Problem Statement

- Write a simple program to show inheritance in scala.
- Write a simple program to show multiple inheritances in scala.

Solution:

A simple program to show inheritance in scala.

Solution:

Inheritance is an object oriented concept which is used to reusability of code. You can achieve inheritance by using extends keyword. To achieve inheritance a class must extend to other class. A class which is extended called super or parent class. a class which extends class is called derived or base class.
Scala supports various types of inheritance including single, multilevel, multiple, and hybrid. You can use single, multilevel and hierarchal in your class. Multiple and hybrid can only be achieved by using traits.

Code:

Simple_Inhertiance.scala

```
package assignment15_1

/*
 * Showing Simple inheritance in Scala
 */


//Employee Class (Parent Class)

class Employee(emp_id:Int)
{
  val id :Int =emp_id
  def work() = println("Working")
}

//Part Time Employee Inheriting Employee Class (Child Class)

class Part_Time_Employee(emp_id:Int) extends Employee(emp_id)
{
  override val id :Int = emp_id
  override def work() = println("Working Part Time Only")
}

//Full Time Employee Inheriting Employee Class (Child Class)

class Full_Time_Employee(emp_id:Int) extends Employee(emp_id)
{
  override val id:Int =emp_id
  override def work() = println("Working Full Time")
}


object Simple_Inheritance
{
  def main(args: Array[String]): Unit = {


    val part__time_emp =  new Part_Time_Employee(10)
    val full__time_emp = new Full_Time_Employee(11)
```

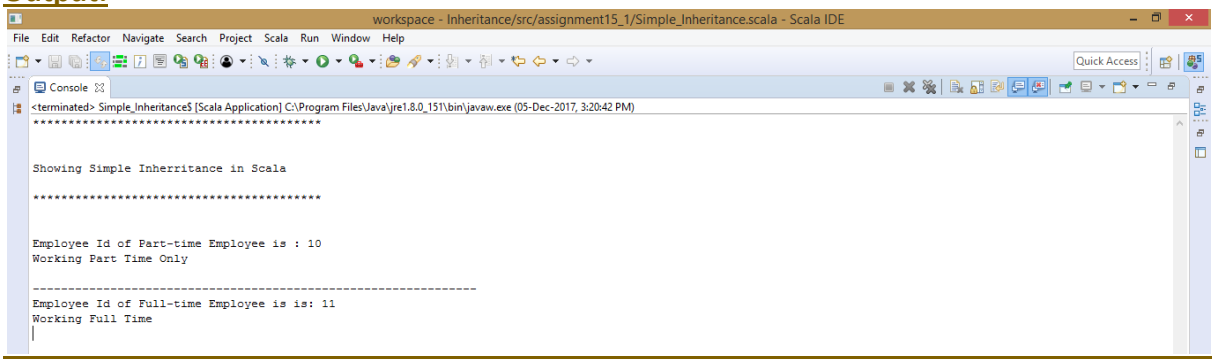
```
    println("****************************************\n")
    println("\nShowing Simple Inherritance in Scala \n")
    println("****************************************\n")
    println()

    println("Employee Id of Part-time Employee is : " + part__time_emp.id)
    part__time_emp.work()

    println()
    println("-------------------------------------------------------------")
    println("Employee Id of Full-time Employee is is: " +full__time_emp.id)
    full__time_emp.work()

  }
}
```

Output:



A simple program to show multiple inheritances in scala.

Solution:

A trait is like an interface with a partial implementation. In scala, trait is a collection of abstract and non-abstract methods. You can create trait that can have all abstract methods or some abstract and some non-abstract methods.

A variable that is declared either by using val or var keyword in a trait get internally implemented in the class that implements the trait. Any variable which is declared by using val or var but not initialized is considered abstract.

If a class extends a trait but does not implement the members declared in that trait, it must be declared abstract.

If a class implements multiple traits, it will extend the first trait, class, abstract class. with keyword is used to extend rest of the traits.

Code:

Multiple_Inheritance.scala

```
package assignment15_1
/*
 * Showing Multiple Inhertiance in Scala using Trait keyword
 */

trait Printable{
    def print()
}

trait Showable{
   def show()
}
```

```
class A6 extends Printable with Showable{
    def print(){
        println("This is printable")
    }
    def show(){
        println("This is showable");
    }
}
object Multiple_Inheritance
{
  def main(args:Array[String])
  {
        var a = new A6()
        println("----------------Showing Multiple Inheritance--------------
-----")
        println()
        println("------------------------------------------------------------
----")
        println()
        a.print()
        a.show()
    }
}
```

Output: