LAPORAN PRAKTIKUM

PEMROGRAMAN WEB

oleh:

Triyono Rifan [20081010003]

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"

JAWA TIMUR

2022

1. Objek Oriented Programing (OOP) PHP

1.1. Tujuan Praktikum

Tujuan dari praktikum ini adalah untuk mengetahui pemrograman yang bebasis objek dan tentunya sudah mulai relevan dengan kehidupan sesungguhnya. OOP membantu untuk memudahkan programmer dalam mengelompokkan objek dan tentu lebih mudah dalam pemrosesannya. Dan juga dapat mengetahui cara penggunaan Class, Objek, Method pada OOP PHP.

1.2. Dasar Teori

Internet sangat berkembang dengan pesat dan penyebaran informasi dapat dilakukan melalui internet seperti menggunakan blog atau website yang sangat dengan mudah dibuka dalam telephone seluler. Seseorang dapat dengan mudah menyebarkan informasi melalui jaringan internet apabila telah memahami cara pembuatan blog atau website.

Pada sisi pembuatan website terdapat dua sisi yakni bagian user dan di balik layer user atau biasa disebut dengan frontend dan backend. Pada pengembangan website, PHP adalah salah satu Bahasa pemrograman yang mampu membuat website menjadi dinamis. PHP (Hypertext Preprocessor) merupakan Bahasa server side scripting yang tentu bersifat open source. Jenis dari server yang sering digunakan dengan PHP biasanya Apache, Nginx, dan LiteSpeed.

Fungsi dari Bahasa PHP sendiri yakni untuk pengembangan website baik statis yang tidak membutuhkan banyak fitu ataupun website dinamis seperti took online dan fiturnya yang banyak.

Menurut sejarahnya, PHP pertama kali muncul tahun 1994 diciptakan oleh Dr Leonardo Bernart. Awalnya PHP memiliki singkatan “Personal Home Page Tools”, selanjutnya PHP diganti nama menjadi FI (Form Interpreter). Sejak kemunculan PHP versi 3.0, nama PHP kembali lagi digunakan dengan singkatan menjadi “Hypertext Preprocessor” hingga sekarang ini.

Object Oriented Programming (OOP) telah menjadi standar dalam dunia pemrograman, termasuk PHP. Walaupun kita bisa membuat program PHP tanpa menggunakan OOP sama sekali, namun untuk membuat aplikasi ‘real world’ yang fleksibel, programmer PHP akan beralih menggunakan OOP.

Pemrograman berorientasi objek memiliki beberapa keunggulan dibandingkan pemrograman prosedural:

- OOP lebih cepat dan lebih mudah untuk dieksekusi
- OOP menyediakan struktur yang jelas untuk program
- OOP membantu menjaga kode PHP KERING "Jangan Ulangi Sendiri", dan membuat kode lebih mudah untuk dipelihara, dimodifikasi, dan di-debug
- OOP memungkinkan untuk membuat aplikasi penuh yang dapat digunakan kembali dengan lebih sedikit kode dan waktu pengembangan yang lebih singkat

1.3. SOAL dan JAWABAN

1. Implementasi pembuatan dan pengaksesan Class
2. Implementasi pembuatan dan pengaksesan Objek
3. Implementasi Enkapsulasi Objek (Public, Protected dan Private)
4. Implementasi Variabel $this dalam Pemrograman Objek
5. Penggunaan Pseudo-Variable $this dalam Objek
6. Cara Membuat Method dalam Pemrograman Objek
7. Implementasi Constructor dan Destructor
8. Implementasi Inheritance (Pewarisan)
9. Cara Mengakses Property dan Method Parent Class
10. Implementasi Constructor dan Destructor padaa Parent Class
11. implementasi Static Property dan Static Method
12. Implementasi Konstanta Class dalam Pemrograman Objek
13. Implementasi Final Method dan Final Class
14. Implementasi Abstract Class dan Abstract Method PHP
15. Implementasi Object Interface
16. implementasi Polimorfisme dalam Pemrograman Objek

2. PENULISAN KODE

Soal 1

```php
<?php
// Membuat Class
class Produk
{
    var $judul, $penerbit, $pengarang, $tahun_terbit;

    function setJudul($judul)
    {
        $this->judul = $judul;
    }
```

```
    function getData()
    {
        return $this->judul;
    }
}

// mengakses Class untuk Objek1
$objek1 = new Produk();
$objek1->setJudul("Pemrograman Objek");
```

Pada potongan kode di atas adalah untuk membuat dan mengakses sebuah class pada OOP PHP dengan memberikan : class NamaClass{} kita sudah menginisialisasi suatu class. Ketika ingin mengakses sebuah class maka harus ditambahkan keyword new untuk memanggil class.

Soal 2

```
<!-- b. Implementasi pembuatan dan pengaksesan Objek -->
<?php
class Produk
{
    public $judul,
        $penerbit,
        $pengarang,
        $tahunTerbit;

    public function setData($judul = "null", $pengarang = "Anonim", $penerbit
= "Anonim", $tahunTerbit = "1999")
    {
        $this->judul = $judul;
        $this->penerbit = $penerbit;
        $this->pengarang = $pengarang;
        $this->tahunTerbit = $tahunTerbit;
    }

    public function getData()
    {
        return "Judul : {$this->judul} , Pengarang: {$this->pengarang},
Penerbit:  {$this->penerbit}, Tahun Terbit : {$this->tahunTerbit}";
    }
}
// Pebuatan objek class
$produk1 = new Produk();
$produk2 = new Produk();

?>
```

Pada potongan kode di atas adalah cara pembuatan objek pada class produk. Pada class produk memiliki beberapa fungsi untuk nantinya dapat digunakan oleh objek yang menggunakan class produk

Soal 3

Enkapsulasi (***encapsulation***) adalah *sebuah metoda untuk mengatur struktur class dengan cara menyembunyikan alur kerja dari class tersebut*.

Struktur class yang dimaksud adalah **property** dan **method**. Dengan *enkapsulasi*, kita bisa membuat pembatasan akses kepada *property* dan *method*, sehingga hanya *property* dan *method* tertentu saja yang bisa diakses dari luar class. *Enkapsulasi* juga dikenal dengan istilah '***information hiding'***.

```php
<?php

class Buku
{
    public $judul, $penulis;

    public function setData($judul = "null", $penulis = "Anonim")
    {
        $this->judul = $judul;
        $this->penulis = $penulis;
    }

    public function getData()
    {
        return "Judul = {$this->judul} , Penulis = {$this->penulis}";
    }
}

$buku1 = new Buku();
$buku1->setData("Pemrograman Flutter", "Budi Raharjo");

echo "<b>Public:</b>";
echo "<br>";
echo $buku1->getData();
echo "<br><br>";
?>
```

Pada potongan kode di atas adalah property dan method yang dinyatakan dalam public. Maka seluruh program diluar class tersebut dapat mengaksesnya.

```php
<!-- protected -->
<?php
class Handphone
{
    protected $Pemilik = "Riyo";

    public function getPemilik()
    {
```

```
        return $this->Pemilik;
    }

    protected function getPIN()
    {
        return "PIN = 123211";
    }

    public function aksesPin()
    {
        return $this->getPIN();
    }
}

$hp1 = new Handphone();

echo $hp1->getPemilik();
echo "<br>";
echo $hp1->aksesPin();
echo "<br><br>";
?>
```

Pada potongan kode di atas adalah property atau method yang dinyatakan sebagai protected yang mana dapat diakses oleh kelas itu sendiri dan turunannya namun tidak dapat diakses diluar class. Maka untuk mengakses fungsi protected harus dibuatkan fungsi public yang nantinya mengembalikan nilai dari hasil fungsi protected. Pada kode diatas seperti fungsi aksesPin()

```
<!-- private -->
<?php
// buat class komputer
class komputer
{

    // property dengan hak akses protected
    private $jenis_processor = "Intel Core i7-4790 3.6Ghz";

    public function tampilkan_processor()
    {
        return $this->jenis_processor;
    }
}

// buat class laptop
class laptop extends komputer
{
```

```
    public function tampilkan_processor()
    {
        return $this->jenis_processor;
    }
}

// buat objek dari class laptop (instansiasi)
$komputer_baru = new komputer();
$laptop_baru = new laptop();

// jalankan method dari class komputer
echo $komputer_baru->tampilkan_processor(); // "Intel Core i7-4790 3.6Ghz"

// jalankan method dari class laptop (error)
echo $laptop_baru->tampilkan_processor();
// Notice: Undefined property: laptop::$jenis_processor
?>
```

Dalam kode di atas, saya membuat 2 buah class, yakni class komputer, dan class laptop. Class laptop merupakan turunan dari class komputer. Di dalam class komputer terdapat property $jenis_processor dengan akses level private. Di dalam class komputer dan class laptop, saya membuat method tampilkan_processor() yang digunakan untuk mengakses property $jenis_processor.

Pengaksesan method tampilkan_processor() dari objek $komputer_baru sukse ditampilkan karena berada di dalam satu class dimana property $jenis_processor berada.

Akan tetapi, jika method tampilkan_processor() diakses dari objek $laptop_baru yang merupakan turunan dari class komputer, PHP akan mengeluarkan error karena property $jenis_processor tidak dikenal.

Akses level private sering digunakan untuk menyembunyikan property dan method agar tidak bisa diakses di luar class.

Soal 4

Implementasi variable $this pada OOP.

```
<!-- tanpa this -->

<?php
class Buku
{
    public $pemilik = "Riyo";

    public function getData()
```

```
    {
        return "Buku ini milik $pemilik";
    }
}

$buku_flutter = new Buku();

echo "buku ini milik $buku_flutter->Pemilik";

echo $buku_flutter->getData();

?>
```

Potongan kode diatas adalah Ketika tidak menggunakan fungsi $this. Maka terdapat eror seperti :

> Notice: Undefined property: Buku::$Pemilik in C:\xampp\htdocs\PemWeb\praktikum4\functions\soal4.php on line 19
>
> buku ini milik
>
> Notice: Undefined variable: pemilik in C:\xampp\htdocs\PemWeb\praktikum4\functions\soal4.php on line 13
>
> Buku ini milik

Variabel $this adalah sebuah variabel khusus dalam OOP PHP yang digunakan sebagai penunjuk kepada objek, ketika kita mengaksesnya dari dalam class. Dalam manual PHP, $this disebut dengan istilah: pseudo-variable. Sehingga penulisannya sebagai berikut.

```
<!-- dengan this -->
<?php
class Buku_PemWeb
{
    public $pemilik = "Riyo";

    public function getData()
    {
        return "Buku ini milik {$this->pemilik}";
    }
}

$buku_flutter = new Buku_PemWeb();
echo "<br><br><br>";
echo $buku_flutter->getData();
?>
```

Potongan kode di atas adalah cara mengakses property pada class agar mampu kita inisialisasi sesuai keinginan.

Soal 5

```php
<?php
class Produk
{
    public $judul,
        $penerbit,
        $pengarang,
        $tahunTerbit;

    public function setData($judul = "null", $pengarang = "Anonim", $penerbit
= "Anonim", $tahunTerbit = "1999")
    {
        $this->judul = $judul;
        $this->penerbit = $penerbit;
        $this->pengarang = $pengarang;
        $this->tahunTerbit = $tahunTerbit;
    }

    public function getData()
    {
        return "Judul : {$this->judul} , Pengarang: {$this->pengarang},
Penerbit:  {$this->penerbit}, Tahun Terbit : {$this->tahunTerbit}";
    }
}

$produk1 = new Produk();
$produk1->setData("Tutorial flutter", "Budi Raharjo", "Informatika", "12 Mei
2018");
echo "Produk 1 = " . $produk1->getData();

?>
```

Potongan kode di atas adalah penggunaan pseudo-variable $this agar kita dapat mengakses property pada class tersebut.

Soal 6

```php
<?php

class Produk
{
```

```
    public $judul,
        $penerbit,
        $pengarang,
        $tahunTerbit;

    public function setData($judul = "null", $pengarang = "Anonim", $penerbit
= "Anonim", $tahunTerbit = "1999")
    {
        $this->judul = $judul;
        $this->penerbit = $penerbit;
        $this->pengarang = $pengarang;
        $this->tahunTerbit = $tahunTerbit;
    }

    public function getData()
    {
        return "Judul : {$this->judul} , Pengarang: {$this->pengarang},
Penerbit:  {$this->penerbit}, Tahun Terbit : {$this->tahunTerbit}";
    }
}

$produk1 = new Produk();
$produk1->setData("Tutorial flutter", "Budi Raharjo", "Informatika", "12 Mei
2018");
echo "Produk 1 = " . $produk1->getData();

?>
```

Potongan kode di atas adalah untuk membuat method yang mana pada code di atas terdapat method mengatur property yakni method setData() yang nantinya mengakses property. Dan fungsi getData() untuk mengambil data yang sudah diset pada fungsi sebelumnya.

Soal 7

```
<?php
// Membuat Class
class Produk
{
    // property
    public $judul,
        $penerbit,
        $pengarang,
        $tahunTerbit;
    // constructor
    public function __construct($judul = "null", $pengarang = "Anonim",
$penerbit = "Anonim", $tahunTerbit = "1999")
```

```
    {
        $this->judul = $judul;
        $this->penerbit = $penerbit;
        $this->pengarang = $pengarang;
        $this->tahunTerbit = $tahunTerbit;
    }

    // method
    public function getData()
    {
        return "Judul : {$this->judul} , Pengarang: {$this->pengarang},
Penerbit:  {$this->penerbit}, Tahun Terbit : {$this->tahunTerbit}";
    }

    // destructor
    public function __destruct()
    {
        echo "<br><br>this destructor <br>";
        echo "Judul : {$this->judul} , Pengarang: {$this->pengarang},
Penerbit:  {$this->penerbit}, Tahun Terbit : {$this->tahunTerbit}";
    }
}

// membuat objek dari Class
$produk1 = new Produk("Tutorial flutter", "Budi Raharjo", "Informatika", "12
Mei 2018");
echo "Dengan Constructor <br>";
echo "Produk 1 = " . $produk1->getData();

$produk2 = new Produk("Pemrograman Web", "Syarif Hidayat", "Informatika",
"2019");
echo "<br><br>";
echo "Produk 2 = " . $produk2->getData();

?>
```

Potongan kode di atas adalah Constructor yakni method khusus yang akan dijalankan secara otomatis pada saat sebuah objek dibuat (instansiasi), yakni ketika perintah “new” dijalankan.

Constructor biasa digunakan untuk membuat proses awal dalam mempersiapkan objek, seperti memberi nilai awal kepada property, memanggil method internal dan beberapa proses lain yang digunakan untuk mempersiapkan objek. Dalam PHP, constructor dibuat menggunakan method __construct().

Destructor adalah method khusus yang dijalankan secara otomatis pada saat sebuah objek dihapus. Di dalam PHP, seluruh objek sebanarnya sudah otomatis dihapus ketika halaman PHP selesai diproses. Tetapi kita juga dapat menghapus objek secara manual.

Destructor biasanya dipakai untuk membersihkan beberapa variabel, atau menjalankan proses tertentu sebelum objek dihapus. Dalam PHP, destructor dibuat menggunakan method __destruct().

Soal 8

```php
<?php
class Fruit
{
    public $name;
    public $color;

    public function __construct($name, $color)
    {
        $this->name = $name;
        $this->color = $color;
    }
    public function intro()
    {
        echo "The fruit is {$this->name} and the color is {$this->color}.";
    }
}

// Strawberry is inherited from Fruit
class Strawberry extends Fruit
{
    public function message()
    {

        echo "Am I a fruit or a berry? ";
    }
}
$strawberry = new Strawberry("Strawberry", "red");
echo "<br><br><br>";
$strawberry->message();
$strawberry->intro();
?>
```

Potongan kode di atas adalah implementasi inherintance atau penurunan. Inheritance adalah konsep OOP dimana sebuah class dapat menurunkan property dan method yang dimilikinya kepada class lain. Konsep inheritance dipakai untuk memanfaatkan fitur code reuse, yakni menghindari terjadinya duplikasi kode program. Dalam bahasa Indonesia, inheritance ini disebut juga sebagai pewarisan atau penurunan. Konsep inheritance membuat sebuah struktur atau hierarchy class dalam kode program. Class yang akan diturunkan bisa disebut sebagai class induk (parent class), super class, atau base class.

Sedangkan class yang menerima penurunan bisa disebut sebagai class anak (child class), sub class, derived class atau heir class. Tidak semua property dan method class induk akan diturunkan. Property dan method dengan hak akses private tidak akan diturunkan kepada class anak. Hanya property dan method dengan hak akses protected dan public saja yang bisa diakses dari class anak.

Soal 9

```php
<?php
// buat class komputer
class komputer2
{

    public function lihat_spec()
    {
        return "Spec Komputer: Acer,
     Processor Intel core i7, Ram 4GB";
    }
}

// turunkan class komputer ke laptop
class laptop2 extends komputer2
{

    public function lihat_spec()
    {
        return "Spec Laptop: Asus,
     Processor Intel core i5, Ram 2GB";
    }

    public function lihat_spec_komputer()
    {
        return parent::lihat_spec();
    }
}

echo "<br /> <br>";

// buat objek dari class laptop (instansiasi)
$gadget_baru = new laptop2();

//panggil method lihat_spec()
echo $gadget_baru->lihat_spec();

echo "<br />";
```

```php
//panggil method lihat_spec_komputer()
echo $gadget_baru->lihat_spec_komputer();
?>
```

Potongan kode di atas adalah untuk mengakses sutu method yang ada pada parent class. Cara nya yakni dengan menambagkan parent::namaMethod . kode diatas diterapkan pada fungsi lihat_spec_komputer() yang mengembalikan nilai pada method parent.

Soal 10

```php
class komputer
{

    // buat constructor class komputer
    public function __construct()
    {
        echo "Constructor dari class komputer <br />";
    }

    // buat destructor class komputer
    public function __destruct()
    {
        echo "Destructor dari class komputer <br />";
    }
}

// turunkan class komputer ke laptop
class laptop extends komputer
{

    // buat constructor class laptop
    public function __construct()
    {
        echo "Constructor dari class laptop <br />";
        parent::__construct();
    }

    // buat destructor class laptop
    public function __destruct()
    {
        echo "Destructor dari class laptop <br>";
        parent::__destruct();
    }
}

// turunkan class laptop ke chromebook
class chromebook extends laptop
```

```php
{

    // buat constructor class chromebook
    public function __construct()
    {
        echo "Constructor dari class chromebook <br />";
        parent::__construct();
    }

    // buat destructor class chromebook
    public function __destruct()
    {
        echo "Destructor dari class chromebook <br />";
        parent::__destruct();
    }
}
// buat objek dari class chromebook (instansiasi)
$gadget_baru = new chromebook();

echo "Belajar OOP PHP <br />";
```

Potongan Kode di atas merupakan cara mengakses constructor dan destructor dari child class ke parent class. Constructor dan destructor parent class akan dijalankan jika child class tidak mendefenisikan constructor dan destructor sendiri. Namun jika child class juga memiliki constructor dan desctructor, maka kita harus memanggil constructor dan destructor parent class secara manual.

Soal 11

```php
<?php
// buat class laptop
class laptop
{
    public $merk;
    public $pemilik;

    // static property
    public static $harga_beli;

    //static method
    public static function beli_laptop()
    {
        return "Beli Laptop";
    }
}

// set static property
laptop::$harga_beli = number_format(4000000, 0, '.', '.');
```

```php
// panggil static method
echo laptop::beli_laptop();

echo "<br />";

// get static property
echo "harga beli : Rp" . laptop::$harga_beli;
?>


<!-- Cara Mengakses Static Property dan Static Method Dari Class Itu Sendiri -->
<?php
// buat class laptop
class laptop2
{
    public $merk;
    public $pemilik;

    // static property
    public static $harga_beli;

    //static method
    public static function beli_laptop()
    {
        return "Beli laptop seharga Rp" . self::$harga_beli;
    }
}

// set static property
laptop2::$harga_beli = 4000000;

// panggil static method
echo laptop2::beli_laptop();
?>
```

Potongan kode diatas adalah cara mengakses static property. Static property dan static method adalah property (variabel) dan method (function) yang melekat kepada class, bukan kepada objek. Konsep static property memang ‘agak keluar’ dari konsep objek sebagai tempat melakukan proses, karena sebenarnya class hanya merupakan ‘blueprint’ saja.

Untuk membuat static property dan static method, kita menambahkan keyword ‘static’ setelah penulisan akses level property atau method.

Soal 12

```php
<!-- 1. Implementasi Konstanta Class dalam Pemrograman Objek -->
<!-- https://www.w3schools.com/php/php_oop_constants.asp -->

<?php
// buat class laptop
class laptopini
{
    // buat konstanta
    const DOLLAR = '12000';
}

// panggil konstanta class
echo "Harga dollar saat ini = Rp. " . laptopini::DOLLAR;
// hasil: Harga dollar saat ini = Rp. 12000

echo "<br>";
// buat objek dari class laptop (instansiasi)
$laptop_baru = new laptopini();

// panggil konstanta class
echo "Harga dollar saat ini = Rp. " . $laptop_baru::DOLLAR;
// hasil: Harga dollar saat ini = Rp. 12000
?>


<!-- Cara Mengakses Konstanta Class dari dalam Class itu Sendiri -->
<?php
// buat class laptop
class laptop
{

    // buat konstanta
    const DOLLAR = '12000';

    // buat method
    public function beli_laptop($harga)
    {
        return "Beli Komputer Baru, Rp. " . $harga * self::DOLLAR;
    }
}
echo "<br> <br>";
// buat objek dari class laptop (instansiasi)
$laptop_baru = new laptop();

echo $laptop_baru->beli_laptop(400);
// hasil: Beli Komputer Baru, Rp. 4800000
?>
```

```php
<!-- Cara Mengakses Konstanta Class milik Parent Class -->
<?php
echo "<br> <br>";
// buat class komputer
class komputer
{
    // buat konstanta class komputer
    const DOLLAR = '11000';
}

// turunkan class komputer ke class laptop
class laptop2 extends komputer
{
    // buat konstanta class laptop
    const DOLLAR = '12000';

    // buat method dengan konstanta class komputer
    public function beli_komputer_baru($harga)
    {
        return "Beli Komputer Baru, Rp ." . $harga * parent::DOLLAR;
    }

    // buat method dengan konstanta class laptop
    public function beli_laptop_baru($harga)
    {
        return "Beli Komputer Baru, Rp ." . $harga * self::DOLLAR;
    }
}

// buat objek dari class laptop (instansiasi)
$laptop_baru = new laptop2();

echo $laptop_baru->beli_laptop_baru(400);
echo "<br />";
echo $laptop_baru->beli_komputer_baru(400);
?>
```

Potongan kode diatas adalah cara menggunakan constanta pada property class. Konstanta Class atau class constant adalah konstanta yang berada di dalam class. Selain memiliki property dan method, PHP juga membolehkan kita menggunakan konstanta (constant) di dalam class.

Sebagaimana sifat konstanta reguler, class constant juga tidak bisa diubah nilainya ketika sudah didefenisikan. Untuk membuat class constant di dalam PHP, kita menggunakan perintah: const.

Soal 13

```php
<?php
class komputer
{
    final public function lihat_spec()
    {
        return "Lihat Spesifikasi Komputer";
    }
}

class laptop extends komputer
{
    public function lihat_spec()
    {
        return "Lihat Spesifikasi Laptop";
    }
}

$laptop_baru = new laptop();
// Fatal error: Cannot override final method
// komputer::lihat_spec()
?>
```

Potongan kode diatas adalah final method dan final class bisa digunakan untuk membuat desain class yang terstruktur. Pada kode program diatas, class komputer memiliki method lihat_spec() yang di-set sebagai final. Kemudian class komputer ini saya turunkan kepada class laptop. Di dalam class laptop, saya mendefenisikan ulang method lihat_spec(), sehingga method lihat_spec() milik class komputer akan tertimpa oleh method lihat_spec() milik class laptop.

Soal 14

```php
<?php
// buat abstract class
abstract class komputer
{
    // buat abstract method
    abstract public function lihat_spec();
}

class laptop extends komputer
{

    // implementasi abstract method
    public function lihat_spec()
    {
        return "Lihat Spec Laptop...";
```

```
    }

    // method 'biasa'
    public function beli_laptop()
    {
        return "Beli Laptop...";
    }
}

// buat objek dari class laptop
$laptop_baru = new laptop();
echo $laptop_baru->lihat_spec();
// Lihat Spec Laptop...

echo "<br />";

echo $laptop_baru->beli_laptop();
// Beli Laptop...
?>
```

Dalam kode diatas, method lihat_spec() telah kita implementasikan di dalam class laptop. Fitur inilah yang menjadi fungsi dari abstract method, yakni ‘memaksa’ setiap class turunan untuk memiliki method lihat_spec().

Implementasi dari abstract method, juga harus sesuai dengan signaturenya, yakni nama method beserta parameter. Jika kita membuat abstract method lihat_spec($merk), maka di dalam class turunan, kita juga harus membuat $merk sebagai parameter method.

Soal 15

```
<?php
// Interface definition
interface Animal
{
    public function makeSound();
}

// Class definitions
class Cat implements Animal
{
    public function makeSound()
    {
        echo " Meow ";
    }
}

class Dog implements Animal
{
```

```php
    public function makeSound()
    {
        echo " Bark ";
    }
}

class Mouse implements Animal
{
    public function makeSound()
    {
        echo " Squeak ";
    }
}

// Create a list of animals
$cat = new Cat();
$dog = new Dog();
$mouse = new Mouse();
$animals = array($cat, $dog, $mouse);

// Tell the animals to make a sound
foreach ($animals as $animal) {
    $animal->makeSound();
}

?>
```

Potongan kode diatas adalah Interface yakni ‘perjanjian method’, dimana jika sebuah class menggunakan interface, maka di dalam class tersebut harus tersedia implementasi dari method tersebut.

Soal 16

Di dalam pemrograman objek, polimorfisme adalah konsep dimana terdapat banyak class yang memiliki signature method yang sama. Implementasi dari method-method tersebut diserahkan kepada tiap class, akan tetapi cara pemanggilan method harus sama. Agar kita dapat ‘memaksakan’ signature method yang sama pada banyak class, class tersebut harus diturunkan dari sebuah abstract class atau object interface.

```php
<?php
// buat abstract class
abstract class komputer2
{
    // buat abstract method
    abstract public function booting_os();
}
```

```php

interface mouse
{
    public function double_klik();
}

class laptop2 extends komputer2 implements mouse
{
    public function booting_os()
    {
        return "Proses Booting Sistem Operasi Laptop";
    }
    public function double_klik()
    {
        return "Double Klik Mouse Laptop";
    }
}

class pc2 extends komputer2 implements mouse
{
    public function booting_os()
    {
        return "Proses Booting Sistem Operasi PC";
    }
    public function double_klik()
    {
        return "Double Klik Mouse PC";
    }
}

class chromebook2 extends komputer2 implements mouse
{
    public function booting_os()
    {
        return "Proses Booting Sistem Operasi Chromebook";
    }
    public function double_klik()
    {
        return "Double Klik Mouse Chromebook";
    }
}

// buat objek dari class diatas
$laptop_baru = new laptop2();
$pc_baru = new pc2();
$chromebook_baru = new chromebook2();

// buat fungsi untuk memproses objek
function booting_os_komputer2($objek_komputer)
```

```
{
    return $objek_komputer->booting_os();
}

function double_klik_komputer($objek_komputer)
{
    return $objek_komputer->double_klik();
}

// jalankan fungsi
echo booting_os_komputer2($laptop_baru);
echo "<br />";
echo double_klik_komputer($laptop_baru);
echo "<br />";
echo "<br />";

echo booting_os_komputer($pc_baru);
echo "<br />";
echo double_klik_komputer($pc_baru);
echo "<br />";
echo "<br />";

echo booting_os_komputer($chromebook_baru);
echo "<br />";
echo double_klik_komputer($chromebook_baru);
```

Pada kode program diatas, saya membuat 1 abstract class: komputer, dan 1 interface: mouse. Keduanya kemudian di turunkan kepada 3 class: class laptop, class pc, dan class chromebook.

Selama sebuah class diturunkan dari abstract class komputer, dan menggunakan interface mouse, fungsi booting_os_komputer() dan fungsi double_klik_komputer() akan selalu berhasil di jalankan, terlepas dari apapun nama objek dan implementasi method yang digunakan.

Tinjauan Pustaka/Dasar Teori

1. **PHP(Hypertext Preprocessor)**

   PHP adalah bahasa pemrograman server-side yang digunakan dalam pembuatan website bersama dengan CSS dan HTML. PHP merubah website dari statis menjadi lebih dinamis dan mengubah konten serta fungsi website yang lebih interaktif untuk keperluan user.

   PHP merupakan bahasa pemrograman yang populer hingga saat ini mengalahkan beberapa bahasa pemrograman lainnya, termasuk ASP.NET. Berdasarkan hasil survey dari W3Techs.com, PHP mendapatkan prosentase 78.9% mengalahkan bahasa pemrograman lainnya. Tentu ini prosentase yang besar jika dibandingkan dengan lainnya. Memang secara fungsi PHP bukan yang terbaik jika dibandingkan pemrograman web lainnya, tetapi secara pengguna PHP masih menjadi nomor satu.

2. Sejarah PHP

Menurut sejarahnya, PHP pertama kali muncul tahun 1994 diciptakan oleh Dr Leonardo Bernart. Awalnya PHP memiliki singkatan “Personal Home Page Tools”, selanjutnya PHP diganti nama menjadi FI (Form Interpreter). Sejak kemunculan PHP versi 3.0, nama PHP kembali lagi digunakan dengan singkatan menjadi “Hypertext Preprocessor” hingga sekarang ini.

Pada survey yang dilakukan bulan Desember tahun 1999, sudah ada lebih dari sejuta website yang menggunakan PHP termasuk diantaranya website NASA, RedHat dan Mitsubishi. Untuk sekarang ini website yang menggunakan PHP sudah tak terhitung lagi jumlahnya.

3. Sintak dasar PHP

PHP dapat kita tempatkan dimana saja sesuai kebutuhan yang sekiranya memerlukan tag tersebut untuk menjalankan fungsi PHP. Tag PHP diawali dengan <?php ?>, dan untuk ekstensi file .php

Penulisan variable pada php diawali dengan tanda ‘$’ contoh $test = “hello world”; . setiap kita membuat variable maka harus diawali tanda dolar yang isinya bisa apa saja seperti integer, string, array, fungsi dan semacamnya.

Untuk mencetak nilai ke layer kitab isa menggunakan sintax “echo” , “print”. Kedua syntak ini fungsinya sama yaitu untuk mencetak nilai ke layar. Tetapi ada perbedaan, fungsi echo tidak mengembalikan nilai. Untuk fungsi print mengembalikan nilai 1 saat dilakukan eksekusi dan hanya bisa menggunakan satu parameter saja, jika lebih dari itu bisa terjadi error.

Mengenal Tipe Data di PHP

Variabel yang sudah kita buat bisa kita simpan dengan berbagai jenis data. Jenis-jenis data ini disebut tipe data.

Ada beberapa macam tipe data yang dapat disimpan dalam variabel:

- Tipe data char (karakter)
- Tipe data string (teks)
- Tipe data integer (angka)
- Tipe data float (pecahan)
- Tipe data boolean
- Tipe data objek
- Tipe data Array
- NULL
- dll.

Percabangan If

Bentuk yang paling sederhana dari percabganan adalah “If” saja. Biasanya digunakan saat hanya ada satu tindakan yang harus dilakukan. Kondisi atau pernyataan ini akan bernilai true dan false. Jika true (benar), maka kode yang ada di dalamnya akan dieksekusi. Namun, apabila false maka tidak akan mengeksekusinya.[3]

Percabangan If/Else

Percabangan If/Else memiliki dua pilihan. Jika <kondisi> bernilai false, maka blok else akan dikerjakan.

Percabangan If/Elseif/Else

Percabangan If/Elseif/Else memiliki lebih dari dua pilihan kondisi.

Percabangan Switch/Case

Percabangan Switch/Case adalah bentuk lain dari percabangan If/Elseif/Else.

Percabangan dengan Operator Ternary

Percabangan menggunakan operator ternary adalah bentuk sederhana dari percabangan If/Else.

Percabangan Bersarang

Percabangan bersarang artinya ada percabangan di dalam percabangan (nested).

Perulangan

Ada dua jenis perulangan dalam pemrograman:

1. Counted loop;
2. Uncounted loop.

Apa perbedaanya?

Counted loop adalah perulangan yang sudah jelas banyak pengulangannya. Sedangkan Uncounted loop tidak pasti berapa kali dia akan mengulang.[4]

Pada PHP ada 4 jenis perulangan yang bisa kita gunakan:

- Perulangan For
- Perulangan While
- Perulangan Do/While
- Perulangan Foreach

Perulangan For

Perulangan For adalah perulangan yang termasuk dalam counted loop, karena kita bisa menentukan jumlah perulangannya.

Perulangan While

Perulangan while adalah perulangan yang termasuk dalam uncounted loop. Karena biasanya digunakan untuk mengulang sesuatu yang belum jelas jumlah pengulangannya.

Namun, perulangan while juga bisa digunakan seperti perulangan for sebagai counted loop.

Perulangan Do/While

Perulangan Do/While sama seperti perulangan while. Ia juga tergolong dalam uncounted loop. Perbedaan Do/While dengan while terletak pada cara iya memulai pengulangan. Perulangan Do/While akan selalu melakukan pengulangan sebanyak 1 kali, kemudian melakukan pengecekan kondisi.

Sedangkan perulangan while akan mengecek kondisi terlebih dahulu, baru melakukan pengulangan.

Perulangan Foreach

Perulangan foreach sama seperti perulangan for. Namun, ia lebih khusus digunakan untuk mecetak array.

Perulangan Bersarang

Perulangan bersarang adalah istilah untuk menyebut perulangan di dalam perulangan. Dalam bahasa inggris, perulangan bersarang disebut nested loop.

Apa itu Array?

Array adalah salah satu struktur data yang berisi sekumpulan data dan memiliki indeks. Indeks digunakan untuk mengakses nilai array.[5] Indeks array selalu dimulai dari nol (0).

Membuat Array di PHP

Array di PHP dapat kita buat dengan fungsi array() dan tanda kurung kotak [].

Menampilkan isi Array

Untuk menampilkan isi array, kita bisa mengaksesnya melalui indeks.

Menghitung isi array

Kita bisa menggunakan fungsi count() untuk menghitung banyaknya isi array.

Menghapus isi Array

Untuk menghapus isi array, kita bisa menggunakan fungsi unset(). Fungsi ini juga dapat digunakan untuk menghapus variabel.

Menambahkan isi Array

Ada dua cara yang bisa dilakukan untuk menambah isi array:

- Mengisi langsung ke nomer indeks yang ingin ditambahkan
- Mengisi langsung ke indeks terakhir

Apabila kita menambahkan pada indeks yang sudah memiliki isi, maka isinya akan ditindih dengan yang baru.

Array Asosiatif

Array asosiatif adalah array yang indeksnya tidak menggunakan nomer atau angka. Indeks array asosiatif berbentuk kata kunci.Pada array asosiatif, kita menggunakan tanda => untuk mengasosiasikan sebuah kata kunci dengan isi array.

Array Multi Dimensi

Array multi dimensi adalah array yang memiliki dimensi lebih dari satu. Biasanya digunakan untuk membuat matrik, graph, dan stuktur data rumit lainnya.

4. **Pemrograman Berbasis Objek**

Object Oriented Programming atau dalam bahasa indonesia diartikan Pemrograman Berbasis Objek, adalah salah satu cara membuat program (programming paradigm) dengan memecah alur program menjadi modul-modul sederhana yang disebut dengan objek. Setiap objek akan memiliki fungsi dan tugas tersendiri. OOP berbeda dengan prosedural programming yang memecah program menjadi fungsi-fungsi/prosedural.

5. **Class, Objek dan Property**

Class adalah cetak biru atau blueprint dari object. Class digunakan hanya untuk membuat kerangka dasar. Yang akan kita pakai nantinya adalah hasil cetakan dari class, yakni object.

Sebagai analogi, class bisa diibaratkan dengan laptop atau notebook. Kita tahu bahwa laptop memiliki ciri-ciri seperti merk, memiliki keyboard, memiliki processor, dan beberapa ciri khas lain yang menyatakan sebuah benda tersebut adalah laptop. Selain memiliki ciri-ciri, sebuah laptop juga bisa dikenakan tindakan, seperti: menghidupkan laptop atau mematikan laptop.

Class dalam analogi ini adalah gambaran umum tentang sebuah benda. Di dalam pemrograman nantinya, contoh class seperti: koneksi_database dan profile_user.

Method adalah tindakan yang bisa dilakukan di dalam class. Jika menggunakan analogi class laptop kita, maka contoh method adalah: menghidupkan laptop, mematikan laptop, mengganti cover laptop, dan berbagai tindakan lain.

Method pada dasarnya adalah function yang berada di dalam class. Seluruh fungsi dan sifat function bisa diterapkan ke dalam method, seperti argumen/parameter, mengembalikan nilai (dengan keyword return), dan lain-lain.

Property (atau disebut juga dengan atribut) adalah data yang terdapat dalam sebuah class. Melanjutkan analogi tentang laptop, property dari laptop bisa berupa merk, warna, jenis processor, ukuran layar, dan lain-lain.

Property ini sebenarnya hanyalah variabel yang terletak di dalam class. Seluruh aturan dan tipe data yang biasa diinput ke dalam variabel, bisa juga diinput kedalam property. Aturan tata cara penamaan property sama dengan aturan penamaan variabel.

Object atau Objek adalah hasil cetak dari class, atau hasil 'konkrit' dari class. Jika menggunakan analogi class laptop, maka objek dari class laptop bisa berupa: laptop_andi, laptop_anto, laptop_duniailkom, dan lain-lain. Objek dari class laptop akan memiliki seluruh ciri-ciri laptop, yaitu property dan method-nya.

Proses 'mencetak' objek dari class ini disebut dengan 'instansiasi' (atau instantiation dalam bahasa inggris). Pada PHP, proses instansiasi dilakukan dengan menggunakan keyword 'new'. Hasil cetakan class akan disimpan dalam variabel untuk selanjutnya digunakan dalam proses program.

Daftar Pustaka / Referensi

[1] D. Y. Aji, "codepolitan," 16 Mei 2016. [Online]. Available: https://www.codepolitan.com/pengenalan-html5-belajar-html/.

[2] anonim, "w3school," [Online]. Available: https://www.w3schools.com/TAgs/default.asp.

[3] anonim, "w3school," [Online]. Available: https://www.w3schools.com/css/.

[4] w3school, "JSON," [Online]. Available: https://www.w3schools.com/js/js_json_intro.asp. [Accessed 25 Februari 2022].

[5] petaniKode, "Petani Kode," Petani Kode, 11 Maret 2016. [Online]. Available: https://www.petanikode.com/json-pemula/. [Accessed 28 Februari 2022].

[6] Anonim, "dev.bertzzie.com," [Online]. Available: http://dev.bertzzie.com/knowledge/javascript-lanjut/XMLHttpRequest-AJAX.html. [Accessed 2 Maret 2022].

[7] Anonim, "hostinger.co.id," [Online]. Available: https://www.hostinger.co.id/tutorial/apa-itu-jquery. [Accessed 2 Maret 2022].

[8] dewa, "Dewaweb," [Online]. Available: https://www.dewaweb.com/blog/belajar-jquery-pengertian-dan-cara-menggunakan-jquery/. [Accessed 4 Maret 2022].

[9] S. Awwaabiin, "Pengertian PHP, Fungsi dan Sintaks Dasarnya," Niagahoster, 2 November 2021. [Online]. Available: https://www.niagahoster.co.id/blog/pengertian-php/. [Accessed 2022 Mei 20222].

[10] E. Lararenjana, "PHP Adalah Bagian dari Bahasa Pemrograman, Berikut Penjelasan Selengkapnya," merdeka.com, 9 Desember 2020. [Online]. Available: https://www.merdeka.com/jatim/php-adalah-bagian-dari-bahasa-pemograman-berikut-penjelasan-selengkapnya-kln.html. [Accessed 26 Mei 2022].

[11] E. O. Choiri, "Pengertian PHP dan Fungsinya Dalam Pemrograman Web," qwords.com, 11 Maret 2020. [Online]. Available: https://qwords.com/blog/pengertian-php/. [Accessed 28 Mei 2022].

[12] w3school, "Learn PHP," w3schools, [Online]. Available: https://www.w3schools.com/php/. [Accessed 26 Mei 2022].

[13] w3school-php, "What is OOP PHP," w3school, 12 April 2015. [Online]. Available: https://www.w3schools.com/php/php_oop_what_is.asp. [Accessed 30 Mei 2022].

[14] duniailkom, "Tutorial Belajar OOP PHP Part 2," duniaIlkom.com, 29 September 2014. [Online]. Available: https://www.duniailkom.com/tutorial-belajar-oop-php-pemrograman-berbasis-objek-php/. [Accessed 31 Mei 2022].