**Name: Hamad Nasir**

**Roll Number: 120312**

**Section: BSCS-6C**

**Lab No:8**

**Task 1:**

**Source Code:**

**Class Person:**

package task1;

class Person implements PersonInterface,AnotherPersonInterface {

//instant variables of person class

int CashSaving;

int RetirementFund;

String FirstName;

String LastName;

//person constructor with four arguments

public Person(int CashSaving,int RetirementFund ,String FirstName,String LastName){

this.CashSaving=CashSaving;

this.RetirementFund=RetirementFund;

this.FirstName=FirstName;

this.LastName=LastName;

}

//Compute wealth method with return type integer

public int ComputeTotalWealth(){

return 30000;

}

//get name method with return type String

public String getName(){

return String.format(FirstName+""+LastName);

}

//measure intelligence method with return type integer with string as argument

public int MeasureIntelligence(String a){

return 50;

}

}

**AnotherPersonInterface Interface:**

package task1;

//interface another person interface

public interface AnotherPersonInterface { //measure intelligence method with string as argumjent

int MeasureIntelligence(String b);

}

**PersonInterface Interface:**

package task1;

//person interface

public interface PersonInterface {

//compute wealth method with integer return type

int ComputeTotalWealth();

//get name method withstring return type

String getName();

}

**Test Class:**

package task1;

public class Task1 {

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. AnotherPersonInterface anotherpersoninterface1 = person1;

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

// anotherpersoninterface1 is the same object instance.

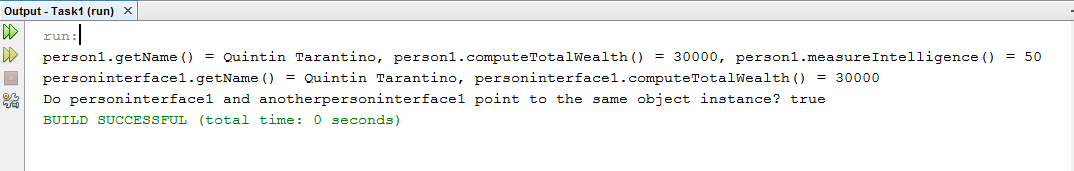
boolean b1 = (personinterface1 == anotherpersoninterface1);

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

}

}

**Output:**

****

**Task3:**

**Source Code:**

**RelationInterface Interface:**

package task2;

// interface RelationInterface

public interface RelationInterface {

boolean isGreater(Object Ln1,Object Ln2);

boolean isLess(Object Ln1,Object Ln2);

boolean isEqual(Object Ln1,Object Ln2);

}

**Line Class:**

package task2;

public class Line implements RelationInterface{

//instant variables

private double x1;

private double x2;

private double y1;

private double y2;

//line constructor

public Line(double x1,double x2, double y1, double y2){

this.x1=x1;

this.x2=x2;

this.y1=y1;

this.y2=y2;

}

//get length methode returning distance formulae

public double getLength(){

return Math.sqrt((x1-x2)\*(x1-x2) + (y1-y2)\*(y1-y2));

}

//boolean is greater method

public boolean isGreater(Object Ln1,Object Ln2){

Line Obj1 = (Line) Ln1;

Line Obj2 = (Line) Ln2;

if(Obj1.getLength()>Obj2.getLength()){

return true;

}

else{

return false;

}

}

//boolean is equal method

public boolean isEqual(Object Ln1,Object Ln2){

Line Obj1 = (Line) Ln1;

Line Obj2 = (Line) Ln2;

if(Obj1.getLength()==Obj2.getLength()){

return true;

}

else{

return false;

}

}

//boolean is less method

public boolean isLess(Object Ln1,Object Ln2){

Line Obj1 = (Line) Ln1;

Line Obj2 = (Line) Ln2;

if(Obj1.getLength()<Obj2.getLength()){

return true;

}

else{

return false;

}

}

}

**Test Class:**

package task2;

public class Task2 {

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);

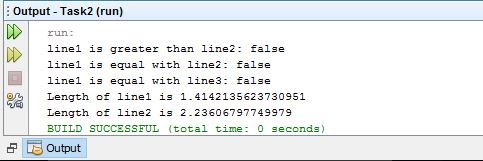
System.out.println("Length of line1 is " + line1.getLength());

System.out.println("Length of line2 is " + line2.getLength());

}

}

**Output:**



**Task3:**

**Source Code:**

**ProductInterface interface:**

package task3;

public interface ProductInterface {

double ComputeSalePrice();

double getRegularPrice();

void setRegularPrice(double x);

}

**ElectronicsInterface interface:**

package task3;

public interface ElectronicsInterface {

String getManufacturer();

}

**Class Product:**

package task3;

//Product is abstract main class

public class Product implements ProductInterface

{

//private instant variable

private double regularPrice;

//1 argument constructor

public Product(double regularPrice){

this.regularPrice=regularPrice;

}

//public compute sale price method

public double ComputeSalePrice(){

return regularPrice;

}

//set regular price method

public void setRegularPrice(double regularPrice){

this.regularPrice=regularPrice;

}

//get regular price method

public double getRegularPrice(){

return regularPrice;

}

}

**Class Electronics:**

package task3;

// Electronics is abstract subclass of product

public class Electronics extends Product implements ElectronicsInterface {

//private instant variable

private String manufacturer;

//2 argument constructor

public Electronics(double regularPrice,String manufacturer){

super(regularPrice);

}

//set manufacturer method

public void setManufacturer(String manufacturer){

this.manufacturer=manufacturer;

}

//get manufacturer method

public String getManufacturer(){

return this.manufacturer;

}

}

**Class Book:**

package task3;

// Class book extends Product

public class Book extends Product{

//private instant variables

private String publisher;

private int yearPublished;

//three aargument constructor

public Book(double regularPrice,String publisher,int yearPublished){

super(regularPrice);

}

//compute sale price method

public double ComputeSalePrice(){

return getRegularPrice()/2;

}

//set publisher method

public void setPublisfer(String publisher){

this.publisher=publisher;

}

//get publisher method

public String getPublisher(){

return this.publisher;

}

//set year published method

public void setYearPublished(int yeatPublished){

this.yearPublished=yearPublished;

}

//get year published method

public int getYearPublished(){

return this.yearPublished;

}

}

**Class MP3Player:**

package task3;

//MP3 is subclass of Electronics

public class MP3Player extends Electronics{

//private instant variable

private String color;

//3 argument constructor

public MP3Player(double regularPrice,String manufacturer,String color){

super(regularPrice,manufacturer);

}

//compute sale price method

public double ComputeSalePrice(){

return getRegularPrice()-getRegularPrice()\*0.1;

}

//set color method

public void SetColor(String color){

this.color=color;

}

//get color method

public String getColor(){

return this.color;

}

}

**Class TV:**

package task3;

//Electronics is subclass of Electronics

public class TV extends Electronics{

//private instant variable

private int size;

//3 argument constructor

public TV(double regulrPrice,String manufacturer,int size){

super(regulrPrice,manufacturer);

this.size = size;

}

//compute sale price method

public double ComputeSalePrice(){

return getRegularPrice()-getRegularPrice()\*0.2;

}

}

**Test Class:**

package task3;

public class Task3 {

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:**

