Session 15

Assignment 1

|  |  |
| --- | --- |
| **Prepared For:** | AcadGild |
|  |  |
| **Document Approval:** | **AcadGild** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Project Title:** | Session 15 – Assignment 1 |
|  |  |
| **Prepared By:** | Duncan Burgess |
|  |  |
|  | dburgess@duncb.com |
|  |  |
| **Primary Engineer:** | Duncan Burgess |
|  |  |
| **Document Reference:** | **Session 15 – Assignment 1** |
|  |  |
| **Start Date:** | 10/10/2017 |
|  |  |
|  |  |



# 

# Contents

[Contents 2](#_Toc495776345)

[Change History 3](#_Toc495776346)

[1. Problem Statement 4](#_Toc495776347)

[2. Solution 4](#_Toc495776348)

[2.1. Write a simple program to show inheritance in scala. 4](#_Toc495776349)

[2.2. Write a simple program to show multiple inheritance in scala. 5](#_Toc495776350)

# Change History

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Document Revision** | **Date** | **Authored By** | **Authorised By** | **Sections Affected** | **Reason for Change** |
| Rev 01 | 13/10/2017 | Duncan Burgess |  | All | Initial release. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Problem Statement

* Write a simple program to show inheritance in scala.
* Write a simple program to show multiple inheritance in scala.

# Solution

## Write a simple program to show inheritance in scala.

**Code writen**

**object** moo **extends** App {

**abstract** **class** Animal(name: *String*) {

**def** speak = name + " says " + sound

**def** sound: *String*

}

**class** Cow(name: *String*) **extends** Animal(name) {

**override** **def** sound() = "mooooooooo"

}

**class** Horse(name: *String*) **extends** Animal(name) {

**override** **def** sound() = "Neeey"

}

**class** Sheep(name: *String*) **extends** Animal(name) {

**override** **def** sound() = "baaaa"

}

**var** h = **new** Horse("Dobin")

println(h.speak)

**var** c = **new** Cow("Ementrude")

println(c.speak)

println(**new** Sheep("Lamsie").speak)

}

**Results**

Dobin says Neeey

Ementrude says mooooooooo

Lamsie says baaaa

## Write a simple program to show multiple inheritance in scala.

Code written

**object** moomoo **extends** App {

**trait** Animal {

**def** sound: *String* = "base"

}

**trait** Cow **extends** Animal {

**override** **def** sound: *String* = "Moooo -> " + **super**.sound

}

**trait** Horse **extends** Animal {

**override** **def** sound: *String* = "Neyyy " + **super**.sound

}

**trait** bigCow **extends** Cow {

**override** **def** sound: *String* = "Big Mooooo-> " + **super**.sound

}

**class** A **extends** Cow **with** Horse { **override** **def** sound = **super**.sound }

**class** B **extends** Horse **with** Cow { **override** **def** sound = **super**.sound }

**class** C **extends** bigCow **with** Horse **with** Cow { **override** **def** sound = **super**.sound }

**class** D **extends** Horse **with** bigCow **with** Cow { **override** **def** sound = **super**.sound }

**val** a = **new** A().sound

println (a)

**val** b = **new** B().sound

println (b)

**val** c = **new** C().sound

println (c)

**val** d = **new** D().sound

println (d)

}

**Results**

Neyyy Moooo -> base

Moooo -> Neyyy base

Neyyy Big Mooooo-> Moooo -> base

Big Mooooo-> Moooo -> Neyyy base

I performed this to demonstrate multiple inheritance Scala linearization is a deterministic process that puts all traits in a linear inheritance hierarchy. By doing that it resolves the diamond problem.

It was a great way to learn and see what happened.