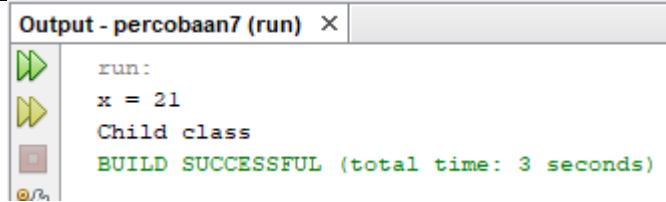






JOBSHEET P7 POLYMORPHISM

Nama : Dini Ayuastina

NIM : F1B021114

Kelompok : 5

No .	Kegiatan dan Latihan	Program	Hasil <i>Running</i>
1	Pengenalan Virtual method Invocation	<pre> package percobaan_7; class parent{ int x = 21; public void info(){ System.out.println("Parent class"); } } class child extends parent{ int x = 11; public void info(){ System.out.println("Child class"); } } public class js1 { public static void main(String[] args) { parent obj = new child(); System.out.println("x = "+obj.x); obj.info(); } } </pre>	

2	Pengenalan Heterogeneous Collection	<pre> package percobaan_7; import java.util.ArrayList; import java.util.List; abstract class shape{ protected double area; public abstract void calculateArea(); public void displayArea(){ System.out.println("Area of the shape is: "+area); } } class circle extends shape{ private double radius; public circle(double newRadius) { this.radius = newRadius; } @Override public void calculateArea(){ area = Math.PI * Math.pow(radius,2); } } // public class rectangle extends shape { // private double width; // private double height; // public rectangle(double width, double height){ // this.width = width; // this.height = height; // } // @Override // public void calculateArea(){ // area = width * height; </pre>	<div> <div>Output - percobaan7 (run) ×</div> <div>  run:  Circle :  Area of the shape is: 50.26548245743669  Traingle: Area of the shape is: 6.0 BUILD SUCCESSFUL (total time: 0 seconds) </div> </div>
---	--	---	--

```
//      }
//  }

class triangle extends shape{
    private double a;
    private double b;
    private double c;

    public triangle(double a, double b, double
c){
        this.a = a;
        this.b = b;
        this.c = c;
    }

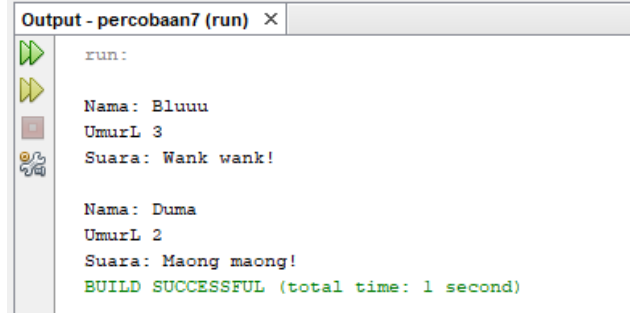
    @Override
    public void calculateArea(){
        double s = (a+b+c)/2;

        area = Math.sqrt(s*(s-a)*(s-b)*(s-c));
    }
}





public class js2 {

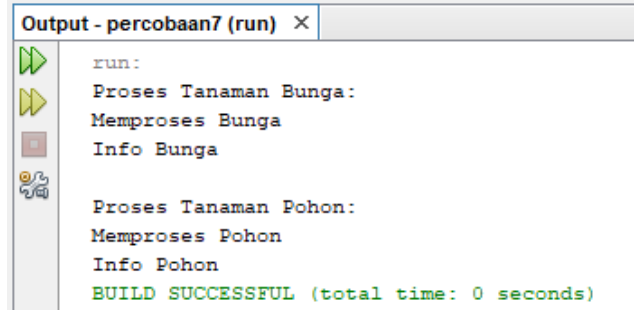
    public static void main(String[] args) {
        List<shape> shapes = new ArrayList<>();
        shapes.add(new circle(4.0));
        // shapes.add(new rectangle(8, 18.0));
        shapes.add(new triangle(3.0, 4.0, 5.0));

        for(shape shape: shapes){
            if(shape instanceof circle){
                System.out.println("Circle : ");
                circle circle = (circle) shape;
                circle.calculateArea();
                circle.displayArea();
            }
            // } else if (shape instanceof
rectangle){
```





		<pre> // System.out.println("Rectangle : "); // rectangle rectangle = (rectangle) shape; // rectangle.calculateArea(); // rectangle.displayArea(); } else if (shape instanceof triangle){ System.out.println("Traingle: "); triangle triangle = (triangle) shape; triangle.calculateArea(); triangle.displayArea(); } } } </pre>	
3	Pengenalan Polymorphic Argument	<pre> package percobaan_7; public class js3 { public static void main(String[] args) { Hewan[] hewan = new Hewan[2]; hewan[0] = new anjing("Bluuu", 3); hewan[1] = new kucing("Duma", 2); for(Hewan binatang : hewan){ System.out.println("\nNama: "+binatang.getName()); System.out.println("UmurL "+binatang.getAge()); System.out.println("Suara: "+binatang.makeSound()); } } } abstract class Hewan{ private final String name; private int age; </pre>	 <pre> run: Nama: Bluuu UmurL 3 Suara: Wank wank! Nama: Duma UmurL 2 Suara: Maong maong! BUILD SUCCESSFUL (total time: 1 second) </pre>

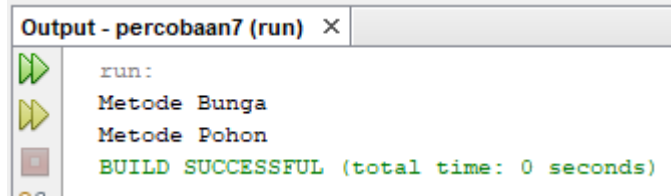
		<pre>Hewan(String nwName, int nwAge){ this.name = nwName; this.age = nwAge; } String getName(){ return name; } int getAge(){ return age; } abstract String makeSound(); } class anjing extends Hewan{ anjing(String name, int age){ super(name, age); } @Override String makeSound(){ return "Wank wank!"; } } class kucing extends Hewan{ kucing(String name, int age){ super(name, age); } @Override String makeSound(){ return "Maong maong!"; } }</pre>	
--	--	---	--

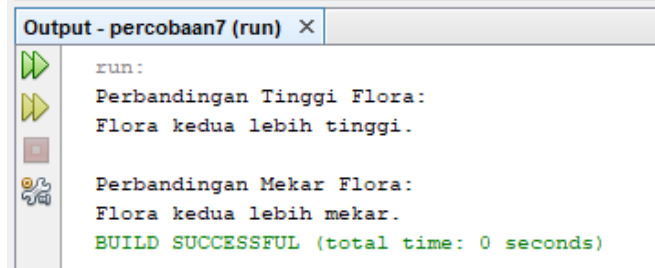
4	Pengenalan Operator instanceof	<pre> package percobaan_7; public class js4 { public static void main(String[] args) { Hewan[] hewan = new Hewan[2]; hewan[0] = new anjing("Bluuu", 3); hewan[1] = new kucing("Duma", 2); for(Hewan binatang : hewan){ if (binatang instanceof anjing){ System.out.println(binatang.getName()+" adalah anjing"); } else if (binatang instanceof kucing){ System.out.println(binatang.getName()+" adalah kucing"); } } } } abstract class Hewann{ private final String name; private int age; Hewann(String nwName, int nwAge){ this.name = nwName; this.age = nwAge; } String getName(){ return name; } int getAge(){ return age; } abstract String makeSound(); </pre>	<div> Output - percobaan7 (run) × </div> <div>     </div> <pre> run: Bluuu bersuara: Wank wank! Duma bersuara: Maong maong! BUILD SUCCESSFUL (total time: 1 second) </pre>
---	---	---	--

		<pre> } class anjing extends Hewan{ anjing(String name, int age){ super(name, age); } @Override String makeSound(){ return "Wank wank!"; } } class kucing extends Hewan{ kucing(String name, int age){ super(name, age); } @Override String makeSound(){ return "Maong maong!"; } } </pre>	
5	Pengenalan Object Casting	<pre> package percobaan7; class Tanaman { void tampilkanInfo() { System.out.println("Info Tanaman Umum"); } } class Bunga extends Tanaman { void tampilkanInfo() { System.out.println("Info Bunga"); } } class Pohon extends Tanaman { </pre>	<div>Output - percobaan7 (run) ×</div>  <pre> run: Proses Tanaman Bunga: Memproses Bunga Info Bunga Proses Tanaman Pohon: Memproses Pohon Info Pohon BUILD SUCCESSFUL (total time: 0 seconds) </pre>

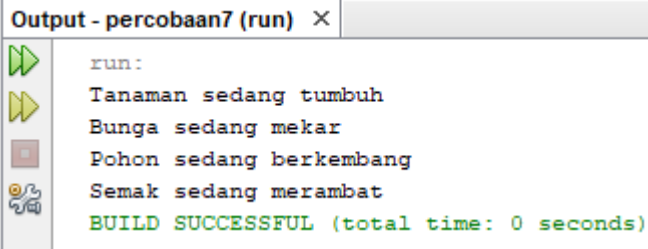
		<pre>void tampilkanInfo() { System.out.println("Info Pohon"); } public class js5 { public static void prosesTanaman(Object tanaman) { if (tanaman instanceof Bunga) { Bunga bunga = (Bunga) tanaman; System.out.println("Memproses Bunga"); bunga.tampilkanInfo(); } else if (tanaman instanceof Pohon) { Pohon pohon = (Pohon) tanaman; System.out.println("Memproses Pohon"); pohon.tampilkanInfo(); } else { System.out.println("Memproses Tanaman Lainnya"); } } public static void main(String[] args) { Tanaman bunga = new Bunga(); Tanaman pohon = new Pohon(); System.out.println("Proses Tanaman Bunga:"); prosesTanaman(bunga); System.out.println("\nProses Tanaman Pohon:"); prosesTanaman(pohon); } }</pre>	
--	--	--	--

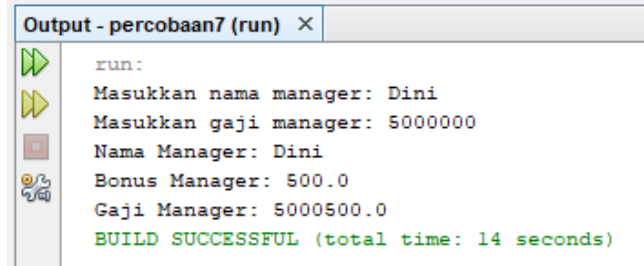
6	Pengenalan Object Casting : Up Casting	<pre> package percobaan7; class Tanamann { protected String jenis; public Tanamann(String jenis) { this.jenis = jenis; } @Override public String toString() { return "Info Tanaman: " + jenis; } } class Bungaa extends Tanamann { public Bungaa(String jenis) { super(jenis); } public String methodeBunga() { return "Metode Bunga"; } } class Pohonn extends Tanamann { public Pohonn(String jenis) { super(jenis); } public String methodePohon() { return "Metode Pohon"; } } public class js6 { public static void main(String[] args) { Bungaa mawar = new Bungaa("Mawar"); Pohonn oak = new Pohonn("Oak"); } } </pre>	<div> Output - percobaan7 (run) × </div> <div>  run:  Info Tanaman: Mawar  Info Tanaman: Oak  BUILD SUCCESSFUL (total time: 0 seconds) </div>
---	---	--	--

		<pre> Tanamann tanaman1 = (Tanamann) mawar; Tanamann tanaman2 = (Tanamann) oak; System.out.println(tanaman1.toString()); System.out.println(tanaman2.toString()); } } </pre>	
7	Pengenalan Object Casting : Down Casting	<pre> package percobaan7; class Tanamannn { protected String jenis; public Tanamannn(String jenis) { this.jenis = jenis; } public String toString() { return "Info Tanaman: " + jenis; } } class Bungaaa extends Tanamannn { public Bungaaa(String jenis) { super(jenis); } public String metodeBunga() { return "Metode Bunga"; } } class Pohonnn extends Tanamannn { public Pohonnn(String jenis) { super(jenis); } } </pre>	

		<pre> } public String methodPohon() { return "Metode Pohon"; } } public class js7 { public static void main(String[] args) { Tanamannn tanaman = new Bungaaa("Mawar"); if (tanaman instanceof Bungaaa) { Bungaaa bunga = (Bungaaa) tanaman; System.out.println(bunga.methodBunga()); } // Membuat objek Tanaman Tanamannn tanaman2 = new Pohonn("Oak"); if (tanaman2 instanceof Pohonn) { Pohonn pohon = (Pohonn) tanaman2; System.out.println(pohon.methodPohon()); } } } </pre>	
8	menggunakan Overloading/statis polimorfism	<pre> package percobaan7; class PerbandinganFlora { public static String bandingkan(int nilai1, int nilai2) { if (nilai1 > nilai2) { return "Flora pertama lebih tinggi."; } else if (nilai1 < nilai2) { return "Flora kedua lebih tinggi."; } else { return "Kedua flora memiliki tinggi yang sama."; } } } </pre>	 <pre> run: Perbandingan Tinggi Flora: Flora kedua lebih tinggi. Perbandingan Mekar Flora: Flora kedua lebih mekar. BUILD SUCCESSFUL (total time: 0 seconds) </pre>

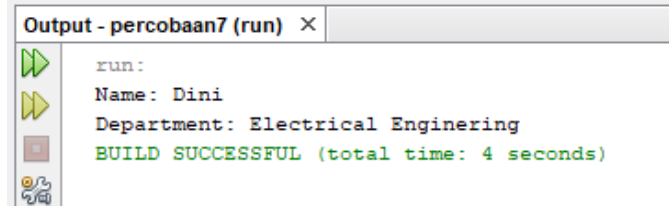
		<pre> } public static String bandingkan(double nilai1, double nilai2) { if (nilai1 > nilai2) { return "Flora pertama lebih mekar."; } else if (nilai1 < nilai2) { return "Flora kedua lebih mekar."; } else { return "Kedua flora mekar dengan seimbang."; } } } public class js8 { public static void main(String[] args) { System.out.println("Perbandingan Tinggi Flora:"); System.out.println(PerbandinganFlora.bandingkan(1 0, 15)); System.out.println("\nPerbandingan Mekar Flora:"); System.out.println(PerbandinganFlora.bandingkan(5 .0, 5.5)); } }</pre>	
--	--	---	--

9	Menggunakan Overriding/dinamis polimorfism	<pre> package percobaan7; class Tanaman9 { public void tumbuh() { System.out.println("Tanaman sedang tumbuh"); } } class Bunga9 extends Tanaman9 { public void tumbuh() { System.out.println("Bunga sedang mekar"); } } class Pohon9 extends Tanaman9 { public void tumbuh() { System.out.println("Pohon sedang berkembang"); } } class Semak extends Tanaman9 { public void tumbuh() { System.out.println("Semak sedang merambat"); } } public class js9 { public static void main(String[] args) { Tanaman9 tanaman91 = new Tanaman9(); tanaman91.tumbuh(); } } </pre>	
---	--	---	---

		<pre> tanaman91 = new Bunga9(); tanaman91.tumbuh(); tanaman91 = new Pohon9(); tanaman91.tumbuh(); tanaman91 = new Semak(); tanaman91.tumbuh(); } </pre>	
10	Menggunakan metode Setter and Getter (Encapsulation)	<pre> package percobaan7; import java.util.Scanner; public class js10 { private String name; private double salary; private static double salary_rise_percent = 0.2; public js10(String nm, double sly) { this.setName(nm); this.setSalary(sly); } public void setName(String nm) { name = nm; } public void setSalary(double sly) { salary = sly; } public static void setPresentase(double percent) { salary_rise_percent = percent; } public String getName() { </pre>	 <pre> Output - percobaan7 (run) X run: Masukkan nama manager: Dini Masukkan gaji manager: 5000000 Nama Manager: Dini Bonus Manager: 500.0 Gaji Manager: 5000500.0 BUILD SUCCESSFUL (total time: 14 seconds) </pre>

		<pre> return name; } public double getSalary() { return salary; } public static double getPresentase() { return salary_rise_percent; } public void salaryUp() { salary += (salary * salary_rise_percent); } } class Manager extends js10 { private static double bonus = 500; public Manager(String nm, double sly) { super(nm, sly); } public double getBonus() { return bonus; } public void setBonus(double bns) { bonus = bns; } @Override public double getSalary() { double salaryBase = super.getSalary(); return (salaryBase + bonus); } } class TestManager {</pre>	
--	--	--	--

		<pre> public static void main(String[] args) { Scanner scanner = new Scanner(System.in); System.out.print("Masukkan nama manager: "); String name = scanner.nextLine(); System.out.print("Masukkan gaji manager: "); double salary = scanner.nextDouble(); Manager mng = new Manager(name, salary); System.out.println("Nama Manager: " + mng.getName()); System.out.println("Bonus Manager: " + mng.getBonus()); System.out.println("Gaji Manager: " + mng.getSalary()); } } </pre>	
11	Menggunakan kata this dan super	<pre> package percobaan7; class Person { String name = "Medi"; int age = 21; } class Lecture extends Person { float salary = 4000f; String name = "Dini Ayu"; int age = 21; public void showInfo() { System.out.println("Name :" + this.name); System.out.println("Age :" + this.age); System.out.println("Salary : \$" + salary); } } </pre>	<pre> Output - percobaan7 (run) × run: Name :Dini Ayu Age :21 Salary : \$4000.0 BUILD SUCCESSFUL (total time: 0 seconds) </pre>

		<pre> } public class js11 { public static void main(String[] args) { Lecture rismon = new Lecture(); rismon.showInfo(); } } </pre>	
12	Memanggil parent class constructor metode overloading	<pre> package percobaan7; import java.util.Date; class Employee { private static final double BASE_SALARY = 15000.00; private String name; private double salary; private Date birthDate; public Employee(String name, double salary, Date DoB) { this.name = name; this.salary = salary; this.birthDate = DoB; } public Employee(String name, double salary) { this(name, salary, null); } public Employee(String name, Date DoB) { this(name, BASE_SALARY, DoB); } public Employee(String name) { this(name, BASE_SALARY); } } </pre>	

		<pre> public String getName() { return name; } } class Managerr extends Employee { private String department; public Managerr(String name, double salary, String dept) { super(name, salary); this.department = dept; } public Managerr(String name, String dept) { super(name); this.department = dept; } public String getDepartment() { return department; } } public class js12 { public static void main(String[] args) { Employee man = new Managerr("Dini", 16000.00, "Electrical Enginering"); if (man instanceof Managerr) { Managerr manager = (Managerr) man; System.out.println("Name: " + manager.getName()); System.out.println("Department: " + manager.getDepartment()); } } }</pre>	
--	--	---	--