JOBSHEET P7

POLYMORPHISM

Nama : I Komang Krisna Dwipayana Wadhiesta

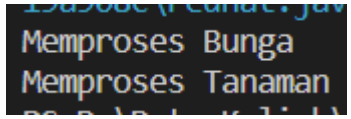
NIM : F1B021050

Kelompok : 2

No.	Kegiatan dan Latihan	Program	Hasil Running

1.	Pengenalan *Virtual method Invocation* (Buatlah program bebas menggunakan virtual method invocation)	`//Bunga.java`
		`package JSNo1;`
		
		`public class Bunga {`
		`  int tinggi = 12;`
		
		`  public void tampilInfo() {`
		`  System.out.println("Class Bunga");`
		`  }`
		`}`
		
		`//Anggrek.java`
		`package JSNo1;`
		
		`public class Anggrek extends Bunga {`
		`  int tinggi = 10;`
		
		`  @Override`
		`  public void tampilInfo() {`
		`  System.out.println("Class Anggrek");`
		`  }`
			`Tinggi = 12 Cm`
			`Class Anggrek`

		`}`
		`//main.java`
		`package JSNo1;`
		
		`public class Main {`
		`  public static void main(String[] args) {`
		`  Bunga bu = new Anggrek();`
		`  System.out.println("Tingginya = "+ bu.tinggi +" Cm");`
		`  bu.tampilInfo();`
		`  }`
		`}`	

2.	Pengenalan Heterogeneous Collection (Buatlah program bebas atau modifikasi program disamping menggunakan Heterogeneous Collection)	`import java.util.ArrayList;`
		
		`class Flora {`
		
		`}`
		
		`class Bunga extends Flora {`
		
		`}`
		
		`class Tanaman extends Flora {`
		
		`}`
		
		`public class JSNo2 {`
		`  public static void prosesFlora(Flora Flora) {`
		`  if (Flora instanceof Bunga) {`
		
		`  System.out.println("Memproses Bunga");`
		`  } else if (Flora instanceof Tanaman) {`
		
		`  System.out.println("Memproses Tanaman");`
			

		`} else {`
		
		`System.out.println("Memproses Flora Lainnya");`
		`}`
		`}`
		
		`public static void main(String[] args) {`
		`ArrayList<Flora> taman = new ArrayList<>();`
		
		`taman.add(new Bunga());`
		`taman.add(new Tanaman());`
		
		`for (Flora Flora : taman) {`
		`prosesFlora(Flora);`
		`}`
		`}`
		`}`	

3.	Pengenalan Polymorphic Argument (uatlah program bebas atau modifikasi program disamping menggunakan polymorphic argument)	`class Mobil {`
		`String merek;`
		
		`Mobil(String merek) {`
		`this.merek = merek;`
		`}`
		
		`void berkendara() {`
		`System.out.println("Mobil " + merek + " sedang berkendara");`
		`}`
		`}`
		
		`class Sedan extends Mobil {`
		`Sedan(String merek) {`
			Lun.
			Sedan Toyota sedang berkendara
			SUV Honda sedang berkendara

```
        super(merek);
    }

    @Override
    void berkendara() {
        System.out.println("Sedan " + merek + "
sedang berkendara");
    }
}

class SUV extends Mobil {
    SUV(String merek) {
        super(merek);
    }

    @Override
    void berkendara() {
        System.out.println("SUV " + merek + " sedang
berkendara");
    }
}

public class Kendaraan {
    public static void Proses(Mobil mobil) {
        if (mobil instanceof Sedan) {
            Sedan sedan = (Sedan) mobil;
            sedan.berkendara();
        } else if (mobil instanceof SUV) {
            SUV suv = (SUV) mobil;
            suv.berkendara();
        } else {
            System.out.println("Memproses Mobil
Lainnya");
        }
```

		`}`
		
		`public static void main(String[] args) {`
		`Mobil sedan = new Sedan("Toyota");`
		`Mobil suv = new SUV("Honda");`
		
		`Proses(sedan);`
		`Proses(suv);`
		`}`
		`}`	

4.	Pengenalan Operator Instanceof (Buatlah program bebas atau modifikasi program disamping menggunakan operator instanceof)	`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 {`
		`void tampilkanInfo() {`
		`System.out.println("Info Pohon");`
		`}`
		`}`
		
		`public class jsp704 {`
		`public static void prosesTanaman(Tanaman tanaman)`
		`{`
		`if (tanaman instanceof Bunga) {`
			`Proses Flora Bunga:`
			`Memproses Bunga`
			
			`Proses Flora Pohon:`
			`Memproses Pohon`

		``
		System.out.println("Memproses Bunga");
		} else if (tanaman instanceof Pohon) {
		
		System.out.println("Memproses Pohon");
		} 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);
		}
		}
		``	

5.	Pengenalan Object Casting (Buatlah program bebas atau modifikasi program disamping menggunakan Object Casting)	`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 {`
		`  void tampilkanInfo() {`
		`  System.out.println("Info Pohon");`
		`  }`
		`}`
		
		`public class jsp705 {`
		`  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 {`
			`Proses Flora Bunga:`
			`Memproses Bunga`
			`Info Bunga`
			
			`Proses Flora Pohon:`
			`Memproses Pohon`
			`Info Pohon`

<table>
<tr>
<td></td>
<td></td>
<td>

```
            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);
    }
}
```

</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>Pengenalan Object Casting : Up Casting (Buatlah program bebas dengan Up Casting)</td>
<td>

```
package jsno6;

class Flora {
    protected String jenis;

    public Flora(String jenis) {
        this.jenis = jenis;
    }

    @Override
    public String toString() {
        return "Info Flora: " + jenis;
    }
}

class Bunga extends Flora {
    public Bunga(String jenis) {
```

</td>
<td>

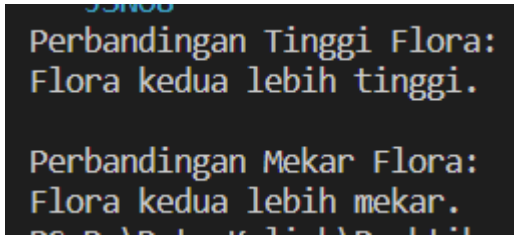
```
Info Flora: Mawar
Info Flora: Jati
```

</td>
</tr>
</table>

```
        super(jenis);
    }

    public String methodeBunga() {
        return "Metode Bunga";
    }
}

class Pohon extends Flora {
    public Pohon(String jenis) {
        super(jenis);
    }

    public String methodePohon() {
        return "Metode Pohon";
    }
}

public class JSNo6 {
    public static void main(String[] args) {
        Bunga mawar = new Bunga("Mawar");
        Pohon jati = new Pohon("Jati");

        Flora Flora1 = (Flora) mawar;


        Flora Flora2 = (Flora) jati;


        System.out.println(Flora1.toString());
        System.out.println(Flora2.toString());
    }
}
```

7.	Pengenalan Object Casting : Down Casting (Buatlah program bebas dengan Down Casting)	<pre>package jsno7;
		
		class Flora {
		  protected String jenis;
		
		  public Flora(String jenis) {
		  this.jenis = jenis;
		  }
		
		  public String toString() {
		  return "Info Flora: " + jenis;
		  }
		}
		
		class Bunga extends Flora {
		  public Bunga(String jenis) {
		  super(jenis);
		  }
		
		  public String methodeBunga() {
		  return "Metode Bunga";
		  }
		}
		
		class Pohon extends Flora {
		  public Pohon(String jenis) {
		  super(jenis);
		  }
		
		  public String methodePohon() {
		  return "Metode Pohon";
		  }
		}</pre>
			Run:
			Metode Bunga
			Metode Pohon

		`public class JSNo7 {`
		`  public static void main(String[] args) {`
		`  Flora flora = new Bunga("Mawar");`
		`  if (flora instanceof Bunga) {`
		`  Bunga bunga = (Bunga) flora;`
		`  System.out.println(bunga.methodeBunga());`
		`  }`
		
		`  Flora flora2 = new Pohon("Jati");`
		
		`  if (flora2 instanceof Pohon) {`
		`  Pohon pohon = (Pohon) flora2;`
		`  System.out.println(pohon.methodePohon());`
		`  }`
		`  }`
		`}`	

8.	menggunakan Overloading/statis Polimorfism (Buatlah program untuk membandingkan kedua nilai menggunakan polimorfis statis)	`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.";`
		`  }`
		`  }`
		
		`  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 JSNo8 {
		    public static void main(String[] args) {
		        System.out.println("Perbandingan Tinggi
		Flora:");
		
		System.out.println(PerbandinganFlora.bandingkan(20,
		25));
		
		        System.out.println("\nPerbandingan Mekar
		Flora:");
		
		System.out.println(PerbandinganFlora.bandingkan(7.0,
		8.5));
		    }
		}
		``	

9.	Menggunakan Overriding/dinamis Polimorfism (Buatlah program bebas menggunakan polimorfis dinamis dengan jumlah class : Akhiran NIM ganjil : 3 class Akhiran NIM genap : 4 class.)	
		package package9;
		
		class Tanaman9 {
		public void tumbuh() {
		System.out.println("Tanaman sedang tumbuh");
		}
		}
		
		class Bunga extends Tanaman9 {
		
		public void tumbuh() {
		System.out.println("Bunga sedang mekar");
		}
		}
		
		class Pohon extends Tanaman9 {
		
		public void tumbuh() {
		System.out.println("Pohon sedang berkembang");
		}
		}
		
		class Semak extends Tanaman9 {
		
		public void tumbuh() {
		System.out.println("Semak sedang merambat");
		}
		}
			Tanaman sedang tumbuh
			Bunga sedang mekar
			Pohon sedang berkembang
			Semak sedang merambat

		`public class jsp709 {`
		`  public static void main(String[] args) {`
		`  Tanaman9 tanaman91 = new Tanaman9();`
		`  tanaman91.tumbuh();`
		
		`  tanaman91 = new Bunga();`
		`  tanaman91.tumbuh();`
		
		`  tanaman91 = new Pohon();`
		`  tanaman91.tumbuh();`
		
		`  tanaman91 = new Semak();`
		`  tanaman91.tumbuh();`
		`  }`
		`}`	

10.	Menggunakan methode Setter and Getter (Encapsulation) (Modifikasi program disamping menggunakan inputan dinamis.)	`import java.util.Scanner;`
		
		`public class JSNo10 {`
		`  private String name;`
		`  private double salary;`
		`  private static double salary_rise_percent = 0.2;`
		
		`  public JSNo10(String nm, double sly) {`
		`  this.setName(nm);`
		`  this.setSalary(sly);`
		`  }`
		
		`  public void setName(String nm) {`
		`  name = nm;`
		`  }`
		
		`  public void setSalary(double sly) {`
		`  salary = sly;`
		`  }`
			`Masukkan nama manager: Kris`
			`Masukkan gaji manager: 500`
			`==========`
			`Nama Manager: Kris`
			`Bonus Manager: 500.0`
			`Gaji Manager: 1000.0`
			`==========`

```
    public static void setPresentase(double percent)
{
        salary_rise_percent = percent;
    }

    public String getName() {
        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 JSNo10 {
    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 {
    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("================================
");
        System.out.println("Nama Manager: " +
mng.getName());
        System.out.println("Bonus Manager: " +
mng.getBonus());
        System.out.println("Gaji Manager: " +
mng.getSalary());
```

		System.out.println("========== "); } }	

11.	Menggunakan kata this dan super
	(Buatlah program tambahan mengikuti contoh disamping
	lalu mengganti keyword super menjadi this.)
		class Person {
		String name = "Kirara";
		int age = 21;
		}
		
		class Lecture extends Person {
		float salary = 4000f;
		String name = "Rizzy";
		int age = 49;
		
		public void showInfo() {
		System.out.println("Name\t : " + this.name);
		System.out.println("Age\t : " + this.age);
		System.out.println("Salary\t : $" + salary);
		}
		}
		
		public class JSNo11 {
		public static void main(String[] args) {
		Lecture rizz = new Lecture();
		rizz.showInfo();
		}
		}
			

12.	Memanggil parent class constructor methode overloading (Modifikasi program mengikuti contoh disamping (bebas))	<pre>package package9;
		
		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);
		  }
		
		  public String getName() {
		  return name;
		  }
		}</pre>
			<pre>Name : Kirara
			Department: Doctor</pre>

```
class Manager extends Employee {
    private String department;

    public Manager(String name, double salary, String
dept) {
        super(name, salary);
        department = dept;
    }

    public Manager(String name, String dept) {
        super(name);
        department = dept;
    }

    public String getDepartment() {
        return department;
    }
}

public class jsp709 {
    public static void main(String[] args) {
        Employee man = new Manager("Dori", 16000.00,
"Electrical");

        if (man instanceof Manager) {
            Manager manager = (Manager) man;
            System.out.println("Name: " +
man.getName());
            System.out.println("Department: " +
manager.getDepartment());
        }
    }
}
```
