Nama : Ario Febri Putra Raharjo

NIM : 201011400814

Pemrograman Berorientasi Objek P5

# Percobaan 1 :

public class Parent {

public int x = 5;

}

class Child extends Parent {

public int x = 10;

public void info(){

System.out.println("Nilai x Sebagai Parameter = " + x);

System.out.println("Data member x di class Child = " + this.x);

System.out.println("Data member x di class Parent = " + super.x);

}

}

public class NilaiX {

public static void main(String[] args) {

Child tes = new Child();

tes.info();

}

}

Penjelasan : Pada program ini terdapat class Parent sebagaai parent class dari childclass nya yaitu Child dan Nilai X(untuk fungsi main). Pada class child dapat dilihat terdapat pemanggilan “super.x”. “super.x” digunakan untuk memanggil value dari variable yang ada di parent class.

# Percobaan 2 :

public class Pegawai {

private String nama; //Seharus nya public

public double gaji;

}

Public class Manajer extends Pegawai{ //seharusnya tidak perlu diisi public

public String departemen;

public void isiData(String n, String d){

nama=n;

departemen=d;

}

}

Penjelasan : ada beberapa hal yang membuat program ini error, salah satu nya Pada parent class pegawai, terdapat field nama yang bersifat private. Seharusnya sifat dari field tersebut adalah public, agar bias diakses pada child class.

Dan pada pembuatan child class juga terdapat masalah, yaitu modifier dari class tersebut, seharusnya default saja, jangan ditambahkan public. Karena parent class sudah menggunakan class public.

# Percobaan 3 :

Pada percobaan 3m constructor berada pada class child, jadi tidak bisa dijalankan. Karena masih belum ada fungsi main, source code belum error

# Percobaan 4 :

public class Employee {

private static final double BASE\_SALARY = 15000.00;

private String Name="";

private double Salary = 0.0;

private Date birthDate;

public Employee(){}

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;

}

public double getSalary() {

return Salary;

}

}

class Manager extends Employee{

//tambahan attribut private String departement

public String departement;

public Manager(String name, double salary, String dept){

super(name,salary);

departement = dept;

}

public Manager (String n, String dept){

super(n);

departement = dept;

}

public Manager(String dept){

super();

departement=dept;

}

public String GetDept(){

return departement;

}

}

public class TestManager {

public static void main(String[] args) {

Manager Utama = new Manager("John",5000000,"Financial");

System.out.println("Name:"+ Utama.getName());

System.out.println("Salary:"+ Utama.getSalary());

System.out.println("Department:"+ Utama.GetDept());

Utama = new Manager("Michael","Accounting");

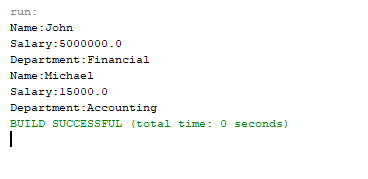
System.out.println("Name:"+ Utama.getName());

System.out.println("Salary:"+ Utama.getSalary());

System.out.println("Department:"+ Utama.GetDept());

}

}



Penjelasan : Program ini memiliki 1 class parent yaitu employee dan memiliki 2 class child yaitu Manager dan TestManager(sebagai main).

Pada class parent berisikan penginisialisasian attribute, constructor untuk inisialisasi dari field serta membuat getter untuk mengambil nilai dari field nantinya.

Pada class child yaitu employee ditambahkan sebuah field baru, yaitu department.

Pada Class TestManager berisikan fungsi main yang digunakan untuk menjalankan program

# Percobaan 5

public class MoodyObject {

protected String getMood(){

return"moody";

}

public void speak(){

System.out.println("I am"+getMood());

}

void laugh(){}

void cry(){}

}

public class SadObject extends MoodyObject {

@Override

protected String getMood(){

return "sad";

}

@Override

public void cry(){

System.out.println("Hoo hoo");

}

}

public class HappyObject extends MoodyObject {

@Override

protected String getMood(){

return"happy";

}

@Override

public void laugh(){

System.out.println("Hahahaha");

}

}

public class MoodyTest {

public static void main(String[]args){

MoodyObject m = new MoodyObject();

//test parent class

m.speak();

//test inheritance class

m = new HappyObject();

m.speak();

m.cry();

//test inheritance class

m = new SadObject();

m.speak();

m.cry();

}

}

Penjelasan :

# Percobaan 6

public class A {

String var\_a = "Variabel A";

String var\_b = "Variabel B";

String var\_c = "Variabel C";

String var\_d = "Variabel D";

A(){

System.out.println("Konstruktor A dijalankan");

}

}

public class B extends A {

B(){

System.out.println("Konstruktor B dijalankan");

var\_a = "Var\_a dari class B";

var\_b = "Var\_b dari class B";

}

public static void main(String args[]){

System.out.println("Object A dibuat");

A aa= new A();

System.out.println("menampilkan nama variabel obyek aa");

System.out.println("aa.var\_a");

System.out.println("aa.var\_b");

System.out.println("aa.var\_c");

System.out.println("aa.var\_d");

System.out.println("");

System.out.println("Object B dibuat");

B bb= new B();

System.out.println("menampilkan nama variabel obyek bb");

System.out.println("bb.var\_a");

System.out.println("bb.var\_b");

System.out.println("bb.var\_c");

System.out.println("bb.var\_d");

}

}

Penjelasan :

# Percobaan 7

public class Bapak {

int a;

int b;

void show\_variable(){

System.out.println("Nilai a="+ a);

System.out.println("Nilai b="+ b);

}

}

public class Anak extends Bapak {

int c;

void show\_variabel(){

System.out.println("Nilai a="+ a);

System.out.println("Nilai b="+ b);

System.out.println("Nilai c="+ c);

}

}

public class InheritExample {

public static void main(String[] args){

Bapak objectBapak = new Bapak();

Anak objectAnak = new Anak();

objectBapak.a=1;

objectBapak.b=1;

System.out.println("Object Bapak (Superclass):");

objectBapak.show\_variable();

objectAnak.c=5;

System.out.println("Object Anak (Superclass dari Bapak ):");

objectAnak.show\_variabel();

}

}

Penjelasan :

# Percobaan 8

public class Parent {

String parentName;

Parent(){

}

Parent (String parentName){

this.parentName = parentName;

System.out.println("Konstruktor parent");

}

}

public class Baby extends Parent {

String babyName;

Baby(String babyName){

super();

this.babyName = babyName;

System.out.println("Konstruktor Baby");

System.out.println(babyName);

}

public void Cry(){

System.out.println("Owek owek");

}

}

Penjelasan :