# Department of Computing

# CS 212: Object Oriented Programming

# Class: BSCS-8

# Lab 08: Interfaces

# Date: April 1,2019

# Instructor:

# Hirra Anwar

**Learning Objectives**

The learning objective of this lab is to understand and practice the concept of polymorphic behavior using interfaces.

**Task #1.**

Write a class Person.java followed implementing two interfaces, PersonInterface.java and AnotherPersonInterface.java.

C:\Users\anis\AppData\Local\Microsoft\Windows\INetCache\Content.Word\t1 (10).png

Once done defining classes use the following client class to test it.

package mypersoninterfaceproject**;**

public class Main **{**

public static void main**(**String**[]** args**)** **{**

// Create an object instance of Person class.

Person person1 **=** **new** Person**(**10000**,** 20000**,** "Quintin"**,** "Tarantino"**);**

// You can assign the object instance to

// PersonInterface type.

PersonInterface personinterface1 **=** person1**;**

// Display data from person1 and personinterface1.

// Observe that they refer to the same object instance.

System**.**out**.**println**(**

"person1.getName() = " **+** person1**.**getName**()** **+** "," **+**

" person1.computeTotalWealth() = " **+**

person1**.**computeTotalWealth**()** **+** "," **+**

" person1.measureIntelligence() = " **+**

person1**.**measureIntelligence**(**person1**.**getName**()));**

System**.**out**.**println**(**

"personinterface1.getName() = " **+** personinterface1**.**getName**()** **+** "," **+**

" personinterface1.computeTotalWealth() = " **+**

personinterface1**.**computeTotalWealth**());**

// You can assign the object instance to

// AnotherPersonInterface type.

AnotherPersonInterfaceExample anotherpersoninterface1 **=** person1**;**

// Check of object instance that is referred by personinterface1 and

// anotherpersoninterface1 is the same object instance.

boolean b1 **=** **(**personinterface1 **==** anotherpersoninterface1**);**

System**.**out**.**println**(**"Do personinterface1 and anotherpersoninterface1 point to the same object instance? " **+** b1**);**

**}**

**}**

Running the test should result in the following output.

person1.getName() = Quintin Tarantino, person1.computeTotalWealth() = 30000, person1.measureIntelligence() = 50

personinterface1.getName() = Quintin Tarantino, personinterface1.computeTotalWealth() = 30000

Do personinterface1 and anotherpersoninterface1 point to the same object instance? true

**Bonus.** Compile error is expected on the following line of code. What could be causing it?

personinterface1.measureIntelligence(personinterface1.getName());

**Task #2.**

Your task is to write MyRelation program by referring to the UML class diagram below.

C:\Users\anis\AppData\Local\Microsoft\Windows\INetCache\Content.Word\t1 (11).png

Once done with the class definitions use the following tester class to confirm its working.

|  |
| --- |
| package myrelationinterfaceproject**;**  public class Main **{**    public static void main**(**String**[]** args**)** **{**    // Create two Line object instances.  Line line1 **=** **new** Line**(**1.0**,** 2.0**,** 3.0**,** 4.0**);**  Line line2 **=** **new** Line**(**2.0**,** 3.0**,** 7.0**,** 5.0**);**    boolean b1 **=** line1**.**isGreater**(**line1**,** line2**);**  System**.**out**.**println**(**"line1 is greater than line2: " **+** b1**);**  boolean b2 **=** line1**.**isEqual**(**line1**,** line2**);**  System**.**out**.**println**(**"line1 is equal with line2: " **+** b2**);**    // Note that the line3 is object instance of Line type.  // Because the Line type is also a type of RelationInterface,  // the line3 variable can be declared as RelationInterface type.  // This is a very very important concept you need to understand.  RelationInterface line3 **=** **new** Line**(**1.0**,** 5.0**,** 7.0**,** 9.0**);**  boolean b3 **=** line3**.**isEqual**(**line1**,** line3**);**  System**.**out**.**println**(**"line1 is equal with line3: " **+** b3**);**    System**.**out**.**println**(**"Length of line1 is " **+** line1**.**getLength**());**  System**.**out**.**println**(**"Length of line2 is " **+** line2**.**getLength**());**  **}**  **}** |

The test should result in the output that resembles the following.

line1 is greater than line2: <true/false>

line1 is equal with line2: <true/false>

line1 is equal with line3: <true/false>

Length of line1 is <length>

Length of line2 is <length>

a) What happens when the following line is added to the Main class? Identify the reason.

System.out.println("Length of line3 is " + line3.getLength());

**b) Change the method definitions and pass only one argument in each method. Observe what changes need to be made.**

c) Have a look at the Comparable Interface example given at the following link:

<https://www.javatpoint.com/Comparable-interface-in-collection-framework>

and implement the Comparable Interface for the Line class.

**Task #3.**

This task has been done as your class activity previously and the lab you need to implement it to check what deficiencies in your activity code were there to improve those.

Your task is to revise the MyOnlineShop program done in the previous lab using the UML class diagram below.

C:\Users\anis\AppData\Local\Microsoft\Windows\INetCache\Content.Word\t1 (12).png

Once done with the class definitions use the following tester class to confirm its working.

|  |
| --- |
| package myonlineshopusinginterface**;**  public class Main **{**    public Main**()** **{}**    public static void main**(**String**[]** args**)** **{**    // Declare and create Product array of size 5  ProductInterface**[]** pa **=** **new** Product**[**5**];**    // Create object instances and assign them to the type of Product.  pa**[**0**]** **=** **new** TV**(**1000**,** "Samsung"**,** 30**);**  pa**[**1**]** **=** **new** TV**(**2000**,** "Sony"**,** 50**);**  pa**[**2**]** **=** **new** MP3Player**(**250**,** "Apple"**,** "blue"**);**  pa**[**3**]** **=** **new** Book**(**34**,** "Sun press"**,** 1992**);**  pa**[**4**]** **=** **new** Book**(**15**,** "Korea press"**,** 1986**);**    // Compute total regular price and total sale price.  double totalRegularPrice **=** 0**;**  double totalSalePrice **=** 0**;**    **for** **(**int i**=**0**;** i**<**pa**.**length**;** i**++){**    // Call a method of the super class to get the regular price.  totalRegularPrice **+=** pa**[**i**].**getRegularPrice**();**    // Since the sale price is computed differently  // depending on the product type, overriding (implementation)  // method of the object instance of the sub-class  // gets invoked. This is runtime polymorphic behavior.  totalSalePrice **+=** pa**[**i**].**computeSalePrice**();**    System**.**out**.**println**(**"Item number " **+** i **+**  ": Type = " **+** pa**[**i**].**getClass**().**getName**()** **+**  ", Regular price = " **+** pa**[**i**].**getRegularPrice**()** **+**  ", Sale price = " **+** pa**[**i**].**computeSalePrice**());**  **}**  System**.**out**.**println**(**"totalRegularPrice = " **+** totalRegularPrice**);**  System**.**out**.**println**(**"totalSalePrice = " **+** totalSalePrice**);**  **}**  **}** |

The test should result in the following output.

Item number 0: Type = myonlineshopusinginterface.TV, Regular price = 1000.0, Sale price = 800.0

Item number 1: Type = myonlineshopusinginterface.TV, Regular price = 2000.0, Sale price = 1600.0

Item number 2: Type = myonlineshopusinginterface.MP3Player, Regular price = 250.0, Sale price = 225.0

Item number 3: Type = myonlineshopusinginterface.Book, Regular price = 34.0, Sale price = 17.0

Item number 4: Type = myonlineshopusinginterface.Book, Regular price = 15.0, Sale price = 7.5

totalRegularPrice = 3299.0

totalSalePrice = 2649.5

**Deliverable**

Compile a single word file and upload on LMS