

**LAPORAN**  
**Jobsheet-12 : PHP-OOP**  
**TUGAS PRAKTIKUM PEMROGRAMAN WEB**

**Dibimbing oleh:** Bapak Dimas Wahyu Wibowo, S.T., M.T.



**Disusun oleh:**  
**ANARADI OCTA LAVECHIA**  
**2241760007 / SIB-2C**

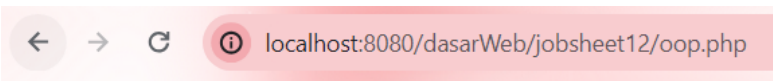
**JURUSAN TEKNOLOGI INFORMASI**  
**PRODI D-IV SISTEM INFORMASI BISNIS**  
**POLITEKNIK NEGERI MALANG**  
**2024**

## Praktikum 1. Basic OOP

Langkah	Keterangan
1	Kelas adalah blueprint atau cetak biru yang mendefinisikan struktur dan perilaku suatu objek. Kelas berisi atribut (data) dan metode (fungsi) yang berkaitan dengan objek tersebut. Objek, di sisi lain, adalah instance konkret dari suatu kelas, memiliki nilai nyata untuk atribut dan mampu menjalankan metode yang didefinisikan dalam kelas. Dalam PHP, Anda dapat membuat kelas dengan kata kunci class dan kemudian membuat objek dari kelas tersebut dengan kata kunci new. Berikut adalah contoh sederhana:
2	Buatlah folder oop dalam folder dasarWeb/ dengan file baru yaitu oop.php.

3	Ketikkan ke dalam file oop.php tersebut kode di bawah ini.
---	--

4	<pre>jobsheet12 &gt; oop &gt; oop.php &gt; ... 1  &lt;?php 2  2 references   0 implementations 3  class Car 4  { 5      3 references 6      public \$brand; 7 8      1 reference   0 overrides 9      public function startEngine() 10     { 11         echo "Engine started!"; 12     } 13 } 14 15 \$car1 = new Car(); 16 \$car1-&gt;brand = "Toyota"; 17 18 \$car2 = new Car(); 19 \$car2-&gt;brand = "Honda"; 20 21 \$car1-&gt;startEngine(); 22 echo \$car2-&gt;brand;</pre>
---	--

5	<p>Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.1)</p> <p>Berdasarkan pemahaman saya, kode di atas digunakan untuk membuat sebuah class Car. Kelas tersebut memiliki properti brand yang dapat diakses dari luar kelas, dan memiliki metode startEngine() yang mencetak output "Engine started!" saat program ini dijalankan. Selanjutnya, objek \$car1 dan \$car2 dibuat dari class Car, masing-masing dengan mereferensikan merek kendaraan yang berbeda. Akhirnya, method startEngine() dipanggil pada objek \$car1, sementara brand dicetak pada \$car2. Berikut merupakan output yang dihasilkan:</p>  <p>Engine started!Honda</p>
---	--

6	<p>Inheritance adalah salah satu konsep dasar dalam pemrograman berorientasi objek (OOP) yang memungkinkan sebuah class untuk mewarisi properti dan metode dari class lain. Class yang mewarisi disebut subclass atau child class, sedangkan class yang memberikan warisan disebut superclass atau parent class. Konsep ini memungkinkan kita untuk menggunakan kembali kode, memperpanjang fungsionalitas, dan membangun hierarki class.</p> <p>Berikut ini adalah contoh sederhana penerapan inheritance dalam PHP:</p>
---	---

```

jobsheet12 > oop > animalinheritance.php > ...
1  <?php
2  2 references | 2 implementations
3  class Animal
4  {
5      5 references
6      protected $name;
7      0 references | 0 overrides
8      public function __construct($name){
9          $this->name = $name;
10     }
11
12     1 reference | 0 overrides
13     public function eat(){
14         echo $this->name . " is eating.<br>";
15     }
16
17     1 reference | 0 overrides
18     public function sleep(){
19         echo $this->name . " is sleeping.<br>";
20     }
21 }
22
23 1 reference | 0 implementations
24 class Cat extends Animal{
25     1 reference | 0 overrides
26     public function meow(){
27         echo $this->name . "says meow!<br>";
28     }
29 }
30
31 1 reference | 0 implementations
32 class Dog extends Animal{
33     1 reference | 0 overrides
34     public function bark(){
35         echo $this->name . " says woof!<br>";
36     }
37 }
38
39 $cat = new Cat("Whiskers");
40 $dog = new Dog("Buddy");
41
42 $cat->eat();
43 $dog->sleep();
44
45 $cat->meow();
46 $dog->bark();
47
48 ?>

```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.2)

Berdasarkan pemahaman saya, kode di atas menunjukkan bahwa terdapat 2 kelas, yaitu class parent `Animal` dan dua childclass yaitu class `Cat` dan `Dog`. Class `Animal` memiliki properti proteksi `name` dan method public untuk `eat` dan `sleep`, sementara class `Cat` dan `Dog` mewarisi sifat dan perilaku dari class `Animal`, serta memiliki metode tambahan masing-masing: `meow` untuk `Cat` dan `bark` untuk `Dog`. Jadi dapat disimpulkan bahwa objek dibuat dari kelas tersebut dan metode dipanggil untuk menunjukkan perilaku masing-masing. Berikut merupakan output yang dihasilkan:

← → ↺ ⓘ localhost:8080/dasarWeb/jobsheet12/animalinheritance.php

Whiskers is eating.  
Buddy is sleeping.  
Whiskers says meow!  
Buddy says woof!

6

Polymorphism adalah konsep dalam pemrograman berorientasi objek yang memungkinkan objek dari class yang berbeda untuk merespon pada pemanggilan metode dengan cara yang sama. Ini dapat diwujudkan dalam PHP melalui penggunaan antarmuka (interface) dan penggunaan overriding metode. Dengan polymorphism, Anda dapat memperlakukan objek dari class yang berbeda dengan cara yang seragam.

Berikut adalah contoh sederhana penggunaan polymorphism dalam PHP menggunakan antarmuka:

```

1 <?php
2 interface Shape
3 {
4     1 reference | 2 overrides
5     public function calculateArea();
6 }
7
8 1 reference | 0 implementations
9 class Circle implements Shape
10 {
11     2 references
12     private $radius;
13
14     1 reference | 0 overrides
15     public function __construct($radius)
16     {
17         $this->radius = $radius;
18     }
19
20     1 reference | 0 overrides
21     public function calculateArea()
22     {
23         return pi() * pow($this->radius, 2);
24     }
25 }
26
27 1 reference | 0 implementations
28 class Rectangle implements Shape
29 {
30     2 references
31     private $width;
32     2 references
33     private $height;
34
35     1 reference | 0 overrides
36     public function __construct($width, $height)
37     {
38         $this->width = $width;
39         $this->height = $height;
40     }
41
42     1 reference | 0 overrides
43     public function calculateArea()
44     {
45         return $this->width * $this->height;
46     }
47 }
48
49 2 references
50 function printArea(Shape $shape)
51 {
52     echo "Area: " . $shape->calculateArea() . "<br>";
53 }
54
55 $circle = new Circle(5);
56 $rectangle = new Rectangle(4, 6);
57
58 printArea($circle);
59 printArea($rectangle);
60
61

```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.3)  
 Berdasarkan pemahaman saya, pada kode di atas terdapat 2 class yang mengimplementasikan sebuah antarmuka (interface) yang mana class `Shape` mengimplementasikan ke class `Circle` dan `Rectangle`. Interface class `Shape` mendefinisikan sebuah method `calculateArea()`, yang harus diimplementasikan oleh semua kelas yang mengimplementasikannya. Class `Circle` mewakili lingkaran dengan properti radius, sedangkan class `Rectangle` mewakili persegi panjang dengan properti lebar dan tinggi. Kedua kelas tersebut mengimplementasikan method `calculateArea()` yang sesuai dengan rumus dari masing-masing bangun datar yaitu luas lingkaran dan luas persegi panjang. Fungsi `printArea()` menerima objek dari class `Shape` dan mencetak luasnya dengan memanggil method `calculateArea()`. Maka alur kerja program yaitu, ketika objek `Circle` dan `Rectangle` dibuat, maka method `printArea()` dipanggil untuk mencetak luas masing-masing bentuk. Yang mana hasil output nya seperti di bawah ini:

← → ↻ ⓘ localhost:8080/dasarWeb/jobsheet12/polymorphism.php

Area: 78.539816339745  
 Area: 24

Encapsulation adalah salah satu konsep dalam pemrograman berorientasi objek (OOP) yang mengizinkan pembungkusan (encapsulation) properti dan metode dalam sebuah class sehingga akses ke mereka dapat dikontrol. Hal ini dapat membantu dalam menerapkan prinsip-prinsip pengelolaan akses dan memastikan bahwa properti dan metode yang mungkin berubah di kemudian hari tidak merusak integritas class atau program secara keseluruhan.

Berikut adalah contoh sederhana encapsulation dalam PHP:

```
jobsheet12 > oop > encapsulation.php > PHP Intelephense > Car > getColor
1  <?php
   3 references | 0 implementations
2  class Car
3  {
   2 references
4      private $model;
   3 references
5      private $color;
6
   3 references | 0 overrides
7      public function __construct($model,$color)
8      {
9          $this->model = $model;
10         $this->color = $color;
11     }
12
   1 reference | 0 overrides
13     public function getModel()
14     {
15         return $this-> model;
16     }
17
   1 reference | 0 overrides
18     public function setColor($color)
19     {
20         $this->color=$color;
21     }
22
   2 references | 0 overrides
23     public function getColor()
24     {
25         return $this->color;
26     }
27 }
28
29 $car = new Car("Toyota","Blue");
30
31 echo "Model: " . $car->getModel() . "<br>";
32 echo "Color: " . $car->getColor() . "<br>";
33
34 $car-> setColor("Red");
35 echo "Updated Color: " . $car->getColor() . "<br>";
36
37
```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.4)

Berdasarkan pemahaman saya, kode di atas digunakan untuk membuat sebuah class `Car` yang mana memiliki 2 properti privat, yaitu `\$model` dan `\$color` yang hanya bisa diakses dari dalam class tersebut. Ada juga konstruktor yang digunakan untuk menginisialisasi objek `Car` dengan nilai model dan warna yang diberikan. Selain itu, terdapat method `getModel()` dan `getColor()` untuk mengambil nilai properti `\$model` dan `\$color`, dan method `setColor()` untuk mengubah nilai properti `\$color`. Hasil akhirnya, sebuah objek `Car` diciptakan dengan model "Toyota" dan warna "Blue", kemudian warnanya diubah menjadi "Red" dan hasilnya ditampilkan. Berikut merupakan output yang dihasilkan:

← → ↺ ⓘ localhost:8080/dasarWeb/jobsheet12/encapsulation.php

Model: Toyota  
Color: Blue  
Updated Color: Red

Abstraction adalah salah satu konsep dasar dalam pemrograman berorientasi objek (OOP) yang memungkinkan Anda menyembunyikan detail internal dan hanya mengekspos fungsionalitas yang diperlukan. Ini membantu dalam menciptakan class dan metode yang bersifat umum dan

fleksibel, memungkinkan pengguna untuk berinteraksi dengan objek tanpa perlu mengetahui implementasi internalnya.

Berikut adalah contoh sederhana abstraksi dalam PHP menggunakan abstract class dan method:

```
jobsheet12 > oop > abstraksi.php > ...
1  <?php
2  5 references | 4 implementations
3  abstract class Shape
4  {
5      3 references | 4 overrides
6      abstract public function calculateArea();
7  }
8
9  2 references | 0 implementations
10 class Circle extends Shape
11 {
12     4 references
13     private $radius;
14     2 references | 0 overrides
15     public function __construct($radius){
16         $this->radius=$radius;
17     }
18
19     3 references | 0 overrides | prototype
20     public function calculateArea()
21     {
22         return pi() * pow($this-> radius, 2);
23     }
24 }
25
26 2 references | 0 implementations
27 class Rectangle extends
28 {
29     4 references
30     private $width;
31     4 references
32     private $height;
33
34     2 references | 0 overrides
35     public function __construct($width, $height)
36     {
37         $this->width=$width;
38         $this->height=$height;
39     }
40
41     3 references | 0 overrides | prototype
42     public function calculateArea()
43     {
44         return $this->width * $this->height;
45     }
46 }
47
48 $circle = new Circle(5);
49 $rectangle = new Rectangle(4,6);
50
51 echo "Area of Circle: " . $circle->calculateArea() . "<br>";
52 echo "Area of Rectangle: " . $rectangle->calculateArea() . "<br>";
53 >>
```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.5)  
Berdasarkan pemahaman saya, kode di atas merupakan pengimplementasian pewarisan (inheritance) dan abstraksi dalam pemrograman berorientasi objek menggunakan bahasa pemrograman PHP. Yang mana pada class tersebut terdapat class abstrak `Shape` yang memiliki 2 class turunan yaitu `Circle` dan `Rectangle`. Class abstrak `Shape` memiliki method abstrak `calculateArea()` yang menetapkan kerangka dasar untuk menghitung luas bentuk. Class `Circle` memiliki properti privat `\$radius` yang merepresentasikan jari-jari lingkaran. Konstruktor menerima jari-jari sebagai argumen. Method `calculateArea()` diimplementasikan untuk menghitung luas lingkaran. Class `Rectangle` memiliki properti privat `\$width` dan `\$height` yang merepresentasikan lebar dan tinggi persegi panjang. Konstruktor menerima lebar dan tinggi sebagai argumen. Method `calculateArea()` diimplementasikan untuk menghitung luas persegi panjang. Objek dibuat untuk masing-masing kelas (`\$circle` dan `\$rectangle`) dengan memberikan nilai-nilai yang diperlukan. Hasil luas lingkaran dan persegi panjang kemudian dicetak menggunakan metode `echo`. Berikut merupakan output yang dihasilkan:

← → ↺ ⓘ localhost:8080/dasarWeb/jobsheet12/abstraksi.php

Area of Circle: 78.539816339745  
Area of Rectangle: 24

9

Interface adalah konsep dalam pemrograman berorientasi objek yang memungkinkan definisi kontrak atau kerangka yang harus diikuti oleh class-class yang mengimplementasikannya. Interface tidak memiliki implementasi sendiri, tetapi hanya menyediakan deklarasi metode dan properti yang harus diimplementasikan oleh class yang menggunakannya. Hal ini memungkinkan untuk mencapai polimorfisme tanpa memerlukan pewarisan tunggal, sehingga sebuah class dapat mengimplementasikan beberapa interface.

Berikut adalah contoh penggunaan interface dalam PHP:

```
jobsheet12 > oop > interface.php > ...
1  <?php
   6 references | 5 implementations
2  interface Shape
3  {
   4 references | 5 overrides
4  |     public function calculateArea();
5  }
6
   1 reference | 3 implementations
7  interface Color
8  {
   1 reference | 1 override
9  |     public function getColor();
10 }
11
   3 references | 0 implementations
12 class Circle implements Shape, Color
13 {
   6 references
   2 references
14 |     private $radius;
15 |     private $color;
16
   3 references | 0 overrides
17 |     public function __construct($radius, $color){
18 |         $this->radius=$radius;
19 |         $this->color=$color;
20 |     }
21
   4 references | 0 overrides
22 |     public function calculateArea()
23 |     {
24 |         return pi() * pow($this->radius, 2);
25 |     }
26
   1 reference | 0 overrides
27 |     public function getColor(){
28 |         return $this->color;
29 |     }
30 }
31 $circle = new Circle(5, "Blue");
32
33 echo "Area of circle: " . $circle->calculateArea() . "<br>";
34 echo "Color of circle: " . $circle->getColor() . "<br>";
35
```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.6)

Berdasarkan pemahaman saya, kode di atas terdapat dua interface yaitu `Shape` dan `Color` yang mendefinisikan method-method yang harus diimplementasikan oleh class yang menggunakan interface tersebut. Kemudian, ada class `Circle` yang mengimplementasikan kedua interface tersebut. Class ini memiliki properti radius dan color, serta method `calculateArea()` untuk menghitung luas lingkaran dan method `getColor()` untuk mendapatkan warna lingkaran. Jadi alur kerja program, objek `\$circle` yang dibuat menggunakan class `Circle` dengan radius 5 dan warna "Blue", dan kemudian ditampilkan luas dan warna lingkaran tersebut. Berikut merupakan output yang dihasilkan:

← → ↺ ⓘ localhost:8080/dasarWeb/jobsheet12/interface.php

Area of circle: 78.539816339745

Color of circle: Blue

10

Constructors dan destructors adalah metode khusus dalam pemrograman berorientasi objek (OOP) yang digunakan dalam PHP untuk menginisialisasi dan membersihkan objek. Constructor adalah metode yang dipanggil secara otomatis ketika objek baru dibuat, sedangkan destructor adalah metode yang dipanggil secara otomatis ketika objek dihapus atau tidak lagi digunakan.

Constructor (Metode Pembuat)

Constructor menggunakan nama khusus `__construct` dalam PHP. Constructor ini akan dipanggil secara otomatis setiap kali objek baru dibuat dari class yang mengandung constructor tersebut.

Destructor (Metode Penghancur)

Destructor menggunakan nama khusus `__destruct` dalam PHP. Destructor ini akan dipanggil secara otomatis ketika objek dihapus atau program selesai dieksekusi.

Berikut adalah contoh constructor dan destructor:

```
jobsheet12 > oop > constructor&destructor.php > ...
1  <?php
   4 references | 0 implementations
2  class Car{
   5 references
3      private $brand;
4
   4 references | 0 overrides
5      public function __construct($brand){
6          echo "A new car is created.<br>";
7          $this->brand=$brand;
8      }
9
   1 reference | 0 overrides
10     public function getBrand(){
11         return $this->brand;
12     }
13
   0 references | 0 overrides
14     public function __destruct(){
15         echo "The car is destroyed.<br>";
16     }
17 }
18
19 $car = new Car("Toyota");
20 echo "Brand : " . $car->getBrand() . "<br>";
```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.7)

Berdasarkan pemahaman saya, kode di atas membuat class `Car`. Kelas ini memiliki properti `brand` yang bersifat private dan method konstruktor `\_\_construct()` untuk menginisialisasi objek. Sedangkan, method `getBrand()` digunakan untuk mengambil nilai dari properti `brand`. Maka, alur kerja program di atas yaitu ketika objek `Car` dibuat dengan menggunakan constructor (`new Car("Toyota")`), pesan "A new car is created." akan ditampilkan. Kemudian, method `getBrand()` dipanggil untuk mengambil merek mobil, yang dalam kasus ini adalah "Toyota". Setelah itu, karena tidak ada referensi lagi yang menunjuk pada objek tersebut, maka method destructor `\_\_destruct()` akan dipanggil dan pesan "The car is destroyed." akan ditampilkan.

Berikut merupakan output yang dihasilkan:

← → ↺ ⓘ localhost:8080/dasarWeb/jobsheet12/constructor&destructor.php

A new car is created.  
Brand : Toyota  
The car is destroyed.



## Encapsulation and Access Modifiers

Encapsulation adalah salah satu konsep utama dalam pemrograman berorientasi objek (OOP), dan itu melibatkan pembungkusan data (variabel) dan metode (fungsi) dalam sebuah class. Ini membantu dalam menyembunyikan implementasi internal suatu class dan hanya mengekspos fungsionalitas yang diperlukan. Access modifiers adalah bagian dari encapsulation yang memungkinkan Anda mengontrol tingkat akses ke properti dan metode dalam sebuah class.

PHP memiliki tiga access modifiers utama yang dapat digunakan dalam class:

**Public (public):** Properti atau metode yang dideklarasikan sebagai public dapat diakses dari luar class, sehingga mereka bersifat terbuka untuk diakses dari mana saja.

**Protected (protected):** Properti atau metode yang dideklarasikan sebagai protected hanya dapat diakses dari dalam class itu sendiri dan dari class turunannya (inheritance).

**Private (private):** Properti atau metode yang dideklarasikan sebagai private hanya dapat diakses dari dalam class itu sendiri. Mereka tidak dapat diakses dari luar class, bahkan oleh class turunannya.

Berikut adalah contoh penggunaan access modifiers dalam PHP:

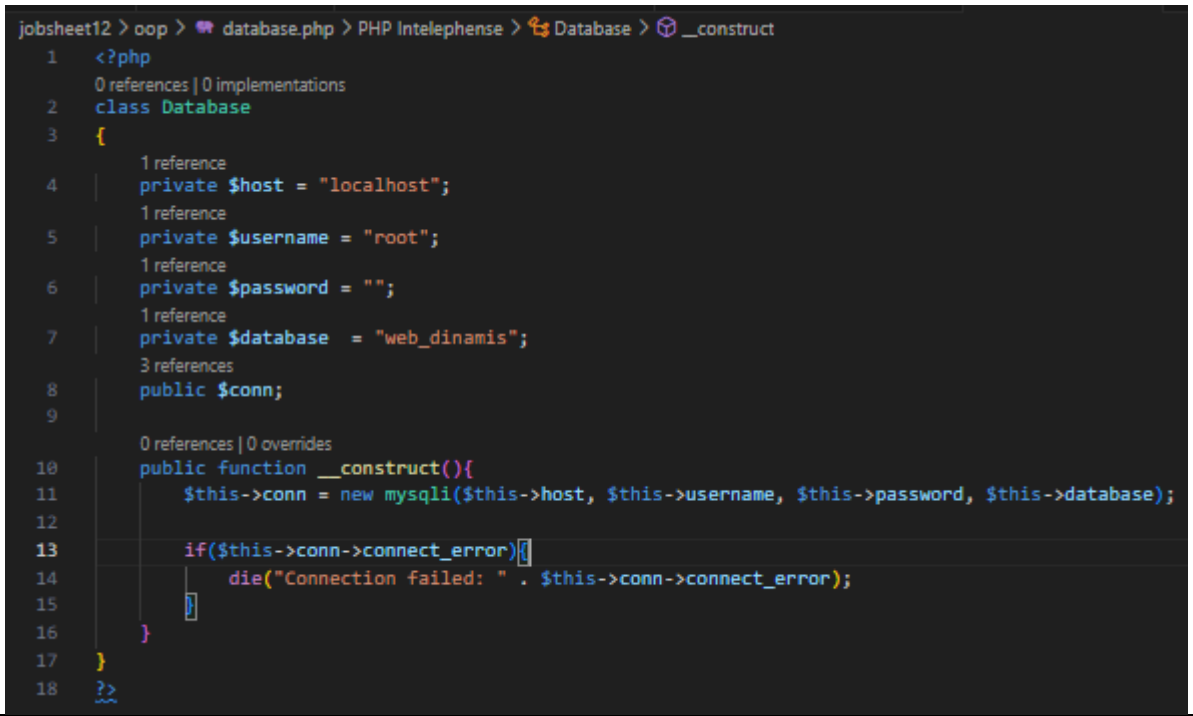
```
1 <?php
2 class Animal
3 {
4     public $name;
5     protected $age;
6     private $color;
7
8     public function __construct($name, $age, $color)
9     {
10         $this->name=$name;
11         $this->age=$age;
12         $this->color=$color;
13     }
14
15     public function getName()
16     {
17         return $this->name;
18     }
19
20     public function getAge()
21     {
22         return $this->age;
23     }
24
25     public function getColor()
26     {
27         return $this->color;
28     }
29 }
30
31 $animal = new Animal("Dog",3,"Brown");
32 echo "Name : " . $animal->getName() . "<br>";
33 echo "Age : " . $animal->getAge() . "<br>";
34 echo "Color : " . $animal->getColor() . "<br>";
35
```

Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.8)

Berdasarkan pemahaman saya, kode di atas terjadi implementasi dari class `Animal` yang memiliki tiga properti: `\$name`, `\$age`, dan `\$color`, yang masing-masing memiliki tingkat akses yang berbeda. Properti `\$name` memiliki akses publik, properti `\$age` memiliki akses dilindungi (protected), dan properti `\$color` memiliki akses privat (private). Class `Animal` memiliki constructor `\_\_construct()` yang digunakan untuk menginisialisasi nilai properti saat objek dibuat. Selain itu, class ini memiliki tiga method publik: `getName()`, `getAge()`, dan `getColor()` yang masing-masing mengembalikan nilai dari properti yang sesuai. Pada bagian akhir, sebuah objek `Animal` dibuat dengan nilai "Dog" untuk nama, 3 untuk usia, dan "Brown" untuk warna. Kemudian, nilai-nilai tersebut ditampilkan menggunakan metode-metode yang telah didefinisikan. Berikut merupakan output yang dihasilkan:

```
localhost:8080/dasarWeb/jobsheet12/encapsulation&access.php
Name : Dog
Age : 3
Color : Brown
```

## Praktikum 2. CRUD dengan OOP

Langkah	Keterangan
1	Buat file baru pada folder oop dengan nama baru bernama database.php. Ketikkan kode seperti di bawah ini.
2	 <pre>jobsheet12 &gt; oop &gt; database.php &gt; PHP Intelephense &gt; Database &gt; __construct 1  &lt;?php    0 references   0 implementations 2  class Database 3  {    1 reference 4      private \$host = "localhost";    1 reference 5      private \$username = "root";    1 reference 6      private \$password = "";    1 reference 7      private \$database = "web_dinamis";    3 references 8      public \$conn; 9    0 references   0 overrides 10 public function __construct(){ 11     \$this-&gt;conn = new mysqli(\$this-&gt;host, \$this-&gt;username, \$this-&gt;password, \$this-&gt;database); 12 13     if(\$this-&gt;conn-&gt;connect_error){ 14         die("Connection failed: " . \$this-&gt;conn-&gt;connect_error); 15     } 16 } 17 } 18 ?&gt;</pre>
3	Buat file baru pada folder oop dengan nama baru bernama crud.php. Ketikkan kode seperti di bawah ini.

4

```

jobsheet12 > oop > crud.php > PHP Intelephense > Crud > read
1  <?php
2  require_once 'Database.php';
   2 references | 0 implementations
3  class Crud
4  {
   6 references
5      private $db;
   2 references | 0 overrides
6      public function __construct()
7      {
8          $this->db = new Database();
9      }
10
11     // Create
   1 reference | 0 overrides
12     public function create($jabatan, $keterangan)
13     {
14         $query = "INSERT INTO jabatan (jabatan, keterangan) VALUES ('$jabatan', '$keterangan')";
15         $result = $this->db->conn->query($query);
16         return $result;
17     }
18
19     // Read
   1 reference | 0 overrides
20     public function read()
21     {
22         $query = "SELECT * FROM jabatan";
23         $result = $this->db->conn->query($query);
24
25         $data = [];
26         if ($result->num_rows > 0) {
27             while ($row = $result->fetch_assoc()) {
28                 $data[] = $row;
29             }
30         }
31         return $data;
32     }
33
34     // Read By Id
   1 reference | 0 overrides
35     public function readById($id)
36     {
37         $query = "SELECT * FROM jabatan WHERE id = $id";
38         $result = $this->db->conn->query($query);
39         if ($result->num_rows == 1) {
40             return $result->fetch_assoc();
41         } else {
42             return null;
43         }
44     }
45
46     // Update
   1 reference | 0 overrides
47     public function update($id, $jabatan, $keterangan)
48     {
49         $query = "UPDATE jabatan SET jabatan = '$jabatan', keterangan = '$keterangan' WHERE id = $id";
50         $result = $this->db->conn->query($query);
51
52         return $result;
53     }
54
55     // Delete
   1 reference | 0 overrides
56     public function delete($id)
57     {
58         $query = "DELETE FROM jabatan WHERE id = $id";
59         $result = $this->db->conn->query($query);
60     }
61 }

```

5

Buat file baru pada folder oop dengan nama baru bernama index.php. Ketikkan kode seperti di bawah ini.

```

jobsheet12 > oop > index.php > ...
1  <?php
2  require_once 'Crud.php';
3
4  $crud = new Crud();
5  if ($_SERVER['REQUEST_METHOD'] === 'POST') {
6      $jabatan = $_POST['jabatan'];
7      $keterangan = $_POST['keterangan'];
8      $crud->create($jabatan, $keterangan);
9  }
10
11 if (isset($_GET['action']) && $_GET['action'] === 'delete') {
12     $id = $_GET['id'];
13     $crud->delete($id);
14 }
15
16 $stampil = $crud->read();
17
18 ?>
19
20 <!DOCTYPE html>
21 <html lang="en">
22
23 <head>
24     <meta charset="UTF-8">
25     <meta name="viewport" content="width=device-width, initial-scale=1.0">
26     <title>CRUD Jabatan</title>
27     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
28 </head>
29
30 <body>
31     <div class="container mt-5">
32         <button type="button" class="btn btn-success mb-3" data-toggle="modal" data-target="#tambahModal">Tambah</button>
33         <table class="table">
34             <thead>
35                 <tr>
36                     <th>ID</th>
37                     <th>Jabatan</th>
38                     <th>Keterangan</th>
39                     <th>Aksi</th>
40                 </tr>
41             </thead>
42             <tbody>
43                 <?php
44                 foreach ($stampil as $show) {
45                     <tr>
46                         <td> . $show['id'] . "</td>";
47                         <td> . $show['jabatan'] . "</td>";
48                         <td> . $show['keterangan'] . "</td>";
49                         <td>
50                             <a href="edit.php?id=" . $show['id'] . '" class="btn btn-primary btn-sm">Edit</a>;
51                             <a href="index.php?action=delete&id=" . $show['id'] . '" class="btn btn-danger btn-sm">Delete</a>;
52                         </td>
53                     </tr>
54                 }
55                 ?>
56             </tbody>
57         </table>
58     </div>
59     <div class="modal fade" id="tambahModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
60         <div class="modal-dialog" role="document">
61             <div class="modal-content">
62                 <div class="modal-header">
63                     <h5 class="modal-title" id="exampleModalLabel">Tambah Data Jabatan</h5>
64                     <button type="button" class="close" data-dismiss="modal" aria-label="Close">
65                         <span aria-hidden="true">&times;</span>
66                     </button>
67                 </div>
68                 <div class="modal-body">
69                     <form action="" method="post">
70                         <div class="form-group">
71                             <label for="name">Jabatan:</label>
72                             <input type="text" name="jabatan" id="jabatan" class="form-control" required>
73                         </div>
74                         <div class="form-group">
75                             <label for="email">Keterangan:</label>
76                             <textarea name="keterangan" id="keterangan" cols="30" rows="10" class="form-control" required></textarea>
77                         </div>
78                         <button type="submit" class="btn btn-primary">Tambah</button>
79                     </form>
80                 </div>
81             </div>
82         </div>
83     </div>
84     <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
85     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
86 </body>
87
88 </html>

```

7

Buat file baru pada folder oop dengan nama baru bernama edit.php. Ketikkan kode seperti di bawah ini.

8

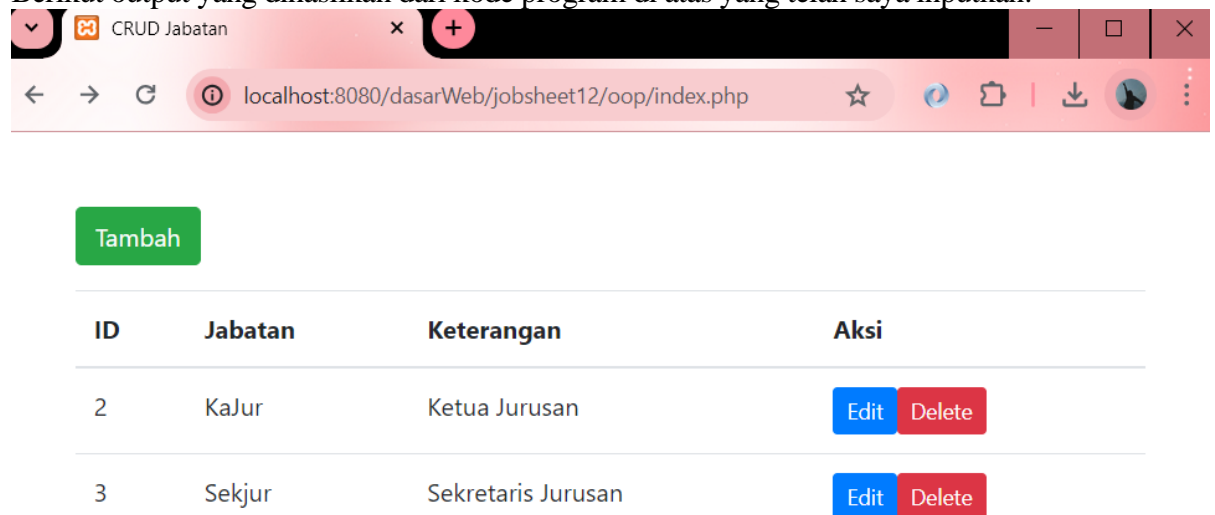
```

jobsheet12 > oop > edit.php > ...
1  <?php
2  require_once 'Crud.php';
3
4  $crud = new Crud();
5  $id = $_GET['id'];
6  $stampil = $crud->readById($id);
7
8  if ($_SERVER['REQUEST_METHOD'] === 'POST') {
9      $jabatan = $_POST['jabatan'];
10     $keterangan = $_POST['keterangan'];
11
12     $crud->update($id, $jabatan, $keterangan);
13
14     header("Location: index.php");
15     exit();
16 }
17 >>
18
19 <!DOCTYPE html>
20 <html lang="en">
21
22 <head>
23     <meta charset="UTF-8">
24     <meta name="viewport" content="width=device-width, initial-scale=1.0">
25     <title>Edit Jabatan</title>
26     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
27 </head>
28
29 <body>
30     <div class="container mt-5">
31         <h2>Edit Jabatan</h2>
32         <form action="" method="post">
33             <div class="form-group">
34                 <label for="jabatan">Jabatan:</label>
35                 <input type="text" name="jabatan" id="jabatan" class="form-control" value="<?php echo $stampil['jabatan']; ?>" required>
36             </div>
37             <div class="form-group">
38                 <label for="keterangan">Keterangan:</label>
39                 <textarea name="keterangan" id="keterangan" cols="30" rows="10" class="form-control" required><?php echo $stampil['keterangan']; ?></textarea>
40             </div>
41             <input type="hidden" name="id" value="<?php echo $stampil['id']; ?>">
42             <button type="submit" class="btn btn-primary">Update</button>
43         </form>
44     </div>
45     <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
46     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
47 </body>
48
49 </html>

```

Jalankan code pada praktikum 2. Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 2.1)

Berikut output yang dihasilkan dari kode program di atas yang telah saya inputkan:



ID	Jabatan	Keterangan	Aksi
2	KaJur	Ketua Jurusan	<a href="#">Edit</a> <a href="#">Delete</a>
3	Sekjur	Sekretaris Jurusan	<a href="#">Edit</a> <a href="#">Delete</a>

Edit Jabatan

localhost:8080/dasarWeb/jobsheet12/oop/edit.php?id=3

## Edit Jabatan

Jabatan:

Keterangan:

Sekretariat Jurusan

Update

localhost:8080/dasarWeb/jobsheet12/oop/index.php

Tambah

ID	Jabatan	Keterangan	Aksi
2	KaJur	Ketua Jurusan	<div>EditDelete</div>
3	SekJur	Sekretariat Jurusan	<div>EditDelete</div>

Berdasarkan pemahaman saya dari hasil output di atas, kode di atas terdapat pengimplementasian CRUD (Create, Read, Update, Delete) sederhana dalam PHP dengan menggunakan OOP (Object-Oriented Programming). Pada file database.php berisi definisi dari class `database` yang digunakan untuk membuat koneksi ke database MySQL menggunakan objek `mysqli`. Koneksi ke database diinisialisasi saat objek `Database` dibuat. Sedangkan, pada file crud.php berisi definisi class `Crud` yang digunakan untuk melakukan operasi CRUD pada tabel `jabatan` dalam database. Class ini memiliki beberapa method seperti `create`, `read`, `update`, dan `delete` untuk masing-masing operasi CRUD. Sementara file index.php merupakan halaman utama yang menampilkan daftar jabatan. Di sini, objek `Crud` dibuat untuk mengakses method CRUD yang didefinisikan dalam class `Crud`. Pada file edit.php digunakan untuk mengedit data jabatan yang mana ketika halaman ini dimuat, data

	jabatan yang akan diedit diambil dari database menggunakan method <code>`readById`</code> dari objek <code>`Crud`</code> . Setelah pengguna mengirimkan formulir edit, data yang diubah disimpan menggunakan method <code>`update`</code> dari objek <code>`Crud`</code> . Setelah formulir diisi dan disubmit, data akan diperbarui dalam database, dan pengguna akan diarahkan kembali ke <code>`index.php`</code> untuk melihat perubahan yang diterapkan.
--	---