**Academic Year 2024-25 Even**

**19CSE313 – Principles of Programming Language**

**B.Tech CSE 2022-26 F Section**

**Practice Set 12 – Composition and Inheritance in Scala**

**Composition**

1. **Example 1 - compose**

scala> val add=(a: Int)=> a+1

scala> val mul=(a: Int)=>

| a\*2

scala> val sub=(a: Int) => a-1

scala> println((add compose mul)(2))

scala> println((mul compose add)(2))

scala> println((add compose mul compose sub)(2))

1. **Example 2 – andThen**

scala> println((add andThen mul)(2))

scala> println(mul(add (2)))

1. Describe briefly Scala's support for function composition, detailing the related keywords and constructs.
2. Define a function f1 that finds the absolute value of a given integer. Define a second function f2 that adds a constant value of 3 to the input integer. Using **andThen** construct, show the results of f1◦f2 and f2◦f1.
3. Solve the Qn.4 using **compose**
4. **Composing functions with two parameters – tupled**

scala> val f = (x: Int, y: Int) => (x + 1, y + 2)

scala> val g = (x: Int, y: Int) => x – y

scala> val h = f.tupled andThen g.tupled

scala> h(5,4)

1. **Another Example**

scala> val f1 = (x: Int, y: Int) => x + y

scala> val g1 = (x: Int) => x + 1

scala> val h1 = f1.tupled andThen g1

scala> h1(5,4)

scala> val h3 = g2.tupled andThen f2

scala> h3(5,6)

**Inheritance**

1. **Single Inheritance**

*Filename: singleInheritance.scala*

class Parent{

var parentName: String = "Vasudev"

}

class Child extends Parent{

var childName: String = "Krishna"

def details()={

println(parentName+ " is the father of "+childName);

}

}

object Main{

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

val ob = new Child();

ob.details();

}

}

*Execution: scala singleInheritance.scala*

1. **Multiple Inheritance**

*Source File: multipleInheritance.scala*

object Multiple{

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

trait A{

var length:Int= \_

def action={

length=length+5

}

}

trait B{

var height:Int = \_

def action={

height = height + 1

}

}

class C extends A with B{

length=3;

height+=6;

override def action={

super[A].action

super[B].action

}

}

var c=new C

c.action

println(c.height)

println(c.length)

}

}

*Execution: scala multipleInheritance.scala*

1. **Multilevel Inheritance**

scala> class A{

| var gp = "Grandparent A"

| }

scala> class B extends A{

| var p = "Parent B"

| }

scala> class C extends B{

| var c = "Child C"

| def display()={

| println(c+" has "+p+" and "+gp)

| }

| }

scala> val child = new C()

scala> child.display()

1. **Briefly explain Hierarchical and Hybrid Inheritance with proper examples**
2. **Another example**

scala> class university{

| var uname = "Amrita Vishwa Vidyapeetham"

| }

scala> class ASE extends university{

| var engg = "School of Engineering"

| }

scala> class ASC extends university{

| var comp = "School of Computing"

| }

scala> class CSE extends ASC{

| var dept = "Computer Science and Engineering"

| }

scala> class MEE extends ASE{

| var dept = "Mechanical Engineering"

| }

scala> val csef = new CSE()

scala> println(csef.dept+","+csef.comp+","+csef.uname)

scala> val meea = new MEE()

scala> println(meea.dept+","+meea.engg+","+meea.uname)

1. **Identify the different types of inheritance used in the above code.**
2. **Differentiate between the keywords class, object and trait in Scala**