3rd Level	Web Programming
Faculty of Computers and Information	Sheet # 6
	27-3-2024

Please answer the following questions: PLEASE DON’T SUBMIT THE ANSWERS

<u>Question 1:</u>

Please correct the errors in the following scripts "if any" then give the HTML output as well as the output on the browser with and without using " error_reporting(E_ALL); " you have to understand every instruction”.

a)

```
<?php
function fun1($string)
{
  $a=$string;
  while ($a = strstr($a, '&'))
  {
   echo "'".$a."'\n";
   $b=strstr($a, ';');
   if (!$b)
   {
     echo "exam...\n";
     $nb=strlen($a);
     return substr($string, 0, strlen($string)-$nb);
    }
    $a=substr($a,1,strlen($a)-1);
   }
    return $string;
}
$string="this is ;<b> &no_bold </b>";
print (fun1($string) . "<br>");
print ($string. "<br>");
echo strspn($string, "tis");
?>
```

b)

```
<?php
function myfun($input, $b, $a)
{
  $substr=substr($input,strlen($b)+strpos($input,$b),(strlen($input)-strpos($input, $a))*(-1));
  return $substr;
}
$string = "1212129123945126789";
$a = "12"; $b = "9";
echo myfun($string, $a, $b);
?>
```

c)

```
<?php
error_reporting(E_ALL);
function mb_string($string1, $string2, $minChars = 5)
{
    $string1=trim($string1);
    //trim returns a string with whitespace stripped from the beginning and end of the string
    $string2 = trim($string2);
    $length1 = strlen($string1);
    $length2 = strlen($string2);
    if ($length1 > $length2) {
        $string3 = $string1; $string1 = $string2;
        $string2 = $string3; $length3 = $length1;
        $length1 = $length2; $length2 = $length3;
        unset($string3); unset($length3);
    }
    if ($length2 > 255) { return null; }
    for ($s = $length1; $s >= $minChars; --$s) {
        for ($i = 0, $ix = $length1 - $s; $i <= $ix; ++$i) {
            $substring1 = substr($string1, $i, $s);
            $found = strpos($string2, $substring1);
            if ($found !== false) {
                return trim(substr($string2, $found, strlen($substring1)));
            }
        }
    }
    return null;
}
echo mb_string("  hello friend  ", "  hey friend how are you?  ");
?>
```

d)

```
<?php
$html = "<b>bold text</b><a href=howdy.html>click me</a>";
preg_match_all("#(<(.*?)>)(.*?)(</(.?)>)#", $html, $matches);
foreach ($matches as $val)
{
  echo "First: " . $val[0]."<br>";
  echo "Second: " . $val[1]."<br><br>";
}
?>
```

e)

```
<?php
function stringExpand($subject, array $vars)
{
 foreach ($vars as $name => $value)
 {
  $subject=preg_replace(sprintf('/\$\{?%s\}?/',$name),$value,$subject);
 }
    return $subject;
}
$a=array(1,2,3);
$subject='${1},$2,{$3}';
echo stringExpand($subject,$a);
?>
```

f)

```
<?php
class Test{
 const MIN = 0, MAX = 100; protected $person, $score;
 public function __construct($p, $s)
 {
   $this->person = $p; $this->score = clamp($s);
 }
 protected static function clamp($val)
 {
   if($val < Test::MIN) $val = Test::MIN;
   if($val > Test::MAX) $val = Test::MAX;
   return $val;
 }
 public function __toString()
 {
  return "Test [person=$this->person, score=$this->score%]";
 }
}
$obj = new Test("John Doe", 83); echo (object)$obj;
?>
```

g)

```
<?php
class Template
{
 const OPEN_BRACKET = "{";
 const CLOSE_BRACKET = "}";
 public static function inject(array $source, $template)
 {
  $ob_size = strlen(Template::OPEN_BRACKET);
  $cb_size = strlen(Template::CLOSE_BRACKET);
  $pos = 0; $end = strlen($template);
  while($pos <= $end)
  {
   if($pos_1 = strpos($template, Template::OPEN_BRACKET, $pos))
   {
    if($pos_1)
    {
       $pos_2 = strpos($template, Template::CLOSE_BRACKET, $pos_1);
       if($pos_2)
       {
         $return_length = ($pos_2-$cb_size) - $pos_1;
         $var = substr($template, $pos_1+$ob_size, $return_length);
         $template=str_replace(Template::OPEN_BRACKET.$var.Template::CLOSE_BRACKET,
         $source[$var], $template);
         $pos = $pos_2 + $cb_size;
       }
       else{
            throw new exception("Incorrectly formed template");
            break;
           }
     }
   }
   else { break;}
  }
  return $template;
 }
}
$array = array("NAME" => "NAME", "DOB"=> "DOB", "ACL" =>"ACL?");
$template = "?{NAME}. {DOB} . {ACL} ";
echo Template::inject($array, $template);
?>
```

h)

```
<?php
function __autoload($class_name)
{
 require_once $class_name . '.php';
}
$anObject = new ClassName();
array_walk_recursive($anObject->getFruits(),array($anObject, 'test_print'));
echo";;;\n\n";
$anObject->arrWalker($anObject->getFruits());
?>
```

Modify the following part if needed and give name to the file that includes it to operate correctly with the previous code

```
<?php
class ClassName {
   private $sweet = array(); private $fruits = array();
   public function __construct()
   {
      $this->sweet = array('a' => 'apple', 'b' => 'banana');
      $this->fruits = array('sweet' => $this->sweet, 'sour' => 'lemon');
   }
   public function getFruits()
   {
      return $this->fruits;
   }
   public function test_print($item, $key)
   {
     echo "$key-$item\n";
   }
   public function arrWalker($arr)
   {
       array_walk_recursive($arr, array($this, 'test_print'));
   }
}
?>
```