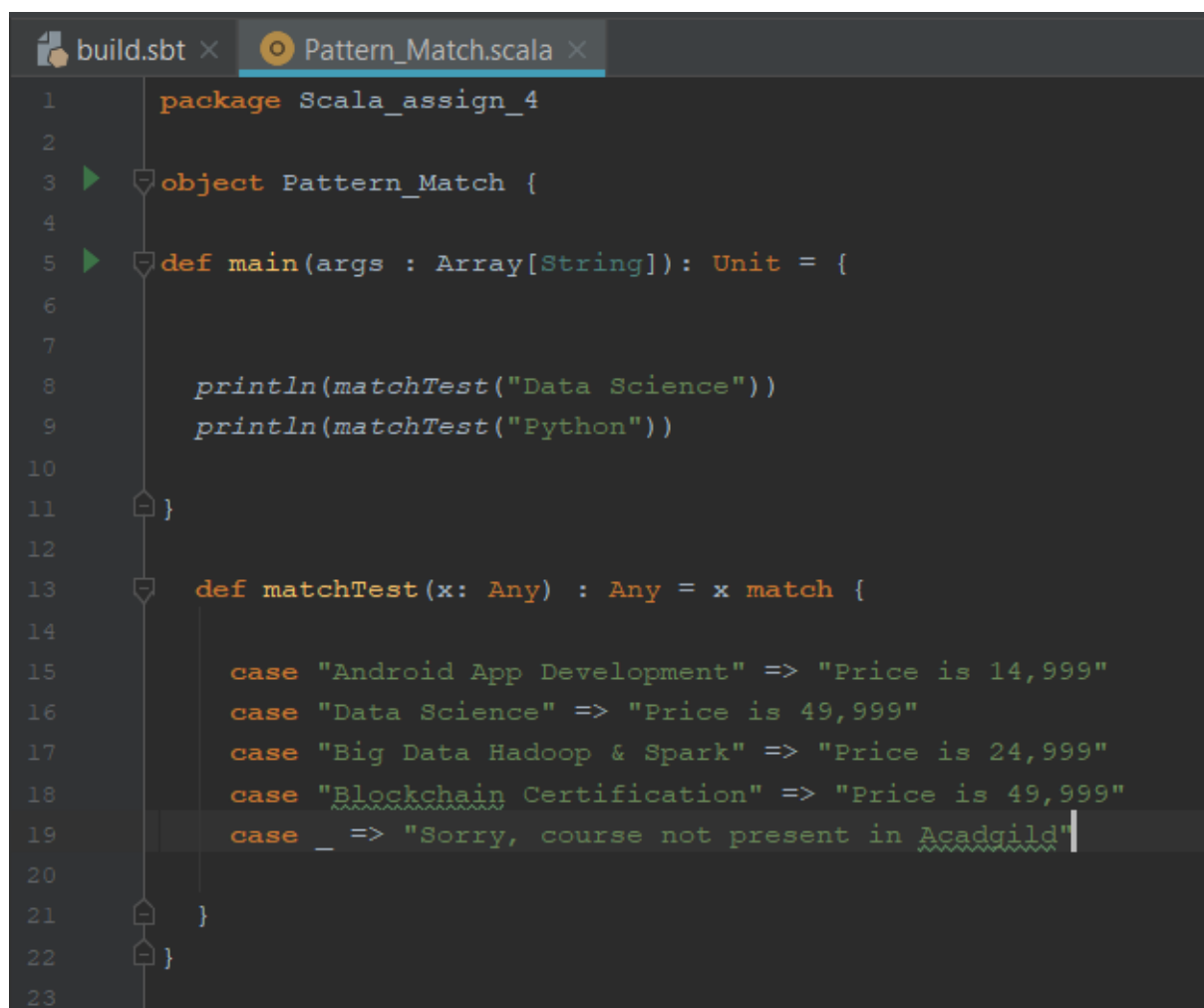
Data Science - 49,999 INR

Big Data Hadoop & Spark Developer – 24,999 INR

Blockchain Certification – 49,999 INR

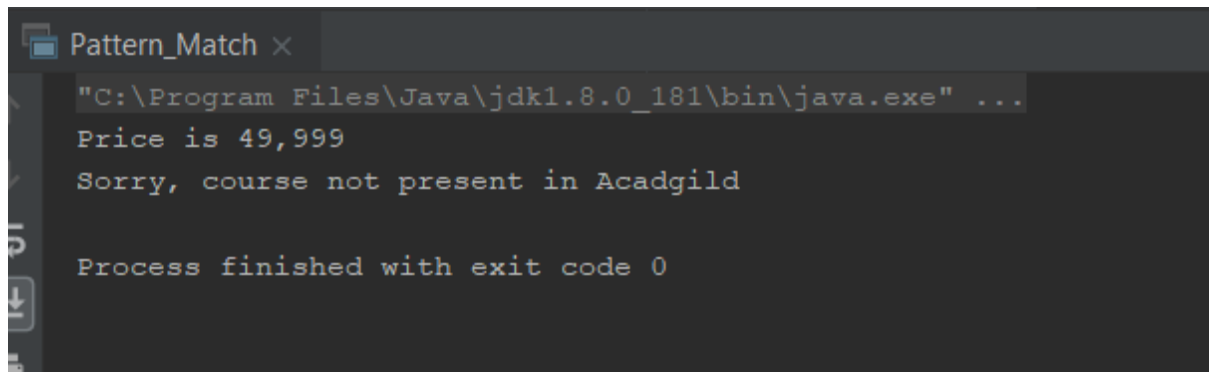
using match and add a default condition if the user enters any other course.

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



```
1 package Scala_assign_4
2
3 object Pattern_Match {
4
5 def main(args : Array[String]): Unit = {
6
7
8     println(matchTest("Data Science"))
9     println(matchTest("Python"))
10
11 }
12
13 def matchTest(x: Any) : Any = x match {
14
15     case "Android App Development" => "Price is 14,999"
16     case "Data Science" => "Price is 49,999"
17     case "Big Data Hadoop & Spark" => "Price is 24,999"
18     case "Blockchain Certification" => "Price is 49,999"
19     case _ => "Sorry, course not present in Acadgild"
20
21 }
22
23 }
```

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

A screenshot of a Java IDE's console window. The window has a tab labeled 'Pattern_Match' with a close button. The console shows the execution of a Java program. The first line is the command prompt: `"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...`. The program outputs two lines: `Price is 49,999` and `Sorry, course not present in Acadgild`. The final line indicates the program completed successfully: `Process finished with exit code 0`.

```
"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
```