PRACTICA 15

alpe FORMACIÓN

PHP

FUNCIONES E INSTRUCCIONES DE CONTROL

php

Por Ramon Abramo

EJERCICIO 1

Complete la función para que encuentre la media de las tres puntuaciones que se le pasaron y devuelva el valor de la letra asociada con esa calificación.

score	Grado
90 <= score <= 100	'A'
80 <= score < 90	'B'
70 <= score < 80	'C'
60 <= score < 70	'D'
0 <= score < 60	'F'

```php
// Complete la función para que encuentre la media de las tres
// puntuaciones que se le pasaron y devuelva el valor de la letra asociada con esa calificación.
// 90 <= score <= 100 'A'
// 80 <= score < 90 'B'
// 70 <= score < 80 'C'
// 60 <= score < 70 'D'
// 0 <= score < 60  'F'
function getGrade($a, $b, $c) {
    return;
}

echo getGrade(1, 3, 4);
```

SOLUCIÓN

```php
function getGrade($a, $b, $c) {
    $valor = ($a + $b + $c) / 3;
    if ($valor < 60) {
        $resultado = 'F';
    } else if ($valor < 70) {
        $resultado = 'D';
    } else if ($valor < 80) {
        $resultado = 'C';
    } else if ($valor < 90) {
        $resultado = 'B';
    } else if ($valor <= 100) {
        $resultado = 'A';
    }

    return $resultado;
}
```

SOLUCION 1

```php
function getGrade($a, $b, $c) {
   $tmp = ($a + $b + $c) / 3;
   if ($tmp >= 90)
       return 'A';
   if ($tmp >= 80)
       return 'B';
   if ($tmp >= 70)
       return 'C';
   if ($tmp >= 60)
       return 'D';
   else
       return 'F';
}
```

SOLUCION 2

```php
function getGrade(...$a) {
    $avr = array_sum($a) / count
($a);if ($avr >= 90)
        return 'A';
    if ($avr >= 80)
        return 'B';
    if ($avr >= 70)
        return 'C';
    if ($avr >= 60)
        return 'D';
    return 'F';
}
```

SOLUCION 3

```php
function getGrade($a, $b, $c) {
    $avg = ($a + $b + $c) / 3;
    switch (true) {
        case $avg < 60:
            return "F";
        case $avg < 70:
            return "D";
        case $avg < 80:
            return "C";
        case $avg < 90:
            return "B";

    }   return "A";
    }
```

SOLUCION 4

```php
function getGrade($a, $b, $c) {

    $mean = (int) (($a + $b + $c) / 3);
    $grade = str_split('ABCDF');
    $result = (ceil((100 - $mean) / 10)) > 5 ? 5 : (ceil((100 - $mean) / 10));

    return $result == 0 ? $grade[$result] : $grade[$result - 1];
}
```

SOLUCION 5

```php
function getGrade($a, $b, $c) {
    return [
        100 => 'A',
        90 => 'A',
        80 => 'B',
        70 => 'C',
        60 => 'D',
        50 => 'F',
        40 => 'F',
        30 => 'F',
        20 => 'F',
        10 => 'F',
        0 => 'F'
            ][floor(( ( $a + $b + $c ) / 3 ) / 10) * 10];
}
```

SOLUCION 6

```php
function getGrade($a, $b, $c) {
    $average = ($a + $b + $c) / 3;
    if ($average >= 90 && $average <= 100) {
      return 'A';
    } else if ($average >= 80 && $average < 90) {
      return 'B';
    } else if ($average >= 70 && $average < 80) {
      return 'C';
    } else if ($average >= 60 && $average < 70) {
      return 'D';
    } else if ($average >= 0 && $average < 60) {
      return 'F';
    }
  }
```

SOLUCION 7

```php
function getGrade($a, $b, $c) {
    $x = ($a + $b + $c) / 3;

    switch (true) {
        case ($x >= 90 && $x <= 100) : return 'A';
        case ($x >= 80 && $x < 90 ) : return 'B';
        case ($x >= 70 && $x < 80 ) : return 'C';
        case ($x >= 60 && $x < 70 ) : return 'D';
        case ($x >= 0 && $x < 60 ) : return 'F';
    }
}
```

SOLUCION 8

```php
function getGrade(int $a, int $b, int $c): string {
    $mean = round(($a + $b + $c) / 3);

    $grades = [
        ['A', 90 <= $mean && $mean <= 100],
        ['B', 80 <= $mean && $mean < 90],
        ['C', 70 <= $mean && $mean < 80],
        ['D', 60 <= $mean && $mean < 70],
        ['F', 0 <= $mean && $mean < 60],
    ];

    return current(array_filter($grades, function ($grade) {
                        return next($grade);
                    }))[0];
}
```

SOLUCION 9

```php
function getGrade($a, $b, $c) {
    $score = ($a + $b + $c) / 3;
    return match (true) {
        $score >= 90 => 'A',
        $score >= 80 => 'B',
        $score >= 70 => 'C',
        $score >= 60 => 'D',
        default => 'F',
    };
}
```

SOLUCION 10

```php
function getGrade($a, $b, $c) {
    $sum = array_sum([$a, $b, $c]);
    $sum = $sum / 3;
    if ($sum >= 90) {
        return 'A';
    } elseif ($sum >= 80) {
        return 'B';
    } elseif ($sum >= 70) {
        return 'C';
    } elseif ($sum >= 60) {
        return 'D';
    } else {
        return 'F';
    }
}
```

SOLUCION 11

```php
function getGrade(int $a, int $b, int $c): string {
    switch (intdiv($a + $b + $c, 30)) {
        case 10: return 'A';
        case 9: return 'A';
        case 8: return 'B';
        case 7: return 'C';
        case 6: return 'D';
        default: return 'F';
    }
}
```

SOLUCION 12

```php
function getGrade($a, $b, $c) {

    $avg = ($a + $b + $c) / 3;

    switch (true) {
        case $avg < 60:
            return "F";
            break;

        case $avg < 70:
            return "D";
            break;

        case $avg < 80:
            return "C";
            break;

        case $avg < 90:
            return "B";
            break;

        default:
            return "A";
    }
}
```

EJERCICIO 2

Función que elimina todos los signos de exclamación de una cadena dada.

```
Función que elimina todos los signos de exclamación de una cadena dada.

function remove_exclamation_marks($string) {
    return ;
}

echo remove_exclamation_marks("Ejemplo!de!clase");
```

SOLUCIÓN

```php
function remove_exclamation_marks($string) {
    return str_replace('!', '', $string);
}
```

SOLUCION 1

```php
function remove_exclamation_marks($string) {
    return str_replace(['!', '¡'], '', $string);
}
```

SOLUCION 2

```php
function remove_exclamation_marks($string) {
    return implode('', explode('!', $string));
}
```

SOLUCION 3

```php
function remove_exclamation_marks($string) {
    return preg_replace('/[!]+?/', '', $string);
}
```

SOLUCION 4

```php
function remove_exclamation_marks($string) {
    $array = str_split($string);

    $result = "";

    foreach ($array as $val) {
        if ($val != "!") {
            $result .= $val;
        }
    }

    return $result;
}
```

SOLUCION 5

```php
function remove_exclamation_marks($string) {
    return strtr($string, ['!' => '']);
}
```

SOLUCION 6

```php
function remove_exclamation_marks($string) {
    $letters = array();

    for ($i = 0; $i < strlen($string); $i++) {
        $letters[] = $string[$i];
    }

    $new = "";

    for ($i = 0; $i < count($letters); $i++) {
        switch ($letters[$i]) {
            case "!":
                $letters[$i] = "";
                break;
            default:
                continue;
                break;
        }
        $new .= $letters[$i];
    }

    return $new;
}
```

SOLUCION 7

```php
function remove_exclamation_marks($string) {
    $result = [];
    for ($i = 0; $i < strlen($string); $i++) {
        if ($string[$i] !== '!') {
            $result[] = $string[$i];
        }
    }
    return implode('', $result);
}
```

SOLUCION 8

```php
function remove_exclamation_marks($s) {
    if (strlen($s) === 0) {
        return "";
    }
    if ($s[0] === "!") {
        return remove_exclamation_marks(substr($s, 1));
    }
    return $s[0] . remove_exclamation_marks(substr($s, 1));
}
```

EJERCICIO 3

Se le dará una matriz a y un valor x. Todo lo que necesita hacer es verificar si la matriz proporcionada contiene el valor.

Array puede contener números o cadenas. X puede ser cualquiera.

Devuelve verdadero si la matriz contiene el valor, falso en caso contrario.

```php
Se le dará una matriz a y un valor x. Todo lo que necesita hacer es verificar si la matriz proporcionada contiene el valor.
Array puede contener números o cadenas. X puede ser cualquiera.
Devuelve verdadero si la matriz contiene el valor, falso en caso contrario.

function solution($a, $x) {
    return ;
}
echo solution(3,[2,3,6,2,3]);
```

SOLUCION

```php
function solution($a, $x) {
    return in_array($x, $a);
}
```

SOLUCION 1

```php
function solution(array $a, $x): bool {
    return in_array($x, $a);
}
```

SOLUCION 2

```php
function solution($a, $x) {
    if (array_search($x, $a) === false) {
        return false;
    }
    return true;
}
```

SOLUCION 3

```php
function solution($a, $x) {
    foreach ($a as $b) {
        if ($b == $x) {
            return true;
        }
    }
    return false;
}
```

SOLUCION 4

```php
function solution($a, $x) {

    return array_search($x, $a) === false ? false : true;
}
```

EJERCICIO 4

Elimine los espacios de la cadena y luego devuelva la cadena resultante.

```
Elimine los espacios de la cadena y luego devuelva la cadena resultante.

function no_space(string $s): string {
  return ;
}
echo no_space(" E ejmplo");
```

SOLUCIÓN

```
function no_space(string $s): string {
    return str_replace(" ", "", $s);
}
```

SOLUCION 1

```
function no_space(string $s): string {
    $result = '';
    for ($i = 0; $i < strlen($s); $i++) {
        if ($s[$i] !== ' ') {
            $result .= $s[$i];
            continue;
        }
    }
    return $result;
}
```

SOLUCION 2

```php
function no_space(string $s): string {

    $ex = explode(" ", $s);

    $im = implode($ex);

    return $im;
}
```

SOLUCION 3

```php
function no_space(string $s): string {
    $word = '';
    foreach (str_split($s) as $char)
        if ($char !== ' ')
            $word .= $char;

    return $word;
}
```

SOLUCION 4

```php
function no_space(string $s): string {
    $arr = str_split($s);
    $result_arr = array_filter($arr, function ($index) {
        return $index != ' ';
    });
    return implode('', $result_arr);
}
```

SOLUCION 5

```php
function no_space(string $s): string {
    return preg_replace('/\s+/', '', $s);
}
```

EJERCICIO 5

Completa la función de suma de cuadrados para que eleve al cuadrado cada número que se le pasa y luego suma los resultados.

Por ejemplo, para [1, 2, 2] debería devolver 1^2 + 2^2 + 2^2 = 9.

```
Completa la función de suma de cuadrados para que eleve al cuadrado
cada número que se le pasa y luego suma los resultados.
Por ejemplo, para [1, 2, 2] deberia devolver 1^2 + 2^2 + 2^2 = 9.

function square_sum($numbers) : int {
  return ;
}

echo square_sum([1,2,2]);
```

SOLUCIÓN

```php
function square_sum($numbers): int {
    $salida = array_map(function ($valor) {
        return $valor ** 2;
    }, $numbers);
    return array_sum($salida);
}
```

SOLUCION 1

```php
function square_sum($n): int {

    return array_sum(array_map(function ($v) {
                return $v * $v;
            }, $n));
}
```

SOLUCION 2

```php
function square_sum(array $numbers): int {
    $sum = 0;
    foreach ($numbers as $number) {
        $sum += $number * $number;
    }

    return $sum;
}
```

SOLUCION 3

```php
function square_sum($numbers): int {
    return array_reduce($numbers, function ($sq_sum, $item) {
        $sq_sum += pow($item, 2);
        return $sq_sum;
    }, 0);
}
```

SOLUCION 4

```php
function square_sum($numbers): int {
    $sum = 0;

    foreach ($numbers as $item) {
        $sum += pow($item, 2);
    }

    return $sum;
}
```

EJERCICIO 6

Escriba una función que tome una cadena de una o más palabras y devuelva la misma cadena, pero con las palabras de cinco o más letras invertidas. Las cadenas pasadas consistirán solo en letras y espacios. Los espacios se incluirán solo cuando haya más de una palabra presente.

```
    Escriba una función que tome una cadena de una o más palabras y devuelva la misma cadena,
pero con las palabras de cinco o más letras invertidas.
Las cadenas pasadas consistirán solo en letras y espacios.
Los espacios se incluirán solo cuando haya más de una palabra presente.

function spinWords(string $str): string {
  return ;
}

echo spinWords("Esto es un test dificil"); //devuelve "Esto es un test licifdi"
```

SOLUCIÓN

```php
function spinWords(string $str): string {
    return implode(" ", array_map(function ($palabra) {
                if (strlen($palabra) >= 5) {
                    return strrev($palabra);
                }
                return $palabra;
            }, explode(" ", $str)));
}
```

SOLUCION 1

```php
function spinWords(string $str): string {
    $boom = explode(' ', $str);

    foreach ($boom as &$word) {
        if (strlen($word) >= 5) {
            $word = strrev($word);
        }
    }

    return implode(' ', $boom);
}
```

SOLUCION 2

```php
function spinWords(string $str): string {
    $words = explode(' ', $str);
    array_walk($words, function (&$value) {
        $value = strlen($value) >= 5 ? strrev($value) : $value;
    });
    return implode(' ', $words);
}
```

SOLUCION 3

```php
function spinWords(string $str): string {
    $a = 0;
    $b = 0;
    $c = 0;
    $arr = [];
    $string = ' ';
    if (strlen($str) >= 5) {
        for ($j = 0, $i = 0; $j < strlen($str); $j++) {
            if (substr($str, $j, 1) === ' ') {
                $b = $j - $c;
                $arr[$i] = substr($str, $a, $b);
                $c = $j + 1;
                $i++;
                $a = $j + 1;
            }
        }
        if ($c < strlen($str)) {
            $arr[] = substr($str, $c, strlen($str) - $c);
        }
        for ($i = 0; $i <= sizeof($arr); $i++) {
            if (strlen($arr[$i]) >= 5) {
                $arr[$i] = strrev($arr[$i]);
            }
        }
    } else {
        return $str;
    }

    return implode(' ', $arr);
}
```

SOLUCION 4

```php
function spinWords(string $str): string {
    return preg_replace_callback('/\w{5,}/', function ($matches) {
        return strrev($matches[0]);
    }, $str);
}
```

EJERCICIO 7

Un Número Narcisista es un número que es la suma de sus propios dígitos, cada uno elevado a la potencia del número de dígitos en una base dada.

Nos limitaremos al decimal (base 10).

Por ejemplo,

153 (3 dígitos): 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153

1634 (4 dígitos): 1^4 + 6^4 + 3^4 + 4^ 4 = 1 + 1296 + 81 + 256 = 1634

Su código debe devolver verdadero o falso dependiendo de si el número dado es un número narcisista en base 10.

No se requiere la verificación de errores para cadenas de texto u otras entradas no válidas, solo se pasarán enteros válidos a la función.

```
    Un Número Narcisista es un número que es la suma de sus propios dígitos,
cada uno elevado a la potencia del número de dígitos en una base dada.
Nos limitaremos al decimal (base 10).
Por ejemplo,
153 (3 dígitos): 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153
1634 (4 dígitos): 1^4 + 6^4 + 3^4 + 4^ 4 = 1 + 1296 + 81 + 256 = 1634

Su código debe devolver verdadero o falso dependiendo de si el número dado es un número narcisista en base 10.

No se requiere la verificación de errores para cadenas de texto u otras entradas no válidas, solo se pasarán enteros válidos a la función.

function narcissistic(int $value): bool {
  return ;
}

echo narcissistic(456);
```

SOLUCIÓN

```php
function narcissistic(int $value): bool {
    $texto = str_split(strval($value));
    return ($value == array_sum(array_map(function ($digito)use ($texto) {
                        return pow($digito, count($texto));
                    }, $texto)));
}
```

SOLUCION 1

```php
function narcissistic(int $value): bool {
    $len = strlen($value);
    $sum = 0;

    foreach (str_split($value) as $num) {
        $sum += pow($num, $len);
    }

    return ($sum == $value);
}
```

SOLUCION 2

```php
function narcissistic(int $value): bool {
    $ints = str_split($value);
    $count = count($ints);
    $result = 0;
    foreach ($ints as $int) {
        $result += pow($int, $count);
    }
    return $result == $value;
}
```

SOLUCION 3

```php
function narcissistic(int $value): bool {
    $total = array_reduce(str_split($value), function ($carry, $item) use ($value) {
        $carry += pow($item, strlen($value));
        return $carry;
    }, 0);
    return $total === $value;
}
```

SOLUCION 4

```php
function narcissistic(int $value): bool {
    $length = strlen((string) $value);

    for ($i = 0; $i < $length; $i++) {
        $numbers[$i] = (int) substr((string) $value, $i, 1);
        $numbers[$i] = pow($numbers[$i], $length);
    }

    $sum = 0;

    foreach ($numbers as $number) {
        $sum = $sum + $number;
    }

    if ($sum == $value) {
        return true;
    }

    return false;
}
```