

Task 1

Write a simple program to show inheritance in scala.

The screenshot shows the IntelliJ IDEA interface with the following details:

- File Menu:** File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
- Breadcrumbs:** ScalaTutorial > src > main > scala > Assignment17.scala
- Code Editor:** The file Assignment17.scala contains the following Scala code:

```
1 class Employee{  
2     var salary : Float = 10000  
3 }  
4  
5 class Accountant extends Employee{  
6     var Allowance: Int = 5000  
7     var AccountantSalary = salary + Allowance  
8     println("Accountant Salary = " + AccountantSalary)  
9 }  
10  
11 object AccountantSalary {  
12     def main(args:Array[String]) :Unit {  
13         new Accountant()  
14     }  
15 }
```
- Run Tab:** The run configuration is set to "AccountantSalary". The output shows:

```
"C:\Program Files\Java\jdk1.8.0_171\bin\java.exe" ...  
Accountant Salary = 15000.0  
Process finished with exit code 0
```

Task 2

Write a simple program to show multiple inheritance in scala

The screenshot shows the IntelliJ IDEA interface with the following details:

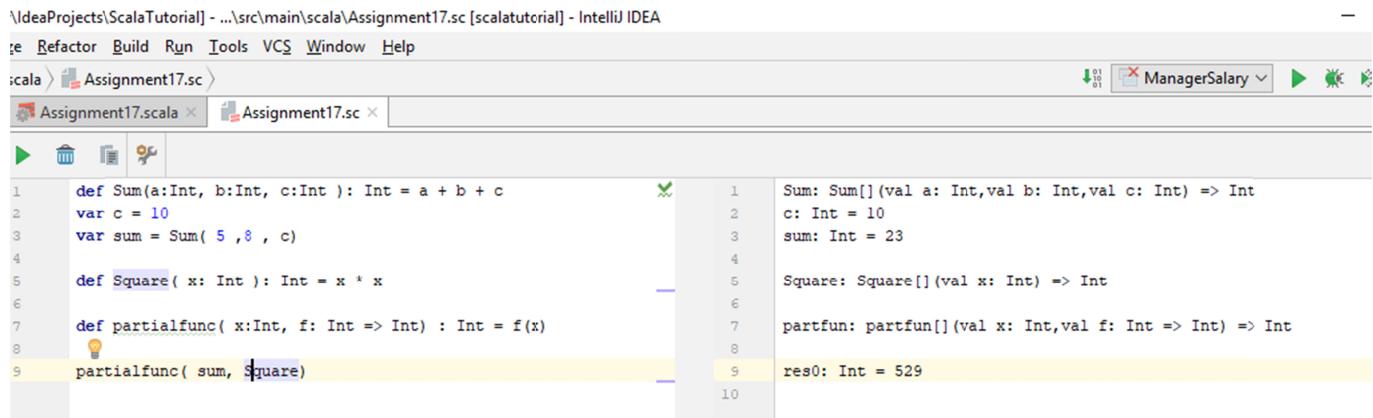
- Project Structure:** The left sidebar shows the project structure under "ScalaTutorial". The "Assignment17.scala" file is open.
- Code Editor:** The main window displays the Scala code:

```
1  class Employee{  
2      var salary : Float = 10000  
3  }  
4  
5  class Accountant extends Employee{  
6      var Allowance: Int = 5000  
7      var AccountantSalary = salary + Allowance  
8  }  
9  
10 class Manager extends Accountant {  
11     var Bonus:Int = 2500  
12     def show() :Unit {  
13         println("Manager salary = "+AccountantSalary)  
14         println("Manager Bonus = "+Bonus)  
15     }  
16 }  
17 object ManagerSalary {  
18     def main(args: Array[String]): Unit = {  
19         var Manager = new Manager()  
20         Manager.show()  
21     }  
22 }  
23
```
- Run Output:** The bottom panel shows the run output for the "ManagerSalary" task:

```
"C:\Program Files\Java\jdk1.8.0_171\bin\java.exe" ...  
Manager salary = 15000.0  
Manager Bonus = 2500  
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.



The screenshot shows the IntelliJ IDEA interface with a Scala project named 'Assignment17'. The code editor contains the following Scala code:

```
def Sum(a:Int, b:Int, c:Int ): Int = a + b + c
var c = 10
var sum = Sum( 5 ,8 , c)

def Square( x: Int ): Int = x * x

def partialfunc( x:Int, f: Int => Int) : Int = f(x)

partialfunc( sum, Square)
```

The code is annotated with line numbers 1 through 9. To the right of the code, the output window shows:

```
Sum: Sum[](val a: Int, val b: Int, val c: Int) => Int
c: Int = 10
sum: Int = 23

Square: Square[](val x: Int) => Int

partialfunc: partialfunc[](val x: Int, val f: Int => Int) => Int

res0: Int = 529
```

The line 'res0: Int = 529' is highlighted in yellow, indicating it is the result of the last evaluated expression.

Task 4

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

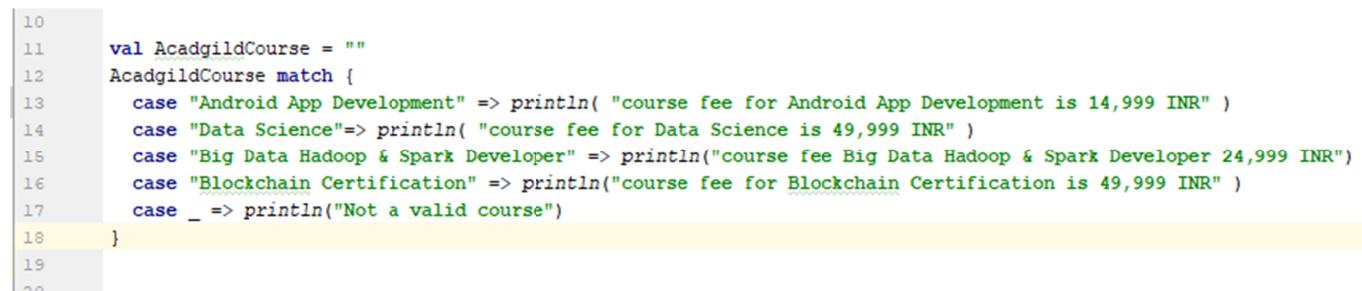
Android App Development -14,999 INR

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.



```
val AcadgildCourse = ""
AcadgildCourse match {
  case "Android App Development" => println( "course fee for Android App Development is 14,999 INR" )
  case "Data Science"=> println( "course fee for Data Science is 49,999 INR" )
  case "Big Data Hadoop & Spark Developer" => println("course fee Big Data Hadoop & Spark Developer 24,999 INR")
  case "Blockchain Certification" => println("course fee for Blockchain Certification is 49,999 INR" )
  case _ => println("Not a valid course")
}
```

Input : AcadgildCourse "Java"

```
10  
11 val AcadgildCourse = "Java"  
12 AcadgildCourse match {  
13   case "Android App Development" => println( "course fee for Android App D  
14   case "Data Science"=> println( "course fee for Data Science is 49,999 IN  
15   case "Big Data Hadoop & Spark Developer" => println("course fee Big Data  
16   case "Blockchain Certification" => println("course fee for Blockchain Ce  
17   case _ => println("Not a valid course")  
18 }  
19
```

10	AcadgildCourse: String = Java
11	Not a valid course
12	res0: Unit = ()
13	
14	
15	
16	
17	
18	
19	

Input AcadgildCourse "Data Science"

```
10  
11 val AcadgildCourse = "Data Science"  
12 AcadgildCourse match {  
13   case "Android App Development" => println( "course fee for Android App D  
14   case "Data Science"=> println( "course fee for Data Science is 49,999 IN  
15   case "Big Data Hadoop & Spark Developer" => println("course fee Big Data  
16   case "Blockchain Certification" => println("course fee for Blockchain Ce  
17   case _ => println("Not a valid course")  
18 }  
19
```

10	AcadgildCourse: String = Data Science
11	course fee for Data Science is 49,999 INR
12	res0: Unit = ()
13	
14	
15	
16	
17	
18	
19	