BYAGARI PAVAN PBL ID: J\_251890123

**Hands-on Assignments for Overriding / Polymorphism**

**Create a base class Fruit with name, taste and size as its attributes.**

**Create a method called eat() which describes the name of the fruit and its taste.**

**Inherit the same in 2 other classes Apple and Orange and override the eat() method to represent each fruit taste.**

import java.util.Scanner;

// Base class class Fruit {

String name; String taste; String size;

Fruit(String name, String taste, String size) { this.name = name;

this.taste = taste; this.size = size;

}

void eat() {

System.out.println("The fruit is " + name + " and it tastes " + taste + ".");

}

}

// Derived class - Apple

class Apple extends Fruit { Apple(String size) {

super("Apple", "sweet and sometimes a little sour", size);

}

@Override

void eat() {

System.out.println(name + " is juicy, " + taste + ", and usually " + size + " in size.");

}

}

// Derived class - Orange

class Orange extends Fruit { Orange(String size) {

super("Orange", "citrusy and tangy", size);

}

@Override void eat() {

System.out.println(name + " is " + taste + ", refreshing, and typically " + size + " in size.");

}

}

// Main class

public class Main {

public static void main(String[] args) { Scanner sc = new Scanner(System.in);

// Input for generic fruit System.out.print("Enter fruit name: "); String name = sc.nextLine(); System.out.print("Enter taste: ");

String taste = sc.nextLine(); System.out.print("Enter size: "); String size = sc.nextLine();

Fruit fruit = new Fruit(name, taste, size);

fruit.eat();

// Input for Apple

System.out.print("Enter size of Apple: "); String appleSize = sc.nextLine();

Fruit apple = new Apple(appleSize); apple.eat();

// Input for Orange

System.out.print("Enter size of Orange: "); String orangeSize = sc.nextLine();

Fruit orange = new Orange(orangeSize); orange.eat();

sc.close();

}

}

**Output:**

The fruit is Mango and it tastes sweet.

Apple is juicy, sweet and sometimes a little sour, and usually medium in size. Orange is citrusy and tangy, refreshing, and typically small in size.

**Write a program to create a class named shape. It should contain 2 methods, draw() and erase() tha prints "Drawing Shape" and "Erasing Shape" respectively.**

**For this class, create three sub classes, Circle, Triangle and Square and each class should overric the parent class functions draw() and erase ().**

**The draw() method should print "Drawing Circle", "Drawing Triangle" and "Drawing Square" respectively.**

**The erase() method should print "Erasing Circle", "Erasing Triangle" and "Erasing Square" respectively.**

**Create objects of Circle, Triangle and Square in the following way and observe the**

**polymorphic nature of the class by calling draw() and erase() method using each object. Shape c=new Circle();**

**Shape t=new Triangle(); Shape s = 1 new Square();** import java.util.Scanner;

// Base class class Shape {

void draw() {

System.out.println("Drawing Shape");

}

void erase() {

System.out.println("Erasing Shape");

}

}

// Derived class - Circle

class Circle extends Shape { @Override

void draw() {

System.out.println("Drawing Circle");

}

@Override void erase() {

System.out.println("Erasing Circle");

}

}

// Derived class - Triangle

class Triangle extends Shape { @Override

void draw() {

System.out.println("Drawing Triangle");

}

@Override void erase() {

System.out.println("Erasing Triangle");

}

}

// Derived class - Square

class Square extends Shape { @Override

void draw() {

System.out.println("Drawing Square");

}

@Override void erase() {

System.out.println("Erasing Square");

}

}

// Main class to test polymorphism public class Main {

public static void main(String[] args) { Scanner sc = new Scanner(System.in);

System.out.println("Choose a shape: 1.Circle 2.Triangle 3.Square"); int choice = sc.nextInt();

Shape shape;

switch (choice) { case 1:

shape = new Circle(); break;

case 2:

shape = new Triangle(); break;

case 3:

shape = new Square(); break;

default:

shape = new Shape(); break;

}

// Polymorphism in action shape.draw();

shape.erase();

sc.close();

}

}

**Output:** Drawing Circle Erasing Circle