# Department of Computing

# CS 212: Object Oriented Programming

# Class: BESE-11 AB

# Lab 12: Polymorphism

# Date: May 25th , 2021

# 

# Instructor:

# Ms. Hania Aslam

# Lab 12: Polymorphism

**Learning Objectives**

The learning objective of this lab is to understand and practice the concept of polymorphism, a very powerful feature of OOP which helps in code extensibility.

**Warm-up.**

Consider there is a hair dresser, a surgeon and an actor. If you ask each of them to 'cut': the hair dresser starts cutting hair, the surgeon makes an incision, and the actor stops acting out the scene. This is polymorphism; same command but yields different behavior for each object.

Now your self-review task is to give a simple real life example of polymorphism!!

**Task #1.**

Write an abstract class LivingThing.java followed by two concrete classes, Human.java and Monkey.java, extending the abstract class.

C:\Users\Its\Downloads\t1 (18).png

Once done defining classes use the following client class to test the functionality of the implemented classes.

package myabstractclassproject**;**

public class Main **{**

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

// Create Human object instance

// and assign it to Human type.

Human human1 **=** **new** Human**(** "Will Rodman"**);**

human1**.**walk**();**

// Create Human object instance

// and assign it to LivingThing type.

LivingThing livingthing1 **=** human1**;**

livingthing1**.**walk**();**

// Create a Monkey object instance

// and assign it to LivingThing type.

LivingThing livingthing2 **=** **new** Monkey**(** "Caesar"**);**

livingthing2**.**walk**();**

// Display data from human1 and livingthing1.

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

System**.**out**.**println**(** "human1.getName() = " **+** human1**.**getName**());**

System**.**out**.**println**(** "livingthing1.getName() = " **+** livingthing1**.**getName**());**

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

// y is the same object instance.

boolean b1 **=** **(** human1 **==** livingthing1**);**

System**.**out**.**println**(** "Do human1 and livingthing1 point to the same object instance? " **+** b1**);**

**}**

**}**

Running the test should result in the following output. Observe the results carefully to enhance your understanding!!

Human Will Rodman walks...

Human Will Rodman walks...

Monkey Caesar also walks...

human1.getName() = Will Rodman

livingthing1.getName() = Will Rodman

Do human1 and livingthing1 point to the same object instance? true

**Code:**

**Class LivingThing:**

package com.company;  
  
*//Creating abstract class*public abstract class *LivingThing* {  
 *//Declaring fields* private String name;  
  
 *//Defining the constructor* LivingThing(String *name*){  
 this.name = *name*;  
 }  
  
 *//Defining required methods* public void breath(){  
 System.*out*.println("Living thing breathing....");  
 }  
 public void eat(){  
 System.*out*.println("Living thing eating....");  
 }  
 public abstract void walk(); *//Abstract method* public String getName(){  
 return name;  
 }  
 public void setName(String *name*){  
 this.name = *name*;  
 }  
  
  
}

**Class Human:**

package com.company;  
  
*//Defining the child class Human*public class Human extends *LivingThing*{  
  
 *//Defining the constructor* Human(String *name*){  
 super(*name*);  
 }  
  
 *//Defining the abstract method* public void walk(){  
 System.*out*.println("Human " + this.getName() + " walking....");  
 }  
}

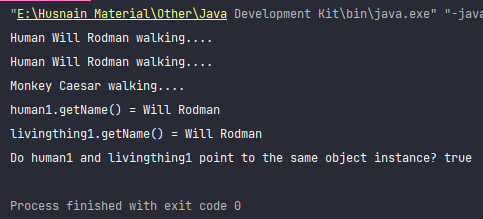
**Class Monkey:**

package com.company;  
  
*//Defining the child class Monkey*public class Monkey extends *LivingThing*{  
  
 *//Defining the constructor* Monkey(String *name*){  
 super(*name*);  
 }  
  
 *//Defining the abstract mehtod* public void walk(){  
 System.*out*.println("Monkey " + this.getName() + " walking....");  
 }  
  
}

**Test Class:**

package com.company;  
  
*//Defining the test class*public class TestTask01 {  
  
 *//Main method* public static void main( String[] *args*) {  
 *// Create Human object instance  
 // and assign it to Human type.* Human human1 = new Human( "Will Rodman");  
 human1.walk();  
  
 *// Create Human object instance  
 // and assign it to LivingThing type.  
 LivingThing* livingthing1 = human1;  
 livingthing1.walk();  
  
 *// Create a Monkey object instance  
 // and assign it to LivingThing type.  
 LivingThing* livingthing2 = new Monkey( "Caesar");  
 livingthing2.walk();  
  
 *// Display data from human1 and livingthing1.  
 // Observe that they refer to the same object instance.* System.*out*.println( "human1.getName() = " + human1.getName());  
 System.*out*.println( "livingthing1.getName() = " + livingthing1.getName());  
  
 *// Check of object instance that is referred by x and  
 // y is the same object instance.* boolean b1 = ( human1 == livingthing1);  
 System.*out*.println( "Do human1 and livingthing1 point to the same object instance? " + b1);  
 }  
 }

**Output Screenshot:**

****

**Bonus.** What happens when you create a LivingThing object in the Main class? For example using the statement,

LivingThing z = new LivingThing();

**Answer**

It results into an error which is as follows:

****

**Task #2.**

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

C:\Users\Its\Downloads\t1 (19).png

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

|  |
| --- |
| package myonlineshop**;**  public class Main **{**    public static void main**(**String**[]** args**)** **{**    // Declare and create Product array of size 5  Product**[]** 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 = myonlineshop.TV, Regular price = 1000.0, Sale price = 800.0

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

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

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

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

totalRegularPrice = 3299.0

totalSalePrice = 2649.5

**Code:**

**Class Product:**

package com.company;  
  
*//Creating the abstract class*public abstract class *Product* {  
 *//Defining fields* private double regularPrice;  
  
 *//Defining the constructor* Product(double *regularPrice*){  
 this.regularPrice = *regularPrice*;  
 }  
 *//Defining the required methods* public abstract double computeSalePrice(); *//Abstract method*

public void setRegularPrice(double *regularPrice*){  
 this.regularPrice = *regularPrice*;  
 }  
 public double getRegularPrice(){  
 return regularPrice;  
 }  
}

**Class Book:**

package com.company;  
  
*//Defining the child class Book*public class Book extends *Product*{  
 *//Defining the fields* private String publisher;  
 private int yearPublished;  
  
 *//Defining the constructor* Book(double *regularPrice*, String *publisher*, int *yearPublished*){  
 super(*regularPrice*);  
 this.publisher = *publisher*;  
 this. yearPublished = *yearPublished*;  
 }  
  
 *//Defining required methods* @Override  
 public double computeSalePrice() { *//Abstract method* return this.getRegularPrice()-((50\*this.getRegularPrice())/100);  
 }  
 public String getPublisher() {  
 return publisher;  
 }  
 public void setPublisher(String *publisher*) {  
 this.publisher = *publisher*;  
 }  
 public int getYearPublished() {  
 return yearPublished;  
 }  
 public void setYearPublished(int *yearPublished*) {  
 this.yearPublished = *yearPublished*;  
 }  
}

**Class Electronics:**

package com.company;  
  
*//Defining the child abstract class Electronics*public abstract class *Electronics* extends *Product*{  
 *//Defining fields* private String manufacturer;  
  
 *//Defining the constructor* Electronics(double *regularPrice*, String *manufacturer*){  
 super(*regularPrice*);  
 this.manufacturer = *manufacturer*;  
 }  
  
 *//Defining the required methods* public void setManufacturer(String *manufacturer*){  
 this.manufacturer = *manufacturer*;  
 }  
 public String getManufacturer(){  
 return manufacturer;  
 }  
}

**Class MP3Player:**

package com.company;  
  
*//Creating the child class*public class MP3Player extends *Electronics*{  
 *//Defining fields* private String color;  
  
 *//Defining the constructor* MP3Player(double *regularPrice*, String *manufacturer*, String *color*){  
 super(*regularPrice*, *manufacturer*);  
 this.color = *color*;  
 }  
  
 *//Defining the required methods* public double computeSalePrice(){ *//Abstract method* return this.getRegularPrice()-((10\*this.getRegularPrice())/100);  
 }  
 public void setColor(String *color*){  
 this.color = *color*;  
 }  
 public String getColor(){  
 return color;  
 }  
}

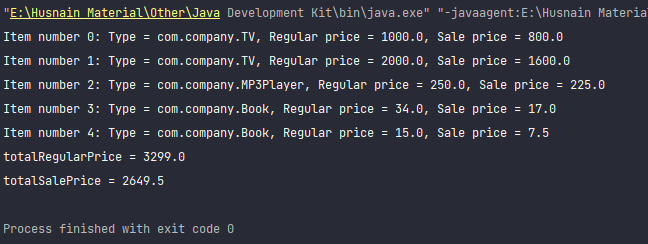
**Class TV:**

package com.company;  
  
*//Defining the child class TV*public class TV extends *Electronics*{  
 *//Declaring fields* private int size;  
  
 *//Defining the constructor* TV(double *regularPrice*, String *manufacturer*, int *size*){  
 super(*regularPrice*, *manufacturer*);  
 this.size = *size*;  
 }  
  
 *//Definiing required methods* public double computeSalePrice() { *//abstract method* return this.getRegularPrice()-((20\*this.getRegularPrice())/100);  
 }  
}

**Test Class:**

package com.company;  
  
public class TestTask02 {  
  
 public static void main(String[] *args*) {  
  
 *// Declare and create Product array of size 5  
 Product*[] 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);  
 }  
}

**Output Screenshot:**

****