UNIT 3 SOLVED EXERCISES. EXCEPTIONS AND ADVANCED PHP OOP

It Works

```
<body>
<?php
 $number=10;
$anverseNumber=1/$number;
echo "<h2>The inverse of $number is $anverseNumber</h2>";
?>
</body>
```

It doesn't work--ERROR

```
<body>
<?php
 $number=0;
 $anverseNumber=1/$number;
 echo "<h2>The inverse of $number is $anverseNumber</h2>";
?>
</body>
```

Fatal error: Uncaught DivisionByZeroError: Division by zero in C:\xampp1\htdocs\form.php:4 Stack trace: #0 {main} thrown in **C:\xampp1\htdocs\form.php** on line **4**

To handle the exception:

```
<?php
   $number=0;
   try {
     $anverseNumber=1/$number;
     echo "<h2>The inverse of $number is $anverseNumber</h2>";
   }
   catch(Throwable $t) {
         echo "An error happened";
   }
?>
```

Handle the exception with messages:

```
<?php
  $number=0;
  try {
     $anverseNumber=1/$number;
     echo "<h2>The inverse of $number is $anverseNumber</h2>";
  }
  catch(Throwable $t) {
     echo "An error {$t->getMessage()} happened<br/>";
     echo "In line {$t->getLine()} of file {$t->getFile()}<br/>";
  }
?>
```

Error not throwable

```
<?php
  function anverse($number) {
    $anverseNumber = 1 / $number;
    return $anverseumber;
  }
  try {
    $number=10;
    echo "<h2>The inverse of $number is ".anverse($number)."</h2>";
```

```
  }
  catch (Throwable $t) {
    echo "An error {$t->getMessage()} happened<br/>";
  }
?>
```

Warning: Undefined variable $anverseumber in C:\xampp1\htdocs\form.php on line 4

Transform warning into exception:

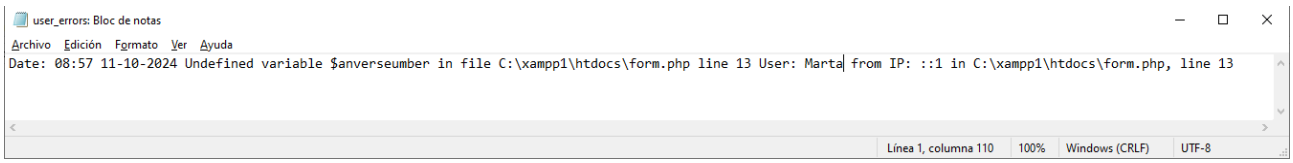
```
<?php
  function handlingErrors($eLevel, $eMessage, $eFile, $eLine) {
      throw new Exception("Error ".$eMessage." in line ".$eLine." of   ".
$eFile); // both warnings and exceptions will be thrown
  }
  function anverse($number) {
         $anverseNumber = 1 / $number;
      return $anverseumber;
  }
  set_error_handler("handlingErrors");
  try {
    $number=10;
    echo "<h2>The inverse of $number is ".anverse($number)."</h2>";
  }
  catch (Throwable $t) {
     echo "An error {$t->getMessage()} happened<br/>";
  }
  restore_error_handler();
?>
```

Same, but creating a log file:

```
<?php
  function handlingErrors($eLevel, $eMessage, $eFile, $eLine) {

      $newMessage = "Date: ".date("H:i d-m-Y ").$eMessage.
                             " in file ".$eFile." line ".$eLine.
                             " User: ".get_current_user()." from IP: ".
      $_SERVER['REMOTE_ADDR'];
      error_log("$newMessage in $eFile, line $eLine", 3,

            "c:/xampp/apache/logs/user_errors");
}
  function anverse($number) {
         $anverseNumber = 1 / $number;
      return $anverseumber;
  }
  set_error_handler("handlingErrors");
  try {
    $number=10;
    echo "<h2>The inverse of $number is ".anverse($number)."</h2>";
  }
  catch (Throwable $t) {
     echo "An error {$t->getMessage()} happened<br/>";
  }
  restore_error_handler();
?>
It creates a log file with the information:
```



1. Create a final class called Menu. It will allow to add all the options that we need. Show the menu horizontally or vertically (depending on which method we call).

```
<html>
<head>
<title></title>
</head>
<body>
<?php
final class Menu {

  private $links=array();
  private $titles=array();

 // Function to save links and options
  public function loadOption($li,$tit)
  {
    $this->links[]=$li;
    $this->titles[]=$tit;
  }

  //Show horizontally the menu
  public function showHorizontally()
  {
    for($f=0;$f<count($this->links);$f++)
    {
      echo '<a href="'.$this->links[$f].'">'.$this->titles[$f].'</a>';
      echo "-";
    }
  }

  //Show vertically the menu
  public function showVertically()
  {
    for($f=0;$f<count($this->links);$f++)
    {
      echo '<a href="'.$this->links[$f].'">'.$this->titles[$f].'</a>';
      echo "<br>";
    }
  }
}

//create menu

$menu1=new Menu();

$menu1->loadOption('https://cerosyunos.es/','cerosyunos');
$menu1->loadOption('http://about.me/sapoclay','About SapoClay');
$menu1->loadOption('https://innowise.com/es/php-development-
services/','Innowise');

//Show the menus
$menu1->showVertically();
echo "<hr>";
$menu1->showHorizontally();
?>
</body>
```

```
</html>
```

1.1. Now, we add a property color as a constant because we always want the same color in all the menus.

```
<html>
<head>
<title></title>
</head>
<body>
<?php
final class Menu {
 private $links=array();
 private $titles=array();
 const color="red";

// Function to save links and options
 public function loadOption($li,$tit)
 {
 $this->links[]=$li;
 $this->titles[]=$tit;
 }

 //Show horizontally the menu
 public function showHorizontally()
 {
  $color=self::color;

  for($f=0;$f<count($this->links);$f++)
  {
   echo '<a href="'.$this->links[$f].'" style="color:'.$color.';" >'.
$this->titles[$f].'</a>';
   echo "-";
  }
 }

 //Show vertically the menu
 public function showVertically()
 {
  $color=self::color;
  for($f=0;$f<count($this->links);$f++)
  {
   echo '<a href="'.$this->links[$f].'" style="color:'.$color.';">'.
$this->titles[$f].'</a>';
   echo "<br>";
  }
 }
}

//create menu: access the constant from outside

$menu1=new Menu();


$menu1->loadOption('https://cerosyunos.es/','cerosyunos');
$menu1->loadOption('http://about.me/sapoclay','About SapoClay');
$menu1->loadOption('https://innowise.com/es/php-development-
services/','Innowise');
```

```
//Show the menus
echo "We paint the links with ".Menu::color."<br/>";
$menu1->showVertically();
echo "<br>";
$menu1->showHorizontally();
?>
</p>
</body>
</html>
```

1.2. Create another menu with the same options, and compare the two instances.

```
$menu1=new Menu();

$menu1->loadOption('https://cerosyunos.es/','cerosyunos');
$menu1->loadOption('http://about.me/sapoclay','About SapoClay');
$menu1->loadOption('https://innowise.com/es/php-development-
services/','Innowise');

$menu2=new Menu();

$menu2->loadOption('https://cerosyunos.es/','cerosyunos');
$menu2->loadOption('http://about.me/sapoclay','About SapoClay');
$menu2->loadOption('https://innowise.com/es/php-development-
services/','Innowise');

if ($menu1==$menu2)
echo "Objects are equal";
else
echo "Objects are not equal";
```

1.3. Change an Option and compare again: **Objects are not equal**

1.4. Add an static method to show a line between the menus

```
static function showLine() {
  echo "<hr/>";
}

  ...

$menu1->showVertically();
$menu1->showLine();  or    Menu::showLine();
$menu1->showHorizontally();
```

1.5. Convert the constant with an static property, and change the color for all the instances

```
<html>
<head>
<title></title>
</head>
<body>
<?php
final class Menu {
```

```
 private $links=array();
 private $titles=array();
 static $color="red";



 static function showLine() {
  echo "<hr/>";

}
// Function to save links and options
 public function loadOption($li,$tit)
 {
  $this->links[]=$li;
  $this->titles[]=$tit;
 }

 //Show horizontally the menu
 public function showHorizontally()
 {
  //$color=self::color; - A constant without $
  $color=self::$color;

  for($f=0;$f<count($this->links);$f++)
  {
   echo '<a href="'.$this->links[$f].'" style="color:'.$color.';" >'.
$this->titles[$f].'</a>';
   echo "-";
  }
 }

 //Show vertically the menu
 public function showVertically()
 {
  $color=self::$color;
  for($f=0;$f<count($this->links);$f++)
  {
   echo '<a href="'.$this->links[$f].'" style="color:'.$color.';">'.
$this->titles[$f].'</a>';
   echo "<br>";
  }
 }
}

 //create menu: access the constant from outside

 $menu1=new Menu();

$menu1->loadOption('https://cerosyunos.es/','cerosyunos');
$menu1->loadOption('http://about.me/sapoclay','About SapoClay');
 $menu1->loadOption('https://innowise.com/es/php-development-
 services/','Innowise');

 $menu2=new Menu();

$menu2->loadOption('https://cerosyunos.es/','cerosyunos');
$menu2->loadOption('http://about.me/sapoclay','About SapoClay');
 $menu2->loadOption('https://innowise.com/es/php-development-
 services/','Innowise');

 //Show the menus
```

```
Menu::$color="green";
echo "menu1<br/>";
$menu1->showVertically();
echo "menu2<br/>";
$menu2->showVertically();

Menu::$color="blue";
echo "menu1<br/>";
$menu1->showHorizontally();
echo "<br/>menu2<br/>";
$menu2->showHorizontally();
?>
</body>
</html>
```

1.6. Try to extend the final class. What happens?

```
class intento extends Menu
{
  private $a;
}
```

```
Fatal error: Class intento cannot extend final class Menu
inC:\xampp1\htdocs\form.phpon line48
```

2. Class that implements “multiple” inheritance: from class Communication you can use methods from both traits: sayHello from trait greet and sayBye from saygoodbye trait

```
<?php
trait greet {
  function sayHello(){
    return "hello";
  }
}

trait saygoodbye {
  function sayBye(){
    return "bye";
  }
}

class Communication {
  use greet, saygoodbye;
}

$communication = new Communication;
echo $communication->sayHello() . ", how are you?. " .
$communication->sayBye();
?>
```

2.1. We can use traits inside other traits

```
trait greetandbye{
```

```
    use greet,saygoodbye;
}

class Communication {
    use greetandbye;
}
$communication = new Communication;
echo $communication->sayHello() . ", how are you?. " . $communication-
>sayBye();
```

3. Create an interface Animal, and a Class Cat which implements the Animal interface:

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

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

$animal = new Cat();
$animal->makeSound();
?>
```

3.1. Multiple class definitions of an interface

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

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

class Dog implements Animal {
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();
}
?>
```