MODUL 2
CLASS DAN OBJECT

2.1 Tujuan

Setelah mempelajari modul ini anda diharapkan mampu :

a. Memahami konsep *class* dan langkah pembuatan *class* menggunakan bahasa pemrograman PHP.
b. Memahami dan mengetahui bagian dan fungsi yang membangun sebuah *class*.
c. Memahami dan merancang sistem yang dilengkapi dengan *class* beserta bagian dan fungsi yang terdapat di dalam *class* tersebut dengan baik.

2.2 Praktikum

a. *Class*

Untuk membuat sebuah *class,* perintah yang digunakan adalah *class* yang diikuti dengan nama class yang diinginkan. Baca kembali diktat teori untuk mengetahui cara pembuatan nama kelas yang baik. Nama *file* yang menyimpan *class* ini sama dengan nama class.

```
1. <?php
2. class Mahasiswa {
5.     // class
6. }
```

Buatlah sebuah *file* dengan nama *Mahasiswa.php* dan masukkan perintah diatas untuk membuat sebuah *class* baru. Jika *file* tersebut diakses melalui *localhost*, maka tampilan *file* hanya kosog saja, karena belum ada *object* yang diciptakan dari *class* tersebut.

b. *Property*

Property merupakan variabel yang terletak di dalam *class*, masing-masing *property* mempunyai *access modifier. Access modifier* merupakan cara bagaimana hak akses terhadap variabel tersebut, akses tersebut yaitu : (1) **Public**, (2) **Private**, (3) **Protected**.

Dengar penjelasan dosen pengampu untuk penjelasan *Property* ini, buatlah sebuah *file* baru dan masukkan perintah berikut ini:

```php
<?php

class Mahasiswa {

    // Access Modifier, Typed Properties dan Property

    // Bisa diakses class yang sama dan class lain.
    public string $nim;
    public string $nama;
    public int $umur;

    // Hanya bisa diakses dalam class yang sama.
    private string $email;

    // Hanya bisa diakses dalam class yang sama dan anak-anaknya.
    protected string $nama_ibu;

    /*  1. bool
        2. int
        3. float
        4. string
        5. array
        6. iterable
        7. object
        8. ?(nullable)
        9. self & parent
        10. Classes & interfaces
    */

}
```

c. *Method*

Method merupakan *function* yang terdapat di dalam *class*, *method* juga memiliki *Access modifier* sama dengan *property*. *Method* setara dengan *property*, jika terdapat variabel di dalam *method*, maka itu tidak disebut dengan *property*, tapi hanya *local* variabel saja. Perhatikan perintah beriku ini, buatlah *file* baru dan masukkan perintah tersebut.

```php
<?php

class Mahasiswa {

    // Method

    public function setNim(string $nim) {
        return $nim;
    }

    public function setNama(string $nama) {
        return $nama;
    }

    public function setUmur(int $umur) {
        return $umur;
    }
}
```

d. Constructor

Constructor merupakan *method default* yang dipanggil oleh *class* ketika tidak disebutkan *method* khusus pada *object* dari *class* tersebut. Perintah yang digunakan untuk *constructor* ini adalah :

__constructor($var, $var, …)

Buatlah sebuah *file* baru dan masukkan perintah beriku ini:

```php
<?php

class Mahasiswa {

    public string $nim;
    public string $nama;

    // Constructor

    public function __construct(string $a, string $b) {

        $this->nim = $a;
        $this->nama = $b;
    }
}
```

e. *Instantiation*

Instantiation atau *Instance* merupakan proses utama yang menjadi *class* ke dalam *object*. Proses ini yang menjadi ujung dari konsep OOP pada bahasa pemrograman PHP. Seluruh *Property* dan *Method* yang ada dalam *class* digunakan oleh o*bject* sesuai dengan tujuan dan *access modifier*-nya. Proses *Instance* menggunakan perintah "**new**". Buatlah *file* baru dan bahaslah kode perintah berikut ini dengan seksama.

```
1. <?php

2. class Mahasiswa {

5.     public string $nim;
6.     public string $nama;
7.     public static string $agama = "Islam";

9.     public function setNim(string $nim) {
10.          return $nim;
11.      }
12.
13.      public function setNama(string $b) {

15.          // $this keyword refers a
   non-static member of a class
16.          return $this->nama = $b;
17.      }
18.
19.      public function getNama() {

21.          // $this keyword refers a
   non-static member of a class
22.          return $this->nama;
23.      }
24.
25.      public static function getAgama() {

27.          // self keyword refers a static member of a class

28.          return self::$agama;
29.      }
30.  }
31.
```

```
32.  // Instantiation

34.  $mhsw = new Mahasiswa();

36.  echo $mhsw->setNim('17021000');

38.  $mhsw->setNama('Faiza');

40.  echo $mhsw->getNama();

42.  echo $mhsw->getAgama();
```

perintah ***new*** pada baris 34 menjadikan $mhsw sebagai *object* dari *class* Mahasiswa()

f. *Static Keyword*

Static keyword digunakan pada *Property* dan *Method* berfungsi agar bisa dipanggil langsung pada objek tanpa melalui proses *instantiation.*

```
1. <?php
2. class Mahasiswa {
5.     public static string $agama = "Islam";
7.     public static function getAgama() {
9.         //
11.      }
12.  }
```

g. *Scope Resolution Operator*

Scope Resolution Operator atau double colon (::) atau titik dua, dua kali, merupakan tanda yang digunakan pada proses *Instantiation* dari *Property* atau *Method Static* (*Static keyword)*. Lihat materi pada huruf f untuk *Static keyword.* Buatlah sebuah *file* baru dan coba perintah dibawah untuk *Scope Resolution Operator*

```php
<?php

class Mahasiswa {

    public static function setNama(string $nama) {

        return $nama;

    }
}

// Instantiation with Scope resolution operator
// Paamayim Nekudotayim

echo Mahasiswa::setNama('Faiza');
```

h. *Error Handling*

Error Handling berfungsi untuk menangani *error* yang terdapat di dalam *class*. Untuk penjelasan lebih lanjut, silakan baca kembali Diktat perkuliahan pada sesi teori. Buatlah sebuah *file* baru dan bahas kode perinta berikut ini.

```php
<?php

class Mahasiswa {

    public int $umur = 22;

    public function getUmur() {

        try {

            if ($this->umur < 25) {
                throw new Exception('Anda masih muda');
            }

        } catch (Exception $e) {

            die ("Maaf Error, " . $e->getMessage());

        }
    }
}

$mhsw = new Mahasiswa();
$mhsw->getUmur();
```

2.3 Latihan

a. Sempurnakanlah Latihan pada praktikum 1 dengan mengimplementasikan bagian dan fungsi *class* sesuai dengan materi praktikum 2.
b. Lengkapi sistem pada latihan praktikum 2 dengan perintah input, edit dan delete data sesuia dengan database praktikum 1.