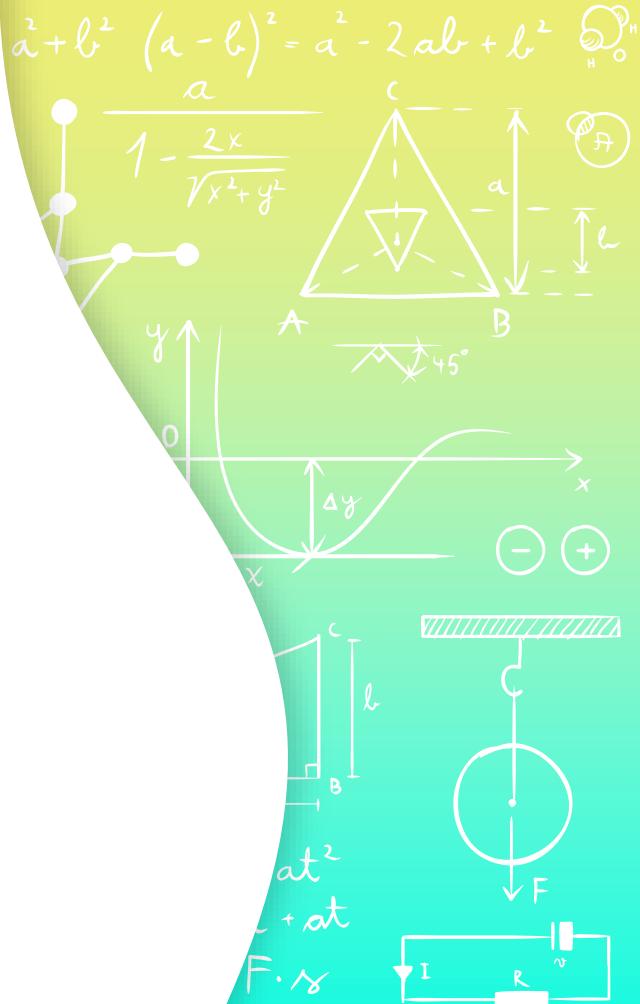


Pemrograman Berorientasi Objek Pewarisan

Tim Tentor KSP Java 2023



Pewarisan / Inheritance

Sebuah konsep Relasi “*is-a*” antar kelas, dimana terdapat pengelompokan beberapa kelas yang memiliki “**beberapa kemiripan**” antar satu dengan yang lain. Sehingga membentuk sebuah struktur hirarki (*terdapat kelas induk dan kelas anak*)



$$\iiint x^2 dx dy dz =$$

$$V: z = 10(x+3y), x+y=1$$

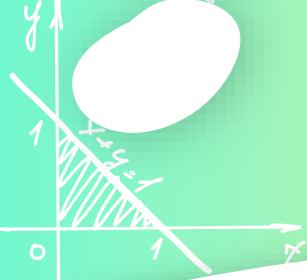
$$x=0, y=0, z=0$$

$$= \int_0^1 \int_0^{1-x} \int_{-10(x+3y)}^5 dx dy dz$$

$$\frac{1}{\sqrt{2}} \arcsin y$$

?

$$2\sqrt{y^2 - x^2}$$



$$x = 2y^2 - 3, x = 5$$

$$z = 1 + \sqrt{9x^2 + 4y^2}$$

$$z = 4 + \sqrt{9x^2 + 4y^2}$$

$$V = \int_0^1 dy \int_{-1}^5 dx \int_{\frac{4+\sqrt{9x^2+4y^2}}{2y^2+3}}^{\frac{4+\sqrt{9x^2+4y^2}}{1+\sqrt{9x^2+4y^2}}} dz$$



$$x = 2y^2 - 3$$

Is-a (?)

Kelas A "merupakan" kelas B

Relasi “is-a”

Relasi “is-a” merupakan sebuah relasi yang menyatakan bahwa “Kelas A” merupakan “Kelas B”. Relasi ini didapatkan dari pengelompokan beberapa kelas yang memiliki beberapa **atribut / method** yang sama.

Contoh :

- Laptop merupakan (“is-a”) BarangElektronik
- Handphone merupakan (“is-a”) BarangElektronik
- Kulkas merupakan (“is-a”) BarangElektronik

Pada Relasi “is-a”, terdapat kelas yang akan “mewarisi” beberapa **atribut / method** – nya dan juga ada kelas yang akan mendapatkan “warisan” tersebut. inilah mengapa konsep ini disebut sebagai **Pewarisan / Inheritance**

pada contoh di atas, kelas yang “mewariskan” {**kelas Induk, Superclass, Kelas Dasar**} adalah Kelas **BarangElektronik**.

Sedangkan kelas **Laptop, Handphone**, dan **Kulkas** merupakan kelas yang “diwariskan” {**Kelas Anak, Subclass, Kelas Turunan**}

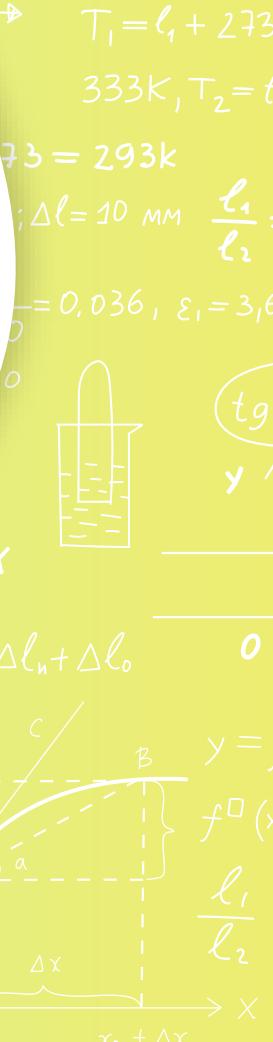
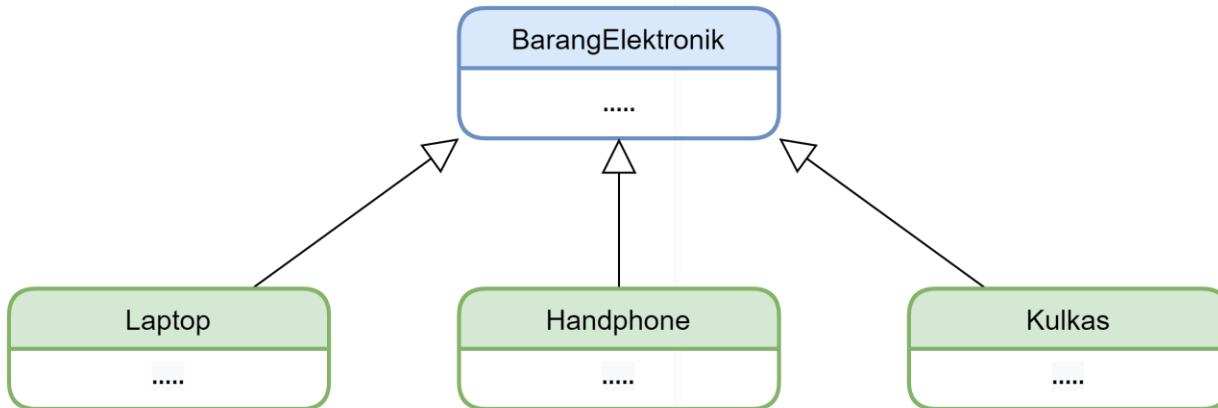


Diagram 1.1



Jika digambarkan, maka struktur dari Kelas kita adalah sebagai berikut.

***Jangan sampai salah dalam penggambaran tanda panah,
dikarenakan gambar tanda panah yang berbeda memiliki arti yang
berbeda pula :)***

Pewarisan / Inheritance

Struktur pada kelas yang kita miliki bisa saja **bertingkat**. Artinya, sebuah **Subclass** bisa saja menjadi **Superclass** untuk **Subclass** lainnya. Perlu diketahui juga bahwa kita tidak boleh memiliki struktur yang *circular*

Contoh Circular :

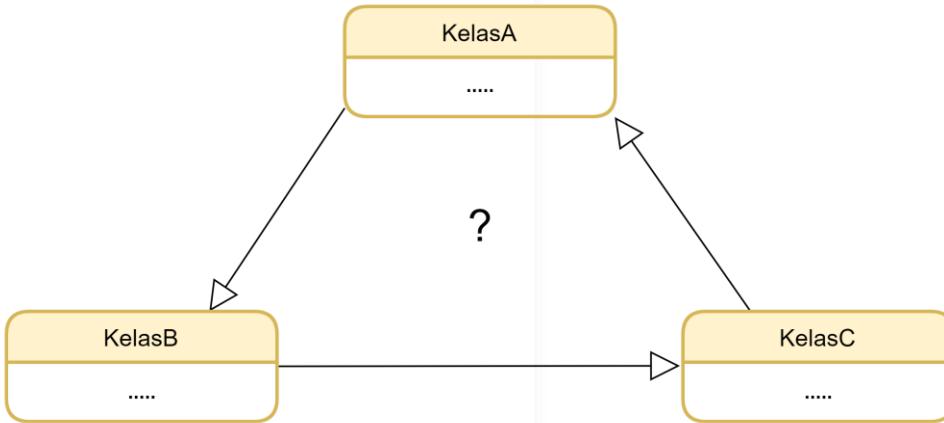
- Kelas A merupakan ("is-a") Kelas B
- Kelas B merupakan ("is-a") Kelas C
- Kelas C merupakan ("is-a") Kelas A

Contoh Struktur Bertingkat :

- BarangElektronik merupakan ("is-a") Barang
- BarangNonElektronik merupakan ("is-a") Barang
- Laptop merupakan ("is-a") BarangElektronik

Diagram 1.2

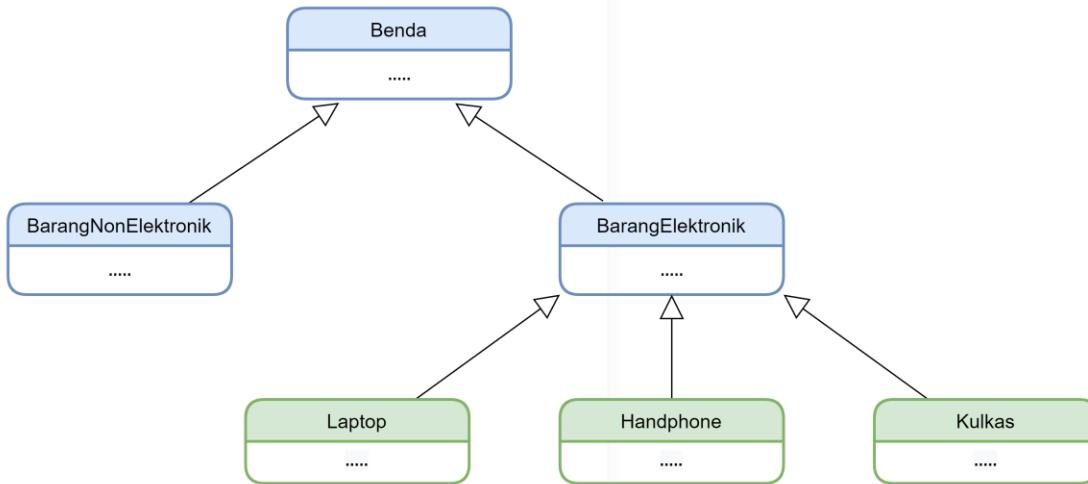
Circular Inheritance



Ini contoh salah yaaa.... Struktur seperti ini tidak diperbolehkan... :")

Diagram 1.3

Multilevel Inheritance



Selama memori komputer teman-teman masih memungkinkan, maka bisa dikatakan tidak ada batasan dalam berapa jumlah level yang dibolehkan.

Tentu disesuaikan juga dengan permasalahan yang teman-teman hadapi.

$$\iiint x^2 dx dy dz =$$

$$V: z = 10(x+3y), x+y=1 \\ x=0, y=0, z=0$$

$$= \int_0^1 \int_0^{1-x} \int_{x+3y}^{10} dx dy dz$$

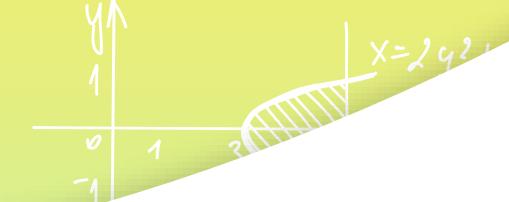
$$\int_0^{1/\sqrt{2}} dy \int_0^{\arcsin y} dx$$

$$2\sqrt{y^2 - x^2}$$



$$x = 2y^2 - 3, x = 5 \\ z = 1 + \sqrt{9x^2 + 4y^2} \\ z = 4 + \sqrt{9x^2 + 4y^2}$$

$$V = \int_{-1}^1 dy \int_{2y^2+3}^5 dx \int_{4+\sqrt{9x^2+4y^2}}^{\infty} dz$$



Aksesibilitas Anggota (Visibilitas)

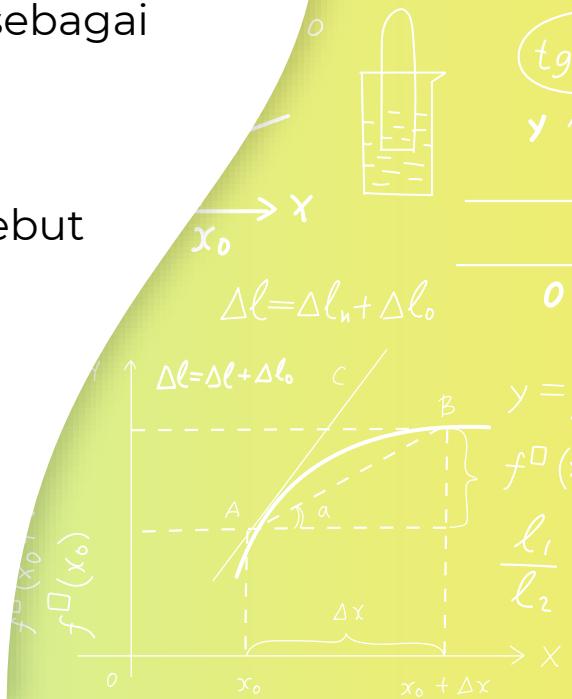
Public, Private, dan Protected

Aksesibilitas Anggota

Seperti yang telah kita ketahui, jika kita mendeklarasikan sebuah atribut atau method untuk kelas tertentu, maka kita memerlukan kata kunci **public/private/protected** sebagai **prefix-nya**.

Ketiga keyword tersebut merupakan **aksesibilitas** dari atribut/method. disebut aksesibilitas karena keyword tersebut akan **menentukan** siapa saja yang **boleh mengakses** atribut/method bersangkutan.

Untuk mempermudah, kita akan menggunakan kata "**Visibilitas**" untuk menggantikan keyword ini yaa... :)



Visibilitas Atribut/Method



Public (+)

Memungkinkan atribut/method agar dapat **digunakan oleh siapapun**. tidak terbatas oleh Kelas, Package, dll



Protected (#)

Memungkinkan atribut/method agar dapat **digunakan oleh Kelas Turunan dan Anggota dari Package yang sama**



Private (-)

Atribut/method **tidak dapat digunakan oleh Kelas Lain**. Hanya dapat digunakan oleh **dirinya sendiri**.

Satu-satunya cara **akses/manipulasi** adalah melalui **method di dirinya sendiri**.

$$\iiint x^2 dx dy dz =$$

$$V: z = 10(x+3y), x+y=1$$

$$x=0, y=0, z=0$$

$$= \int_0^1 \int_0^{1-x} \int_{-10(x+3y)}^5 dx dy dz$$

$$\int_0^{1/\sqrt{2}} dy \int_{-\infty}^{\arcsin y} dx$$

$$2\sqrt{y^2 - x^2}$$



$$x = 2y^2 - 3, x = 5$$

$$z = 1 + \sqrt{9x^2 + 4y^2}$$

$$z = 4 + \sqrt{9x^2 + 4y^2}$$

$$V = \int_{-1}^1 dy \int_{2y^2+3}^5 dx \int_{1+\sqrt{9x^2+4y^2}}^{4+\sqrt{9x^2+4y^2}} dz$$



Breakdown The Code

Pelan-pelan aja dulu ygy... :))

Class Structure...

Laptop
+ kapasitasBaterai : Int
+ noSeri : String
+ statusPower : Bool
+ CPU : String
+ switchPower (bool) : void
+ getStatusBaterai () : int
+ setTerang (int) : void
+ buildProgram (String) : void

Handphone
+ kapasitasBaterai : Int
+ noSeri : String
+ statusPower : Bool
+ keamananBiometrik : String
+ switchPower (bool) : void
+ getStatusBaterai () : int
+ ambilFoto () : void
+ cekBiometrik (String) : Bool

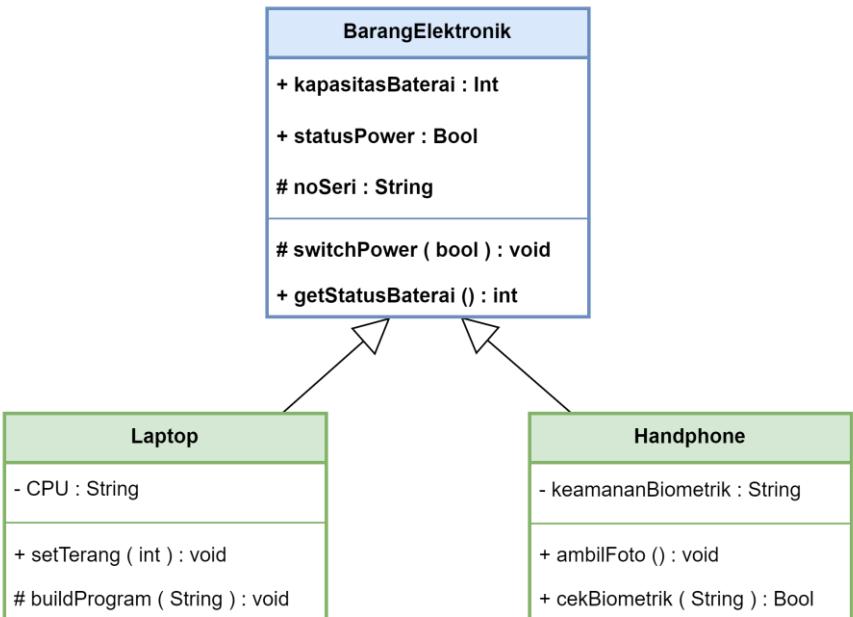
Laptop & Handphone

Straight to the point aja... :)

Anggapanlah kita memiliki **2 kelas** seperti pada gambar disamping.

Bisa dilihat bahwa terdapat **beberapa** atribut dan/atau method **yang mirip**. Sehingga, kita bisa membuat **kelas induk** yang nantinya akan memiliki atribut dan/atau method yang **duplicat** tersebut.

Class Structure...



Barang Elektronik & Laptop dan Handphone

Nah... yang terjadi adalah....
kita memisahkan atribut+method yang sama
tadi ke **Kelas BarangElektronik**.

Lalu kita membuat **Kelas Laptop** dan **Kelas Handphone** sebagai **Subclass** dari **Kelas BarangElektronik**

Perlu diingat bahwa, **Subclass** akan **mewarisi semua** atribut+method yang dimiliki oleh **Superclass**, disesuaikan dengan **visibilitas** dari atribut/method bersangkutan.

Code Superclass...



```
BarangElektronik.java x
Source History ... #:
1 package mytesting;
2
3
4 public class BarangElektronik {
5
6     public int kapasitasBaterai;
7     public boolean statusPower;
8     protected String noSeri;
9
10    private int dayaListrik;
11
12    public BarangElektronik(int kapasitasBaterai, String noSeri) {
13        this.kapasitasBaterai = kapasitasBaterai;
14        this.noSeri = noSeri;
15
16        this.statusPower = false;
17        this.dayaListrik = 100;
18    }
19
20    protected void switchPower(boolean value){
21        this.statusPower = value;
22    }
23
24    public int getStatusBaterai(){
25        return this.kapasitasBaterai;
26    }
27
28}
29
30}
```

Barang Elektronik

untuk **Superclass** sendiri, kita hanya perlu menuliskannya seperti **kelas biasa** sesuai dengan **Diagram Kelas** yang ada. Tidak ada yang spesial yang perlu ditambahkan pada **Superclass**.

Note :

Silahkan perhatikan **line 11**. saya menambahkan satu atribut *private* berupa **dayaListrik**. Atribut ini tidak lain hanya digunakan untuk **memberikan contoh** pewarisan dengan **visibilitas private**.

Code Subclass...



```
BarangElektronik.java x Laptop.java x
Source History ... 1
2 package mytesting;
3
4 public class Laptop extends BarangElektronik{
5
6     private String cpu;
7
8     public Laptop(int kapasitasBaterai, String noSeri, String cpu){
9         super(kapasitasBaterai, noSeri);
10        this.cpu = cpu;
11    }
12
13    public void setTerang(int value){
14        System.out.println("Set Keterangan Menjadi " + value);
15    }
16
17    protected void buildProgram(String program){
18        System.out.println("Build : " + program);
19    }
20
21 }
22
23
24 }
```

Laptop

Cara sebuah Kelas untuk menjadi **Subclass** dari suatu kelas lainnya adalah dengan menggunakan syntax **extends**.

Pada contoh, dikarenakan **Kelas Laptop** akan menjadi **Subclass**, dan **Kelas BarangElektronik** akan menjadi **Superclass**. Maka kita menuliskan

Public class Laptop extends BarangElektronik

Syntax ini akan menunjukkan bahwa **Kelas Laptop** merupakan sebuah **Kelas Turunan** dari **Kelas Elektronik**

Code Subclass...



```
1 package mytesting;
2
3 public class Handphone extends BarangEletronik{
4
5     private String keamananBiometrik;
6
7     public Handphone(int kapasitasBaterai, String noSeri, String keamanan){
8
9         super(kapasitasBaterai, noSeri);
10
11         this.keamananBiometrik = keamanan;
12     }
13
14     public void ambilFoto(){
15         System.out.println("Cekrek... wkwkwk");
16     }
17
18     public boolean cekBiometrik(String value){
19         return value.equals(this.keamananBiometrik);
20     }
21
22
23 }
24
25
26
27
28 }
```

Handphone

Hal lainnya yang perlu diperhatikan adalah mengenai **Constructor**. Perlu diketahui bahwa **Constructor tidak akan diwarisi** meskipun memiliki **visibilitas public**.

Namun, **Subclass** dapat memanggil **Constructor** dari **Superclass** menggunakan syntax **Super()**

Parameter **Super()** disesuaikan dengan kebutuhan **constructor** milik **Superclass**. Apabila **Superclass** tidak membutuhkan parameter sama sekali pada Constructornya, maka pemanggilan **Super() dapat diabaikan**

Code main...



```
1 package mytesting;
2
3 public class MyTesting {
4
5     public static void main(String[] args) {
6
7         Laptop laptop = new Laptop(1000, "AAAAAA", "Intel Kadaluarsa");
8
9         System.out.println(laptop.noSeri);
10        // System.out.println(laptop.cpu);
11
12        laptop.switchPower(true);
13        System.out.println("Switch Power : " + laptop.statusPower);
14        System.out.println("Kapasitas Baterai : " + laptop.getStatusBaterai());
15
16        laptop.setTerang(500);
17        laptop.buildProgram("Kalkulator");
18
19    }
20
21 }
22
```

Output - MyTesting (run) - Editor

Output - MyTesting (run)

```
▶ AAAAAA
▶ Switch Power : true
▶ Kapasitas Baterai : 1000
▶ Set Keterangan Menjadi 500
▶ Build : Kalkulator
BUILD SUCCESSFUL (total time: 0 seconds)
```

Laptop

Nahh... pada contoh dapat terlihat program kita berjalan seperti biasa. bisa diperhatikan bahwa laptop tetap **dapat mengakses** atribut yang **diwariskan** pada-nya.

dibuktikan dengan **laptop** yang dapat **menampilkan noSeri** dan **statusPower**, serta kemampuan **laptop** untuk **menggunakan** method **getStatusBaterai** dan **switchPower** milik **Superclass**.

Note: jika teman-teman **uncomment line 11**, maka akan menimbulkan error akibat visibilitas **private**. Sehingga untuk mengakses **CPU** diperlukan **getter/setter**

Code main...



```
BarangElektronik.java x Laptop.java x Handphone.java x MyTesting.java x
Source History Run Build Clean Help
1
2 package mytesting;
3
4 public class MyTesting {
5
6     public static void main(String[] args) {
7
8         Handphone handphone = new Handphone(12000, "BBBB", "sidikku");
9
10        System.out.println(handphone.noSeri);
11        System.out.println(handphone.dataListrik);
12
13        handphone.ambilFoto();
14
15        if(handphone.cekBiometrik("sidikmu")){
16            System.out.println("Sukses Login");
17        }else {
18            System.out.println("Gagal Login");
19        }
20
21        System.out.println("Kapasitas Baterai : " + handphone.getStatusBaterai());
22
23    }
24
25
26
27
28
29
30
31
32
33
34
35
Output - MyTesting (run) - Editor
Output - MyTesting (run) x
run:
BBBB
Cekrek... wkwkwk
Gagal Login
Kapasitas Baterai : 12000
```

Handphone

Begitu pula pada **Kelas Handphone**. Kelas **Handphone** juga dapat **mengakses atribut** serta **menggunakan method** pada **Superclass** sama halnya seperti **Kelas Laptop**.

Lalu, tidak lupa untuk **atribut private** yang dimiliki oleh **Superclass**, yakni **dayaListrik**.

Apabila teman-teman **uncomment Line 11**, maka yang terjadi adalah program teman-teman akan **Error**. Hal ini kembali dikarenakan **visibilitas** atribut **dayaListrik** merupakan **private**. Dan **atribut/method** yang memiliki **visibilitas private** tidak akan **diwariskan** ke **Subclass**.

$$e = f^2(x+4gh)^2(s) \cdot (\wedge)^3 \div (gh)^2 - x^2$$

$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(hc)^3 + x^2 - 2x^2$$

$$g = x^2 \div (x)(2x)^2 + (hf)^2 4x^3(3h)(f)^2(e)^2 + x^2 4s^2$$

$$h = ef^2 - (x)^2 + (3)^2(f)^3 + x(4x)^2$$

$$(d)(ef)^2 = x^2$$

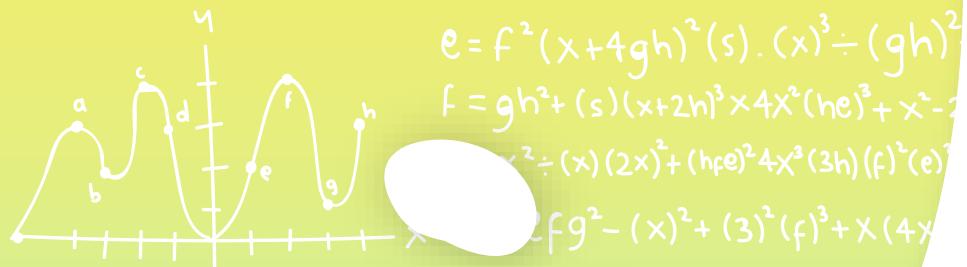
$$(b)^2 = \frac{4x^2 hd}{2s+4x}$$

$$\left. \begin{array}{l} ab = \frac{4x^2 + (ef)^2}{hc \cdot s^2(x)_3} \\ dc = \frac{3x^2 + ab(s)^3}{xy^3 - (x)(s)_1} \end{array} \right\}$$

Guided

Ketik Ulang

Code PPT yoo...



$$(x)^2 = ab$$

$$dh(x) = bc$$

$$a = x(s^1) + (h)(c) + (d)(e)f^2 = x^2$$

$$b = 2x + (h)(d) - (s^1)(h^2)(b)^2 =$$

$$c = 3x(c) da (2x)^3 \div (x)(x)^2 2x$$

$$d = \frac{x^4 + 4(s^1)(s)^2 + ab \div c^2(h)}{x^2(x)^1 s^1 + s^2 xy + c}$$



$$K = x^2 + (p)^2$$

$$P = 40^\circ (s)$$

$$O = X^2(x.Kp)^2 \div (4)^2 + (x)(K) = 23^\circ$$



Thanks!

Do you have any questions?

masing2tentor@gmail.com

+62 812 3456 789

Youtube.com



Please keep this slide for attribution

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by

Flaticon and infographics & images by

Freepik

