

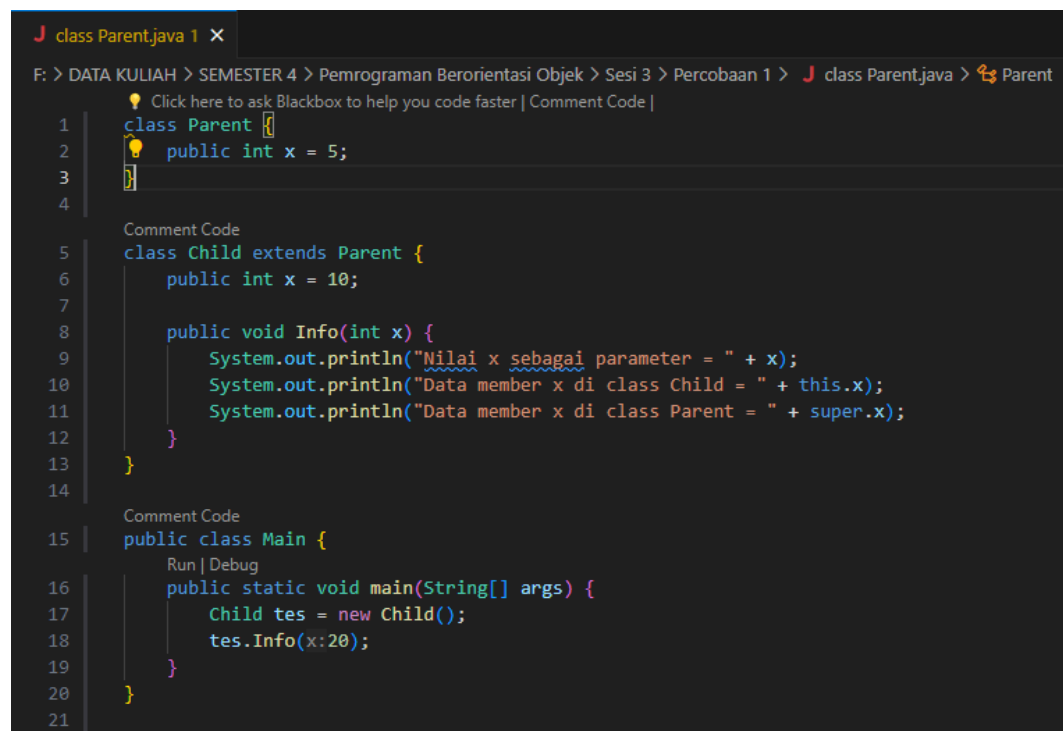
Nama : Haikal Muhammad Kurniawan
Nim : 20220040008
Kelas : TI22M

Percobaan 1

Ada beberapa kesalahan dalam kode tersebut:

1. Tidak ada import yang diperlukan.
2. Penulisan "super.x" yang salah pada baris ***System.out.println("Data member x di class Parent = 11 + super.x");***, seharusnya menjadi "super.x" saja tanpa angka 11.
3. Terdapat spasi yang tidak diperlukan sebelum kata "Info" pada baris ***tes.Info (20);***.
4. Penamaan kelas "Nilaix" yang sebaiknya diubah menjadi sesuatu yang lebih deskriptif atau relevan.

Berikut adalah perbaikan kode tersebut:



```
class Parent.java 1 X
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 1 > class Parent.java > Parent
Click here to ask Blackbox to help you code faster | Comment Code |
1 class Parent {
2     public int x = 5;
3 }
4
5 class Child extends Parent {
6     public int x = 10;
7
8     public void Info(int x) {
9         System.out.println("Nilai x sebagai parameter = " + x);
10        System.out.println("Data member x di class Child = " + this.x);
11        System.out.println("Data member x di class Parent = " + super.x);
12    }
13 }
14
15 public class Main {
16     public static void main(String[] args) {
17         Child tes = new Child();
18         tes.Info(x:20);
19     }
20 }
21
```

Perubahan yang dilakukan:

1. Menghilangkan spasi sebelum kata "Info".

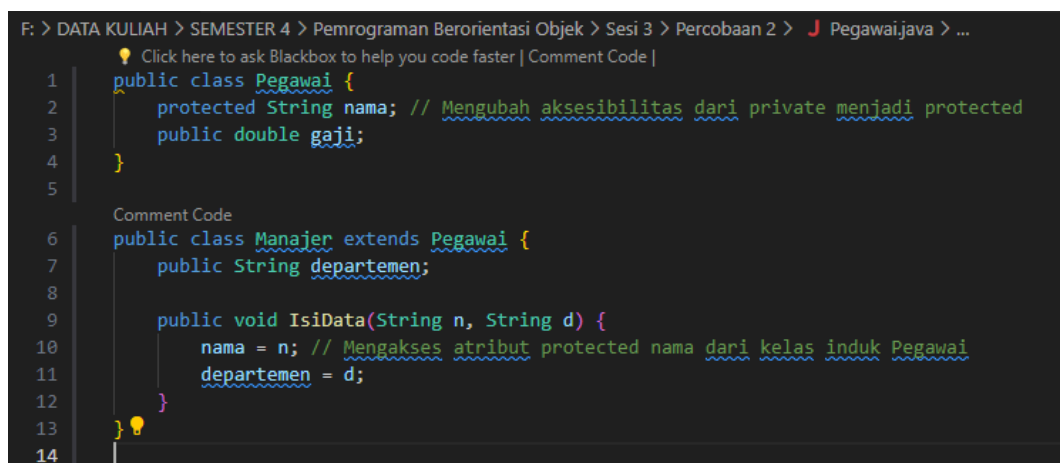
2. Mengubah "11 + super.x" menjadi "super.x".
3. Mengubah nama kelas menjadi "Main".
4. Menggunakan kurung kurawal yang lebih konsisten untuk blok kode.

Percobaan 2

Kode yang diberikan mengalami kesalahan karena atribut **nama** dari kelas **Pegawai** memiliki akses privat (**private**), sehingga tidak dapat diakses langsung dari kelas turunannya (**Manajer**). Dalam Java, atribut dengan akses privat hanya dapat diakses secara langsung dari dalam kelas yang sama.

Solusinya adalah dengan mengubah aksesibilitas atribut **nama** menjadi **protected** atau **public** agar dapat diakses dari kelas turunannya (**Manajer**).

Berikut adalah kode yang sudah diperbaiki:



```
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 2 > J Pegawai.java > ...
Click here to ask Blackbox to help you code faster | Comment Code |
1 public class Pegawai {
2     protected String nama; // Mengubah aksesibilitas dari private menjadi protected
3     public double gaji;
4 }
5
Comment Code
6 public class Manajer extends Pegawai {
7     public String departemen;
8
9     public void IsiData(String n, String d) {
10         nama = n; // Mengakses atribut protected nama dari kelas induk Pegawai
11         departemen = d;
12     }
13 }
14
```

Perubahan yang dilakukan:

1. Mengubah aksesibilitas atribut **nama** dalam kelas **Pegawai** menjadi **protected**.
2. Mengakses atribut **nama** dari kelas turunan **Manajer** menggunakan aksesibilitas **protected** yang telah diperbaiki.

Percobaan 3

Dalam Java, ketika membuat kelas turunan (**child class**), konstruktor default (**tanpa parameter**) dari kelas induk (**parent class**) akan dipanggil secara implisit kecuali ada konstruktor yang didefinisikan di kelas induk yang membutuhkan parameter.

Dalam kasus ini, kelas induk (**Parent**) tidak memiliki konstruktor default, sehingga ketika kelas turunan (**Child**) mencoba membuat konstruktor tanpa parameter, akan terjadi kesalahan karena konstruktor yang diwariskan secara implisit tidak ditemukan.

Solusinya adalah dengan menambahkan konstruktor default (tanpa parameter) di kelas induk (**Parent**) atau secara eksplisit memanggil konstruktor kelas induk yang tersedia menggunakan kata kunci **super()** dalam konstruktor kelas turunan (**Child**).

Berikut adalah contoh perbaikan dengan menambahkan konstruktor default di kelas induk (**Parent**):



```
1  public class Parent {
2      // Konstruktor default
3      public Parent() {
4          // Kosong
5      }
6  }
7
8  public class Child extends Parent {
9      int x;
10
11     public Child() {
12         x = 5;
13     }
14 }
15
```

Dalam perbaikan ini, konstruktor default ditambahkan ke kelas **Parent** untuk memastikan bahwa kelas turunan (**Child**) dapat mengaksesnya secara implisit.

Percobaan 4

Terdapat beberapa kesalahan:

1. Typo pada pemanggilan konstruktor pada objek **Utama** di dalam method **main**.
2. Pada konstruktor di kelas **Employee**, terdapat kesalahan penulisan konstanta **BASE_SALARY**.
3. Penulisan **DOB** pada konstruktor kelas **Employee** yang seharusnya **DoB**.
4. Pada kelas **TestManager**, pada pemanggilan konstruktor kedua objek **Utama** terdapat kesalahan sintaksis.

Terdapat beberapa kesalahan:

1. Typo pada pemanggilan konstruktor pada objek **Utama** di dalam method **main**.
2. Pada konstruktor di kelas **Employee**, terdapat kesalahan penulisan konstanta **BASE_SALARY**.
3. Penulisan **DOB** pada konstruktor kelas **Employee** yang seharusnya **DoB**.
4. Pada kelas **TestManager**, pada pemanggilan konstruktor kedua objek **Utama** terdapat kesalahan sintaksis.

Berikut adalah perbaikan kode tersebut:

```
1  import java.util.Date;
2
3  class Employee {
4      private static final double BASE_SALARY = 15000.00;
5      private String Name = "";
6      private double Salary = 0.0;
7      private Date birthDate;
8
9      public Employee() {}
10
11     public Employee(String name, double salary, Date DoB) {
12         this.Name = name;
13         this.Salary = salary;
14         this.birthDate = DoB;
15     }
16
17     public Employee(String name, double salary) {
18         this(name, salary, null);
19     }
20
21     public Employee(String name, Date DoB) {
22         this(name, BASE_SALARY, DoB);
23     }
24
25     public Employee(String name) {
26         this(name, BASE_SALARY);
27     }
28
29     public String GetName() {
30         return Name;
31     }
32
33     public double GetSalary() {
34         return Salary;
35     }
36 }
37
38 class Manager extends Employee {
39     private String department;
40
41     public Manager(String name, double salary, String dept) {
42         super(name, salary);
43         department = dept;
44     }
45
46     public Manager(String n, String dept) {
47         super(n);
48         department = dept;
49     }
50
51     public Manager(String dept) {
52         super();
53         department = dept;
54     }
55
56     public String GetDept() {
57         return department;
58     }
59 }
60
61 public class TestManager {
62     public static void main(String[] args) {
63         Manager Utama = new Manager("John", 5000000, "Financial");
64         System.out.println("Name:" + Utama.GetName());
65         System.out.println("Salary:" + Utama.GetSalary());
66         System.out.println("Department:" + Utama.GetDept());
67
68         Manager newUtama = new Manager("Michael", "Accounting");
69         System.out.println("Name:" + newUtama.GetName());
70         System.out.println("Salary:" + newUtama.GetSalary());
71         System.out.println("Department:" + newUtama.GetDept());
72     }
73 }
74
```

Perubahan yang dilakukan:

1. Memperbaiki pemanggilan konstruktor pada objek *Utama*.
2. Mengubah *DOB* menjadi *DoB* pada konstruktor kelas *Employee*.
3. Mengubah *Utama new Manager("Michael", "Accounting");* menjadi *Manager newUtama = new Manager("Michael", "Accounting");* pada kelas *TestManager*.

Percobaan 5

Terdapat beberapa kesalahan:

1. Metode *cry()* tidak ada dalam kelas *MoodyObject* tetapi dideklarasikan di luar kelas, sehingga menyebabkan kesalahan kompilasi.
2. Deklarasi metode *laugh()* pada kelas *MoodyObject* tidak memiliki body metode, dan juga deklarasi tersebut tidak perlu ada di kelas induk karena metode itu hanya ada pada kelas *HappyObject*.

Berikut adalah perbaikan kode tersebut:

```
J MoodyObject.java 1 x
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 5 > J MoodyObject.java > MoodyObject

Click here to ask Blackbox to help you code faster | Comment Code |
1 public class MoodyObject {
2     protected String getMood() {
3         return "moody";
4     }
5
6     public void speak() {
7         System.out.println("I am " + getMood());
8     }
9 }
10
Comment Code
11 public class SadObject extends MoodyObject {
12     protected String getMood() {
13         return "sad";
14     }
15
16     public void cry() {
17         System.out.println("Hoo hoo");
18     }
19 }
20
Comment Code
21 public class HappyObject extends MoodyObject {
22     protected String getMood() {
23         return "happy";
24     }
25
26     public void laugh() {
27         System.out.println("Hahaha");
28     }
29 }
30
Comment Code
31 public class MoodyTest {
32     Run | Debug
33     public static void main(String[] args) {
34         MoodyObject m = new MoodyObject();
35         m.speak();
36
37         m = new HappyObject();
38         m.speak();
39         ((HappyObject) m).laugh();
40
41         m = new SadObject();
42         m.speak();
43         ((SadObject) m).cry();
44     }
45 }
```

Perubahan yang dilakukan:

1. Menghapus deklarasi metode *cry()* dan *laugh()* dari kelas *MoodyObject* karena keduanya hanya ada di kelas turunannya.
2. Mengubah penggunaan objek dan memanggil metode *laugh()* dan *cry()* sesuai dengan jenis objeknya.

Percobaan 6

Terdapat beberapa kesalahan:

1. Terdapat kesalahan penulisan nama kelas di kelas *B*, yang seharusnya adalah *extends A*, bukan *extends Al*.
2. Terdapat kesalahan penulisan dalam deklarasi variabel pada konstruktor kelas *B*.
3. Terdapat kesalahan dalam pemanggilan variabel pada method *main()*, karena variabel yang dipanggil seharusnya dimulai dengan huruf kecil *var_a*, *var_b*, *var_c*, dan *var_d*.

Berikut adalah perbaikan dari kode tersebut:

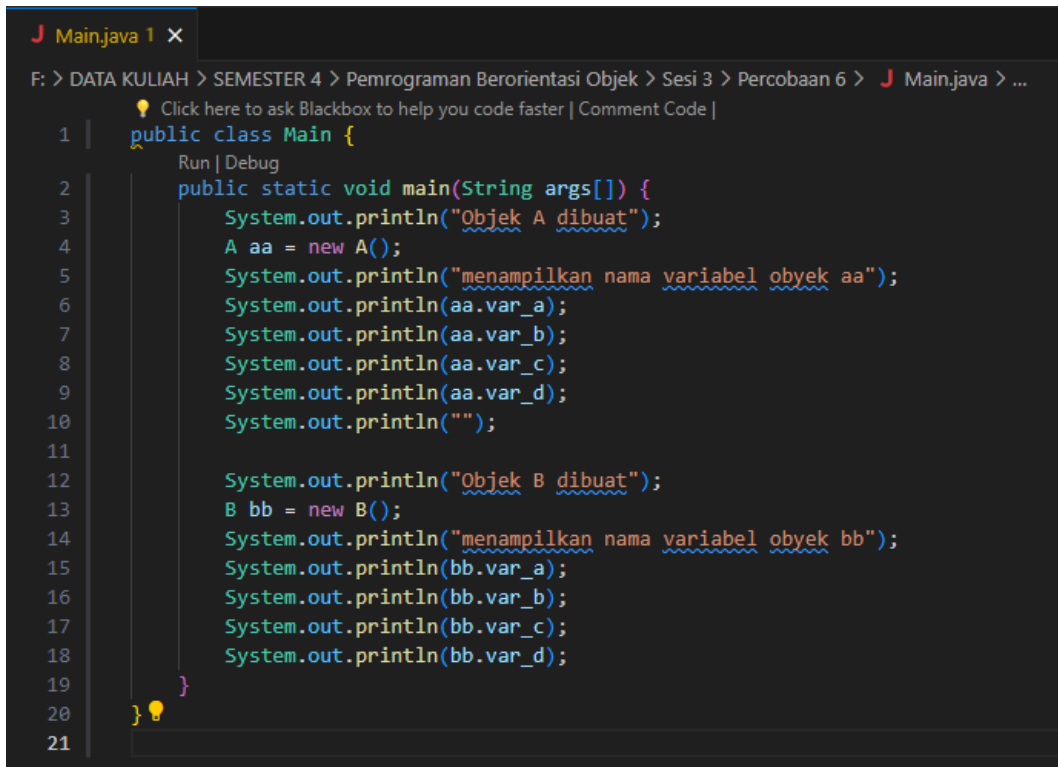
Kelas A (A.java):

```
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 6 > J A.java > ...
Click here to ask Blackbox to help you code faster | Comment Code |
1 public class A {
2     String var_a = "Variabel A";
3     String var_b = "Variabel B";
4     String var_c = "Variabel C";
5     String var_d = "Variabel D";
6
7     public A() {
8         System.out.println("Konstruktor A dijalankan");
9     }
10 }
11
```

Kelas B (B.java):

```
J B.java 1 x
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 6 > J B.java > ...
Click here to ask Blackbox to help you code faster | Comment Code |
1 public class B extends A {
2     B() {
3         System.out.println("Konstruktor B dijalankan ");
4         var_a = "Var a dari class B";
5         var_b = "Var a dari class B";
6     }
7 }
8 +
```

Main Class (Main.java):



```
1 public class Main {
2     public static void main(String args[]) {
3         System.out.println("Objek A dibuat");
4         A aa = new A();
5         System.out.println("menampilkan nama variabel obyek aa");
6         System.out.println(aa.var_a);
7         System.out.println(aa.var_b);
8         System.out.println(aa.var_c);
9         System.out.println(aa.var_d);
10        System.out.println("");
11
12        System.out.println("Objek B dibuat");
13        B bb = new B();
14        System.out.println("menampilkan nama variabel obyek bb");
15        System.out.println(bb.var_a);
16        System.out.println(bb.var_b);
17        System.out.println(bb.var_c);
18        System.out.println(bb.var_d);
19    }
20 }
21
```

Perubahan yang dilakukan:

1. Penyesuaian penamaan kelas B yang seharusnya adalah *extends A*.
2. Penyesuaian penulisan variabel pada konstruktor kelas B agar sesuai dengan sintaks yang benar.
3. Penyesuaian pemanggilan variabel pada method *main()* agar sesuai dengan nama variabel yang telah dideklarasikan.

Percobaan 7

Terdapat beberapa kesalahan:

1. Terdapat kesalahan dalam penulisan nama method *show variabel* dan *show_variabel*.
2. Penulisan method *show_variabel()* pada kelas *Anak* tidak menggunakan keyword *@Override*.
3. Pada method *show_variabel()* pada kelas *Anak*, nilai variabel *a* dan *b* tidak diambil dari superclass menggunakan *super*.
4. Pada pemanggilan method *show_variabel()* pada objek *objectAnak*, terdapat kesalahan penulisan spasi antara *show* dan *variabel*.

Berikut adalah perbaikan dari kode tersebut:

```
J Bapak.java 1 x
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 7 > J Bapak.java > Bapak
Click here to ask Blackbox to help you code faster | Comment Code |

1 class Bapak {
2     int a;
3     int b;
4     void show_variabel() {
5         System.out.println("Nilai a="+ a);
6         System.out.println("Nilai b="+ b);
7     }
8 }
9
Comment Code
10 class Anak extends Bapak {
11     int c;
12     @Override
13     void show_variabel() {
14         super.show_variabel();
15         System.out.println("Nilai c="+ c);
16     }
17 }
18
Comment Code
19 public class InheritExample {
20     Run | Debug
21     public static void main(String[] args) {
22         Bapak objectBapak = new Bapak();
23         Anak objectAnak = new Anak();
24         objectBapak.a = 1;
25         objectBapak.b = 1;
26         System.out.println("Object Bapak (Superclass): ");
27
28         objectBapak.show_variabel();
29         objectAnak.c = 5;
30         System.out.println("Object Anak (Superclass dari Bapak):");
31         objectAnak.show_variabel();
32     }
33 }
```

Perubahan yang dilakukan:

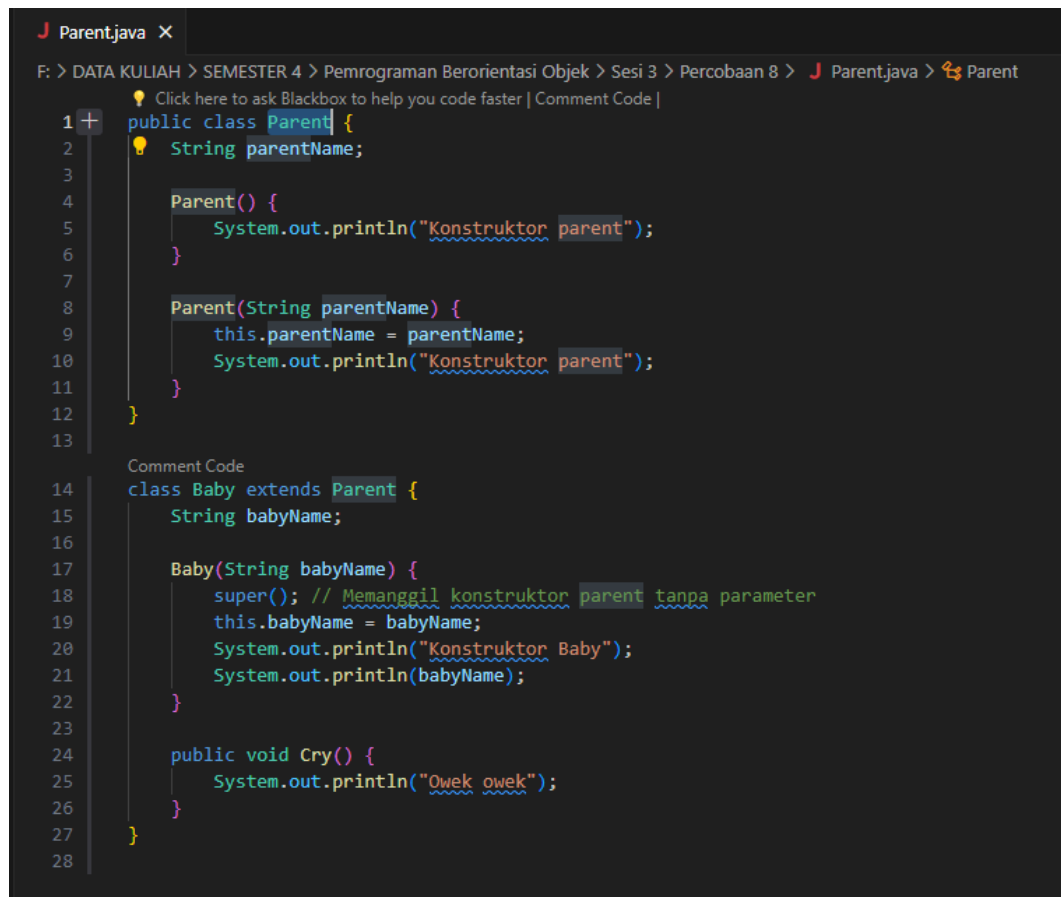
1. Memperbaiki penulisan nama method menjadi *show_variabel*.
2. Menambahkan annotation *@Override* pada method *show_variabel()* di kelas *Anak*.
3. Menggunakan *super.show_variabel()* di dalam method *show_variabel()* di kelas *Anak* untuk memanggil method dari superclass dan menampilkan nilai variabel *a* dan *b*.
4. Memperbaiki penulisan pemanggilan method *show_variabel()* pada objek *objectAnak*.

Percobaan 8

Terdapat beberapa kesalahan:

1. Konstruktor kelas **Baby** tidak memanggil konstruktor superclass dengan parameter *parentName*.
2. Pada kelas **Parent**, tidak ada konstruktor yang tanpa parameter untuk dipanggil secara implisit ketika menggunakan *super()* di konstruktor kelas **Baby**.

Berikut adalah perbaikan dari kode tersebut:



```
Parent.java
F: > DATA KULIAH > SEMESTER 4 > Pemrograman Berorientasi Objek > Sesi 3 > Percobaan 8 > Parent.java > Parent

Click here to ask Blackbox to help you code faster | Comment Code |

1 + public class Parent {
2     String parentName;
3
4     Parent() {
5         System.out.println("Konstruktor parent");
6     }
7
8     Parent(String parentName) {
9         this.parentName = parentName;
10        System.out.println("Konstruktor parent");
11    }
12 }
13
Comment Code
14 class Baby extends Parent {
15     String babyName;
16
17     Baby(String babyName) {
18         super(); // Memanggil konstruktor parent tanpa parameter
19         this.babyName = babyName;
20         System.out.println("Konstruktor Baby");
21         System.out.println(babyName);
22     }
23
24     public void Cry() {
25         System.out.println("Owek owek");
26     }
27 }
28
```

Perubahan yang dilakukan:

1. Pada konstruktor kelas **Baby**, menambahkan pemanggilan konstruktor superclass *super()* tanpa parameter untuk memanggil konstruktor superclass yang sesuai.
2. Menghapus parameter *parentName* pada konstruktor **Parent** yang kosong, karena tidak digunakan dalam kode.