

**Object Oriented Programming**

**ASSIGNMENT NO 9**

**SUBMITTED BY:**

Hasaan Ahmad SP22-BSE-017

**SUBMITTED TO: Sir Muzaffar Iqbal**

**Activity 1:**

package LAB9;

/\*\*

 \* RegisterForExams

 \*/

public interface RegisterForExams {

    public void register();

}

package LAB9;

public class EmployeeTask implements RegisterForExams {

    private String name;

    private String date;

    private int salary;

    public EmployeeTask() {

        name = null;

        date = null;

        salary = 0;

    }

    public EmployeeTask(String name, String date, int salary) {

        this.name = name;

        this.date = date;

        this.salary = salary;

    }

    @Override

    public void register() {

        System.out.println("Employee is registered " + "Name " + name + "salary " + salary + " date " + date);

    }

}

package LAB9;

public class StudentTask implements RegisterForExams {

    private String name;

    private int age;

    private double gpa;

    public StudentTask() {

        name = null;

        age = 0;

        gpa = 0;

    }

    public StudentTask(String name, int age, double gpa) {

        this.name = name;

        this.age = age;

        this.gpa = gpa;

    }

    @Override

    public void register() {

        System.out.println("Student is registered " + "Student name " + name

                + " gpa " + gpa);

    }

}

**Runner Class**

package LAB9;

public class Runner {

    public static void main(String[] args) {

        EmployeeTask e = new EmployeeTask("Ahmed", "11,02,2001", 20000);

        StudentTask s = new StudentTask("Ali", 22, 3.5);

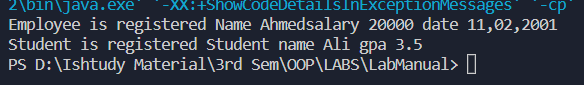
        e.register();

        s.register();

    }

}

**Output**

****

**Activity 2**

package LAB9;

interface I1 {

    void methodI1(); // public static by default

}

interface I2 extends I1 {

    void methodI2(); // public static by default

}

class A1 {

    public String methodA1() {

        String strA1 = "I am in methodC1 of class A1";

        return strA1;

    }

    public String toString() {

        return "toString() method of class A1";

    }

}

class B1 extends A1 implements I2 {

    public void methodI1() {

        System.out.println("I am in methodI1 of class B1");

    }

    public void methodI2() {

        System.out.println("I am in methodI2 of class B1");

    }

}

class C1 implements I2 {

    public void methodI1() {

        System.out.println("I am in methodI1 of class C1");

    }

    public void methodI2() {

        System.out.println("I am in methodI2 of class C1");

    }

}

// Note that the class is declared as abstract as it does not

// satisfy the interface contract

abstract class D1 implements I2 {

    public void methodI1() {

    }

    // This class does not implement methodI2() hence declared abstract.

}

public class InterFaceEx {

    public static void main(String[] args) {

        I1 i1 = new B1();

        i1.methodI1();

        I2 i2 = new B1();

        String var2 = ((A1) i1).methodA1();

        System.out.println("var2 : " + var2);

        String var3 = ((B1) i1).methodA1();

        System.out.println("var3 : " + var3);

        String var4 = i1.toString();

        System.out.println("var4 : " + var4);

        String var5 = i2.toString();

        System.out.println("var5 : " + var5);

        I1 i3 = new C1();

        String var6 = i3.toString();

        System.out.println("var6 : " + var6);

        Object o1 = new B1();

        ((I1) o1).methodI1(); // 1

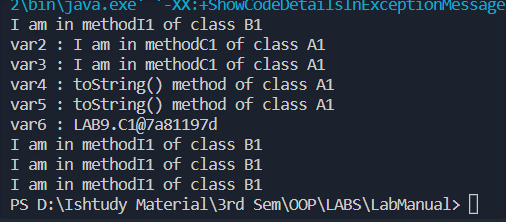
        ((I2) o1).methodI1(); // 2

        ((B1) o1).methodI1(); // 3

    }

}

**Output**



**GLT1**

package LAB9;

/\*\*

 \*

 \*/

interface Shape {

    double getArea();

}

/\*\*

 \* GLT1

 \*/

class Circle implements Shape {

    private double radius;

    public Circle() {

        radius = 0;

    }

    public Circle(double radius) {

        this.radius = radius;

    }

    @Override

    public double getArea() {

        return Math.PI \* radius \* radius;

    }

}

class Rectangle implements Shape {

    private double length;

    private double width;

    public Rectangle() {

        length = 0;

        width = 0;

    }

    public Rectangle(double length, double width) {

        this.length = length;

        this.width = width;

    }

    @Override

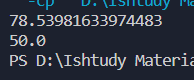
    public double getArea() {

        return length \* width;

    }

}

**Output:**

****

**GLT2:**

package LAB9;

interface Payable {

    Double getPaymentAmount();

}

class Invoice implements Payable {

    private String partNumber;

    private String partDescription;

    private int quantity;

    private double pricePerItem;

    public Invoice() {

        partNumber = null;

        partDescription = null;

        quantity = 0;

        pricePerItem = 0;

    }

    public Invoice(String partNumber, String partDescription, int quantity, double pricePerItem) {

        this.partNumber = partNumber;

        this.partDescription = partDescription;

        this.quantity = quantity;

        this.pricePerItem = pricePerItem;

    }

    @Override

    public Double getPaymentAmount() {

        return quantity \* pricePerItem;

    }

}

class Employee implements Payable {

    private String firstName;

    private String lastName;

    private String socialSecurityNumber;

    public Employee() {

        firstName = null;

        lastName = null;

        socialSecurityNumber = null;

    }

    public Employee(String firstName, String lastName, String socialSecurityNumber) {

        this.firstName = firstName;

        this.lastName = lastName;

        this.socialSecurityNumber = socialSecurityNumber;

    }

    @Override

    public Double getPaymentAmount() {

        return null;

    }

}

class SalariedEmployee extends Employee {

    private double weeklySalary;

    public SalariedEmployee() {

        weeklySalary = 0;

    }

    public SalariedEmployee(double weeklySalary) {

        this.weeklySalary = weeklySalary;

    }

    public SalariedEmployee(String firstName, String lastName, String socialSecurityNumber, double weeklySalary) {

        super(firstName, lastName, socialSecurityNumber);

        this.weeklySalary = weeklySalary;

    }

    @Override

    public Double getPaymentAmount() {

        return weeklySalary;

    }

}

public class GLT2 {

    public static void main(String[] args) {

        Payable[] payableObjects = new Payable[4];

        payableObjects[0] = new Invoice("01234", "seat", 2, 375.00);

        payableObjects[1] = new Invoice("56789", "tire", 4, 79.95);

        payableObjects[2] = new SalariedEmployee("Hasaan", "Ahmad", "111-11-1111", 800.00);

        payableObjects[3] = new SalariedEmployee("Mujtaba", "", "888-88-8888", 1200.00);

        System.out.println("Invoices and Employees processed polymorphically:");

        for (Payable currentPayable : payableObjects) {

            System.out.printf("%n%s %n%s: $%,.2f%n", currentPayable.toString(), "payment due",

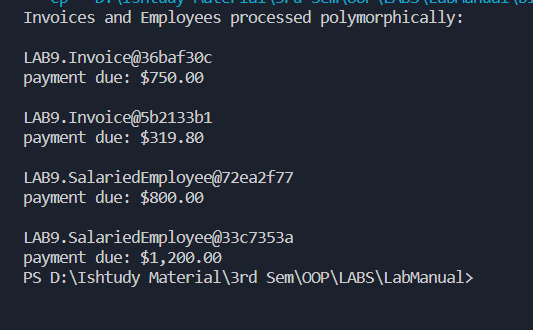
                    currentPayable.getPaymentAmount());

        }

    }

}

**Output:**

****

**GLT3:**

package LAB9;

interface Compare {

    boolean compareObjects(Object o);

}

class Inventory implements Compare {

    private String name;

    private int quantity;

    private double price;

    public Inventory() {

        name = null;

        quantity = 0;

        price = 0;

    }

    public Inventory(String name, int quantity, double price) {

        this.name = name;

        this.quantity = quantity;

        this.price = price;

    }

    @Override

    public boolean compareObjects(Object o) {

        if (o instanceof Inventory) {

            Inventory i = (Inventory) o;

            if (name.equals(i.name) && quantity == i.quantity && price == i.price) {

                return true;

            }

        }

        return false;

    }

}

class GLT3 {

    public static void main(String[] args) {

        Inventory i1 = new Inventory("Apple", 10, 1.5);

        Inventory i2 = new Inventory("Apple", 10, 1.5);

        Inventory i3 = new Inventory("Orange", 10, 1.5);

        System.out.println(i1.compareObjects(i2));

        System.out.println(i1.compareObjects(i3));

    }

}

**Output:**



**GLT4:**

package LAB9;

/\*\*

 \* GLT4

 \* Create constructor and abstract methods of interface in the class

 \* NameCollection.

 \* Then write a main method that creates a NamesCollection object with a sample

 \* array of strings,

 \* and then iterates through the enumeration outputting each name using the

 \* getNext() method.

 \*/

interface Enumeration {

    public boolean hasNext(int index);

    public Object getNext(int index);

}

class NameCollection implements Enumeration {

    private String[] names; // Array of names

    private int index;

    public NameCollection() {

        names = null;

        index = 0;

    }

    public NameCollection(String[] names) {

        this.names = names;

        index = 0;

    }

    @Override

    public boolean hasNext(int index) {

        if (index < names.length) {

            return true;

        }

        return false;

    }

    @Override

    public Object getNext(int index) {

        if (hasNext(index)) {

            return names[index++];

        }

        return null;

    }

    // Method to print names

    public void printNames() {

        for (int i = 0; i < names.length; i++) {

            System.out.println(names[i]);

        }

    }

    @Override

    public String toString() {

        return "NameCollection [index=" + index + ", names=" + names + "]";

    }

}

public class GLT4 {

    public static void main(String[] args) {

        // Fill array with data

        String[] names = { "Hasaan", "Mujtaba", "Haider", "Ali", "Salman" };

        NameCollection nameCollection = new NameCollection(names);

        System.out.println(nameCollection.toString());

        // Print names

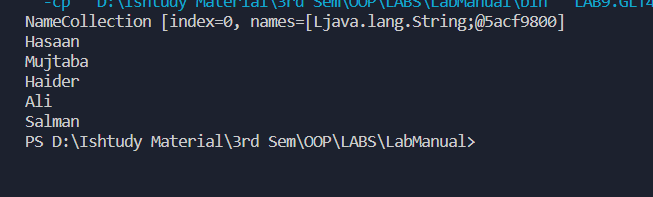
        nameCollection.printNames();

        nameCollection.toString();

    }

}

**Output:**

****