Session 17 Assignment 1

Task 1

Write a simple program to show inheritance in scala.

```
object Residence {
        class House {
                val doors: Int = 4
                val windows: Int = 20
                val address: String = "1234 Main Street, Charlotte, NC 20000"
                val rooms: Int = 13
        }

        class Room extends House {
                val room: String = "Kitchen"
                val inventor = ("Sink","Stove","Dishwasher","Cabinets")
        }

        def main(args: Array[String]): Unit = {
                val k = new Room()
                println(k.room)
                println(k.address)
                println(k.inventor)

        }


}
~
~
~
```

```
[acadgild@localhost Session17-ScalaIV]$ scalac Residence.scala
[acadgild@localhost Session17-ScalaIV]$ scala Residence
Kitchen
1234 Main Street, Charlotte, NC 20000
(Sink,Stove,Dishwasher,Cabinets)
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Session17-ScalaIV]$
```

Task 2

Write a simple program to show multiple inheritance in scala

```
object Appliance {
        class House {
                val doors: Int = 4
                val windows: Int = 20
                val address: String = "1234 Main Street, Charlotte, NC 20000"
                val rooms: Int = 13
        }

        class Room extends House {
                val room: String = "Kitchen"
                val inventor = ("Sink","Stove","Dishwasher","Cabinets")
        }

        class Appliances extends Room {
                val appliance: String ="Refrigerator"
                val voltage: Int = 120
                val age: Int = 5

        }

        def main(args: Array[String]): Unit = {
                val k = new Room()
                println("*** Inhertance ***")
                println("")
                println(k.room)
                println(k.address)
                println(k.inventor)

                println("")
                println("*** Multiple Inhertance ***")
                val r = new Appliances
                println(r.appliance)
                println(r.voltage)
                println(r.room)
                println(r.address)
                println("")

        }


}
~
~
~
```

```
[acadgild@localhost Session17-ScalaIV]$ scalac Appliance.scala
You have new mail in /var/spool/mail/acadgild
[acadgild@localhost Session17-ScalaIV]$
[acadgild@localhost Session17-ScalaIV]$ scala Appliance
*** Inhertance ***

Kitchen
1234 Main Street, Charlotte, NC 20000
(Sink,Stove,Dishwasher,Cabinets)

*** Multiple Inhertance ***
Refrigerator
120
Kitchen
1234 Main Street, Charlotte, NC 20000

[acadgild@localhost Session17-ScalaIV]$
```

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.

```
scala> val pf3 = (num1: Int, num2: Int) => num1 + num2 + 45
pf3: (Int, Int) => Int = $$Lambda$1213/1897193207@9129ab1

scala> pf3(3,5)
res14: Int = 53

scala> pf3(0,0)
res15: Int = 45

scala> val square = (input: Int) => input * input
square: Int => Int = $$Lambda$1214/1168145303@5d6bbb25

scala> square(pf3(0,0))
res16: Int = 2025

scala> square(pf3(2,3))
res17: Int = 2500

scala>
```

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> val Acadgild = Map("Android App Development" -> "14,999",
     | "Data Science" -> "49,999",
     | "Big Data Hadoop & Spark Developer" -> "24,999",
     | "Blockchain Certification" -> "49,999")
Acadgild: scala.collection.immutable.Map[String,String] = Map(Android App Develo
pment -> 14,999, Data Science -> 49,999, Big Data Hadoop & Spark Developer -> 24
,999, Blockchain Certification -> 49,999)

scala> def getAcadgild(course:String) : String = {
     | val mycourse = util.Try(Acadgild(course)) getOrElse "Unknown"
     | return mycourse
     | }
getAcadgild: (course: String)String

scala> getAcadgild("Data Science")
res18: String = 49,999

scala> getAcadgild("mcclain")
res19: String = Unknown

scala> getAcadgild("Android App Development")
res20: String = 14,999

scala>
```