Ejercicios PHP 8

1. Banco

Creamos la clase CuentaBancaria con las propiedades titular, saldo y tipoDeCuenta y los métodos depositar (Ingresa dinero en la cuenta), retirar (Retira dinero de la cuenta) y mostrarInfo (Muestra los datos de la cuenta). Creamos una instancia de la clase y probamos los métodos.

```php
<?php
class CuentaBancaria{

    public $titular; //Titular de la cuenta
    public $saldo; //Saldo de la cuenta
    public $tipoDeCuenta; //Tipo de la cuenta

    public function depositar($cantidad){ //Ingresa dinero en la cuenta

        return $this->saldo = $this->saldo + $cantidad;

    }

    public function retirar($cantidad){ //Retira dinero de la cuenta

        if ($cantidad > $this->saldo){ //Comprueba si hay suficiente
dinero para retirar

            throw new Exception("Saldo insuficiente");

        } else {

            return $this->saldo = $this->saldo - $cantidad;

        }
    }

    public function mostrarInfo(){ //Muestra los datos de la cuenta

        return "Titular: ".$this->titular."\n".
        "Saldo: ".$this->saldo."\n".
        "Tipo de cuenta: ".$this->tipoDeCuenta."\n";

    }
}

$cuenta = new CuentaBancaria();
```

```php
$cuenta->titular = "Mariano Reyes";
$cuenta->saldo = 500;
$cuenta->tipoDeCuenta = "cuenta de ahorro";

do{
    $operacion = readline("Introduzca la operación deseada: ");

    try {

        if ($operacion == "depositar"){ //Ingresa dinero

            $cantidad = readline("Introduzca la cantidad a depositar: ");
            $cuenta->depositar($cantidad);
            echo "Operación completa"."\n";

        } elseif ($operacion == "retirar"){ //Retira dinero

            $cantidad = readline("Introduzca la cantidad a retirar: ");
            $cuenta->retirar($cantidad);
            echo "Operación completa"."\n";

        } elseif ($operacion == "info"){ //Muestra los datos de la cuenta

            echo $cuenta->mostrarInfo();

        } elseif ($operacion == "salir"){ //Cierra el programa

            break;

        } else { //Mensaje si no se escoge una opción válida

            echo "Operación inválida";

        }
    }

    catch(Exception $e) {

        echo "Saldo insuficiente"."\n";

    }
} while ($operacion != "salir");

?>
```

```
TERMINAL
PS C:\xampp\htdocs\programacion\ejercicio8> php banco.php
Introduzca la operación deseada: info
Titular: Mariano Reyes
Saldo: 500
Tipo de cuenta: cuenta de ahorro
Introduzca la operación deseada: depositar
Introduzca la cantidad a depositar: 50
Operación completa
Introduzca la operación deseada: retirar
Introduzca la cantidad a retirar: 800
Saldo insuficiente
Introduzca la operación deseada: retirar
Introduzca la cantidad a retirar: 25
Operación completa
Introduzca la operación deseada: info
Titular: Mariano Reyes
Saldo: 525
Tipo de cuenta: cuenta de ahorro
Introduzca la operación deseada: salir
PS C:\xampp\htdocs\programacion\ejercicio8>
```

2. Gestor de tareas

Creamos la clase Tarea con las propiedades nombre, descripción, fechaLimite y estado y los métodos marcarComoCompletada, editarDescripcion y mostrarTarea. Creamos una instancia de la clase y probamos los métodos.

```php
<?php
class Tarea{

    public $nombre; //Nombre de la tarea
    public $descripcion; //Descripción de la tarea
    public $fechaLimite; //Fecha límite de la tarea
    public $estado; //Estado de la tarea

    public function marcarComoCompletada(){ //Marca la tarea como
completada

        return $this->estado = "completada";

    }

    public function editarDescripcion($nuevaDescripcion){ //Cambia la
descripción de la tarea

        return $this->descripcion = $nuevaDescripcion;
```

```
    }

    public function mostrarTarea(){ //Muestra los datos de la tarea

        return "Nombre: ".$this->nombre."\n".
        "Descripción: ".$this->descripcion."\n".
        "Fecha límite: ".$this->fechaLimite."\n".
        "Estado: ".$this->estado."\n";

    }
}

$tarea1 = new Tarea;
$tarea1->nombre = "Tarea 1";
$tarea1->descripcion = "Primera tarea";
$tarea1->fechaLimite = "16/1/2025";
$tarea1->estado = "incompleto";

$tarea2 = new Tarea;
$tarea2->nombre = "Tarea 2";
$tarea2->descripcion = "Segunda tarea";
$tarea2->fechaLimite = "17/1/2025";
$tarea2->estado = "incompleto";

$tarea3 = new Tarea;
$tarea3->nombre = "Tarea 3";
$tarea3->descripcion = "Tercera tarea";
$tarea3->fechaLimite = "18/1/2025";
$tarea3->estado = "incompleto";

$tareas = array(1=>$tarea1, 2=>$tarea2, 3=>$tarea3);

do{
    $tarea = readline("Elija la tarea: "); //Escoge con que tarea
interactuar

    do{
        $opcion = readline("Escoja que hacer con la tarea: "); //Escoge
una opción

        if ($opcion == "Completar"){

            $tareas[$tarea]->marcarComoCompletada();
            echo "Tarea completada"."\n";

        } elseif ($opcion == "editar"){
```

```php
            $nuevaDescripcion = readline("Introduzca la nueva
descripción: ");
            $tareas[$tarea]->editarDescripcion($nuevaDescripcion);
            echo "Descripción editada"."\n";

        } elseif ($opcion == "mostrar"){

            echo $tareas[$tarea]->mostrarTarea();

        } elseif ($opcion == "salir"){

            break;

        } else{

            echo "Opción inválida"."\n";

        }
    } while ($opcion != "salir"); //Permite cambiar de tarea al
introducir salir
} while ($tarea != "salir") //Cierra la aplicacion al introducir salir al
elegir tarea
?>
```

```
PS C:\xampp\htdocs\programacion\ejercicio8> php tarea.php
Elija la tarea: 1
Escoja que hacer con la tarea: mostrar
Nombre: Tarea 1
Descripción: Primera tarea
Fecha límite: 16/1/2025
Estado: incompleto
Escoja que hacer con la tarea: editar
Introduzca la nueva descripción: tarea 1
Descripción editada
Escoja que hacer con la tarea: Completar
Tarea completada
Escoja que hacer con la tarea: mostrar
Nombre: Tarea 1
Descripción: tarea 1
Fecha límite: 16/1/2025
Estado: completada
Escoja que hacer con la tarea: salir
Elija la tarea: salir
Escoja que hacer con la tarea: salir
PS C:\xampp\htdocs\programacion\ejercicio8>
```

3. Empleado y Consultor

Creamos la clase Empleado con las propiedades nombre, sueldo y añosExperiencia y los métodos calcularBonus y mostrarDetalles. Creamos la clase hija Consultor que añade la propiedad horasPorProyecto y modificamos calcularBonus para incluirlo. Creamos una instancia de las clases, probamos los métodos y comparamos los bonuses.

```php
<?php
class Empleado{

    public $nombre; //Nombre del empleado
    public $sueldo; //Sueldo del empleado
    public $aniosExperiencia; //Años trabajados del empleado

    public function calcularBonus(){ //Calcula el bonus del empleado

        return "Bonus del empleado: ".($this->sueldo*5/100)*($this->aniosExperiencia/2)."€"."\n";

    }

    public function mostrarDetalles(){ //Muestra datos del empleado

        return "Nombre: ".$this->nombre."\n".
        "Sueldo: ".$this->sueldo."\n".
        "Años de experiencia: ".$this->aniosExperiencia."\n";

    }
}

class Consultor extends Empleado{

    public $horasPorProyecto; //Horas ue dura cada proyecto

    public function calcularBonus(){ //Calcula el bonus del consultor

        if ($this->horasPorProyecto > 100){

            return "Bonus del consultor: ".(($this->sueldo*5/100)*($this->aniosExperiencia/2)*2)."€"."\n";
        } else{

            return "Bonus del consultor: ".($this->sueldo*5/100)*($this->aniosExperiencia/2)."€"."\n";

        }
    }
```

```php
}

$empleado = new Empleado;
$empleado->nombre = "Manuel Pérez";
$empleado->sueldo = 1500;
$empleado->aniosExperiencia = 15;

$consultor = new Consultor;
$consultor->nombre = "Antonio Perales";
$consultor->sueldo = 1900;
$consultor->aniosExperiencia = 20;
$consultor->horasPorProyecto = 300;

echo $empleado->calcularBonus();
echo $consultor->calcularBonus();

if ($empleado->calcularBonus() == $consultor->calcularBonus()){

    echo $empleado->nombre." y ".$consultor->nombre." tienen el mismo
bonus"."\n";

} elseif ($empleado->calcularBonus() > $consultor->calcularBonus()){

    echo "El bonus de ".$empleado->nombre." es mayor que el de
".$consultor->nombre."\n";

} else{

    echo "El bonus de ".$consultor->nombre." es mayor que el de
".$empleado->nombre."\n";
}
?>
```

```
TERMINAL
PS C:\xampp\htdocs\programacion\ejercicio8> php Empleado.php
Bonus del empleado: 562.5€
Bonus del consultor: 1900€
El bonus de Manuel Pérez es mayor que el de Antonio Perales
PS C:\xampp\htdocs\programacion\ejercicio8>
```

4. Carrito de compras

Creamos la clase Carrito con la propiedad productos y los métodos agregarProducto, quitarProducto, calcularTotal y mostrarDetallesCarrito. Creamos una instancia de la clase y probamos los métodos. (Ejercicio incompleto)

```php
<?php
class Carrito{

    public $productos; //Productos del carrito

    public function agregarProducto($nombre, $precio, $cantidad){ //Añade
productos al carrito

        $producto = array("nombre"=>$nombre, "precio"=>$precio,
"cantidad"=>$cantidad);
        return array_push($this->productos, $producto);


    }

    public function quitarProducto($nombre){ //Quita productos del
carrito

        for ($x = 0; $x <= count($this->productos); $x++){

            $producto = $this->productos[$x];

            if ($producto["nombre"] == $nombre) {

                unset($this->productos[$x]);

            }
        }
    }

    public function calcularTotal(){ //Calcula el precio total de la
compra

        $precioTotal = 0;
        for ($x = 0; $x <= count($this->productos); $x++){

            $producto = $this->productos[$x];
            $precioProducto = $producto["precio"]*$producto["cantidad"];
            $precioTotal = $precioTotal + $precioProducto;

        }

        return $precioTotal;
```

```
    }

    public function mostrarDetalleCarrito(){ //Muestra los datos del
carrito

        for ($x = 0; $x <= count($this->productos); $x++){

            $producto = $this->productos[$x];

            echo "Producto: ".$producto["nombre"]."\n".
            "Precio: ".$producto["precio"]."\n".
            "Cantidad: ".$producto["cantidad"]."\n";

        }
    }
}

$carrito = new Carrito;
$carrito->productos = array();

do{

    $opcion = readline("Elija una opción: ");

    if ($opcion == "añadir"){

        $nombre = readline("Introduzca el nombre del producto a añadir:
");
        $precio = readline("Introduzca el precio del producto a añadir:
");
        $cantidad = readline("Introduzca la cantidad del producto a
añadir: ");

        $carrito->agregarProducto($nombre, $precio, $cantidad);
        echo "Producto añadido correctamente"."\n";

    } elseif ($opcion == "quitar"){

        $nombre = readline("Introduzca el nombre del producto a eliminar:
");

        $carrito->quitarProducto($nombre);
        echo "Producto eliminado correctamente"."\n";

    } elseif ($opcion == "total"){

        echo "Precio total: ".$carrito->calcularTotal()." €"."\n";
```

```
    } elseif ($opcion == "detalles"){

        echo $carrito->mostrarDetalleCarrito()."\n";

    } elseif ($opcion == "salir"){

        break;

    } else{

        echo "Opción no válida"."\n";

    }
} while ($opcion != "salir")
?>
```

5. Juego de Rol

Creamos la clase personaje con las propiedades nombre, nivel, puntosVida y puntosAtaque y los métodos atacar, curarse y subir de nivel. Creamos varios personajes y escogemos uno con el que jugar. (Ejercicio incompleto)

```
<?php
class Personaje{

    public $nombre; //Nombre del personaje
    public $nivel; //Nivel del personaje
    public $puntosVida; //Vida del personaje
    public $puntosAtaque; //Ataque del personaje

    public function atacar(Personaje $objetivo){ //Dos personajes pelean

        $objetivo->puntosVida = $objetivo->puntosVida - $this-
>puntosAtaque;
        return $objetivo->nombre." ha sufrido ".$this->puntosAtaque."
puntos de daño"."\n";
    }

    public function curarse(){ //El personaje se cura

        $this->puntosVida = $this->puntosVida + 2 * $this->nivel;
        return $this->nombre." ha recuperado ".(2 * $this->nivel)."
puntos de vida"."\n";

    }

    public function subirNivel(){ //El personaje sube de nivel
```

```
        $this->nivel =$this->nivel + 1;
        $this->puntosAtaque = $this->puntosAtaque + 2;
        $this->puntosVida = $this->puntosVida + 2;
        echo $this->nombre." ha subido al nivel ".$this->nivel."\n";
        echo "El ataque de ".$this->nombre." ha aumentado en 2"."\n";
        echo "La vida de ".$this->nombre." ha aumentado en 2"."\n";

    }
}

$personaje1 = new Personaje;
$personaje1->nombre = "Ringabel";
$personaje1->nivel = 1;
$personaje1->puntosVida = 4;
$personaje1->puntosAtaque = 7;

$personaje2 = new Personaje;
$personaje2->nombre = "Erdrick";
$personaje2->nivel = 1;
$personaje2->puntosVida = 8;
$personaje2->puntosAtaque = 3;

$personaje3 = new Personaje;
$personaje3->nombre = "Cloud";
$personaje3->nivel = 1;
$personaje3->puntosVida = 5;
$personaje3->puntosAtaque = 5;

$personajes = array($personaje1, $personaje2, $personaje3);

$jugador = readline("Elija que personaje quiere controlar: ");

$objetivo = $personajes[]
?>
```

https://github.com/Rodrigo-Garcia-Ortiz/Programacion.git