MATERIA: TECNOLOGÍA Y APLICACIONES WEB (TAW)

MAESTRO: MSI MARIO HUMBERTO RODRÍGUEZ CHÁVEZ

ALUMNO: LUIS ANGEL TORRES GRIMALDO

PERIODO: ABRIL-JULIO 2019

FECHA: DOMINGO 5 DE MAYO DEL 2019

PROGRAMACIÓN ORIENTADA A OBJETOS (POO): Se puede decir que la POO es un conjunto de técnicas que ayudan a enfocar todas las variables pertenecientes a un problema o situación y de esta manera llegar a una solución.

La aplicación de esta técnica difiere un poco en a sintaxis de los muchos lenguajes que existen en el mercado, regularmente la diferencia serían simbologías que se utilizan para delimitar, tales como:

- {}
- :
- ;

Siempre recordando que la lógica es la misma, junto con las palabras clave, elementos, componentes y reglas del paradigma.

Las 4 características que definen este paradigma son:

1. Encapsulamiento
2. Herencia
3. Polimorfismo
4. Abstracción

Encapsulamiento: Lleva a un mismo nivel todos los conceptos dentro de la aplicación que tengan relación de forma directa, proporcionando un mejor control. Una aplicación notoria de esto es la reducción de variables dentro del código ya que un objeto de una clase puede contener dentro las variables y funciones que se requieran.
Herencia: Este término se puede asociar al concepto de "reutilización de código", ya que esta propiedad del paradigma permite delegar a una clase hipotética "b" las propiedades y métodos de una clase hipotética "a" con tan solo una pequeña modificación al definir la clase "b". Puede llegar a ser fácil de confundir el "reutilizar código" a copiar y pegar lineas de un archivo a otro cuando se es un beginer en la programación, sin embargo no es así.
Abstracción Propiedad que define solo y únicamente dentro de una clase sus propiedades y métodos necesarios, ya sean relativamente muchos o pocos.
Modularidad Característica que permite dividir una aplicación en varias partes más pequeñas (denominadas módulos), independientes unas de otras**.**
Ocultación (aislamiento) Los objetos están aislados del exterior, protegiendo a sus propiedades para no ser modificadas por aquellos que no tengan derecho a acceder a las mismas.
Polimorfismo Esta es la propiedad que permite que un objeto a tenga las mismas propiedades que un objeto b, sin embargo esto no quiere decir que sean necesariamente iguales o que dependa uno del otro, ya que sus datos pueden llegar a estar almancenados en un espacio de memoria distinto.

Recolección de basura Es la técnica que consiste en destruir aquellos objetos cuando ya no son necesarios, liberándolos de la memoria.

Dentro de una clase se puede generar 3 distintos tipos de elementos:

1. Variables
2. Constantes
3. Métodos

Su sintaxis en PHP es la siguiente:

```php
<?php
class bloques
{
    // declaración de una propiedad
    const CONSTANTE = 'mi constante';

    // declaración de una propiedad
    public $var = 'valor por defecto';

    // declaración un de un método
    public function imprimirVar() {
        echo $this->var;
    }
}

?>
```

Ahora se procede a mostrar la sintaxis de este paradigma en el lenguaje PHP:

```php
<?php
    class persona{
        private $nombre;
        private $edad;

        public function persona($_nombre,$_edad){//Método constructor
            $this->nombre=$_nombre;
            $this->edad=$_edad;
        }
        public function mostrar_datos(){
            echo "<br>Nombre: $this->nombre, edad: $this->edad<br><br>";
        }
    }

    $persona_1 = new persona("Luis",22);
    $persona_1->mostrar_datos();
?>
```

Con respecto al concepto de ocultación existen 3 niveles que se aplican tanto en variables de la clase (atributos) como en las funciones de la clase (métodos). Son:

1. **Public**. A este tipo de elementos pueden acceder desde cualquier lugar, ya sea dentro de la clase o fuera por algún otro objeto de una clase distinta.
2. **Private**. Solo es visible y accesible dentro de la clase por los métodos que se implementan dentro.
3. **Protected**. Es parecido a **Public** solo que a esta accederán solo elementos que heredan la clase donde se definió el método o propiedad protegida o también se puede acceder desde la misma clase donde se declaró.

Otro punto importante dentro del paradigma orientado a objetos es **el método constructor**, el cual se encarga de inicializar las variables que se quieran. En el ejemplo anterior se representa la sintaxis de un constructor en PHP, como se puede observar el constructor debe ser declarado como publico y debe tener el mismo nombre de la clase. Específicamente en PHP no es estrictamente necesario este método ya que de no declararse un método constructor se define un objeto de la siguiente forma:

```php
<?php
    class persona{
        public $nombre;
        public $edad;

     #      public function persona($_nombre,$_edad){//Método constructor
      #         $this->nombre=$_nombre;
       #        $this->edad=$_edad;
        #   }
        public function mostrar_datos(){
            echo "<br>Nombre: $this->nombre, edad: $this->edad<br><br>";
        }
    }

#    $persona_1 = new persona("Luis",22);
 #   $persona_1->mostrar_datos();

    $persona_2 = new persona();
    $persona_2->nombre="Luis";
    $persona_2->edad=22;
    $persona_2->mostrar_datos();
?>
```

En este código se comentó el código dónde se definió el constructor anteriormente y las variables o atributos de la clase se colocaron como públicas para poder acceder a ellas fuera del constructor. Otro dato curioso es que en la POO existe un término llamado sobrecarga de métodos, el cual permite definir más de un método con el mismo nombre pero siempre cambiando el orden y número de operadores, sin embargo, esto no existe en PHP. En este lenguaje o se declara un método constructor o no se declara.