UNIVERSITY of GREENWICH

Alliance with FPT Education

PHP Associate Arrays and Functions- Lab

This document defines a set of tasks to be done as a part of the PHP Associate Array and Functions lecture's exercises.

Part I: Associative Arrays

Counting characters in text

Write a program that reads some text and counts the occurrences of each character in it. Print the results.

Examples

Input	Output
`Hello`	`["h" => "1","e" => "1","l" => "2","o" => "1"]`
`Learning PHP is FUN!`	`["l" => "1","e" => "1","a" => "1","r" => "1","n" => "3","i" => "2","g" => "1","p" => "2","h" => "1","s" => "1","f" => "1","u" => "1","!" => "1"]`

Sum by Town

Read towns and incomes (like shown below) and print an array holding the total income for each town (see below). Print the towns in their natural order as object properties.

Examples

Input	Output
`Sofia, 20, Varna, 10, Sofia, 5`	`["Sofia" => "25","Varna" => "7"]`
`Plovdiv, 40, Pernik, 20, Vidin, 8, Sliven, 44, Plovdiv, 1, Vidin, 7, Chirpan, 0`	`["Plovdiv" => "41","Pernik" => "20","Vidin" => "15","Sliven" => "44","Chirpan" => "0"]`

Periodic Table

You are given an n number of chemical elements. You need to keep track of all elements and at the end print all **unique ones**.

Examples

Input	Output
`Ce, O, Mo, O, Ce, Ee, Mo`	`Ee`
`Ge, Ch, O, Ne, Nb, Mo, Tc, O, Ne`	`Ge Ch Nb Mo`

Part II: Functions

Exercises: Functions

Palindrome

Write a function **isPalindrome** to check a string for symmetry.

Examples

Input	Output
abcccba	true
xyz	false

Day of Week

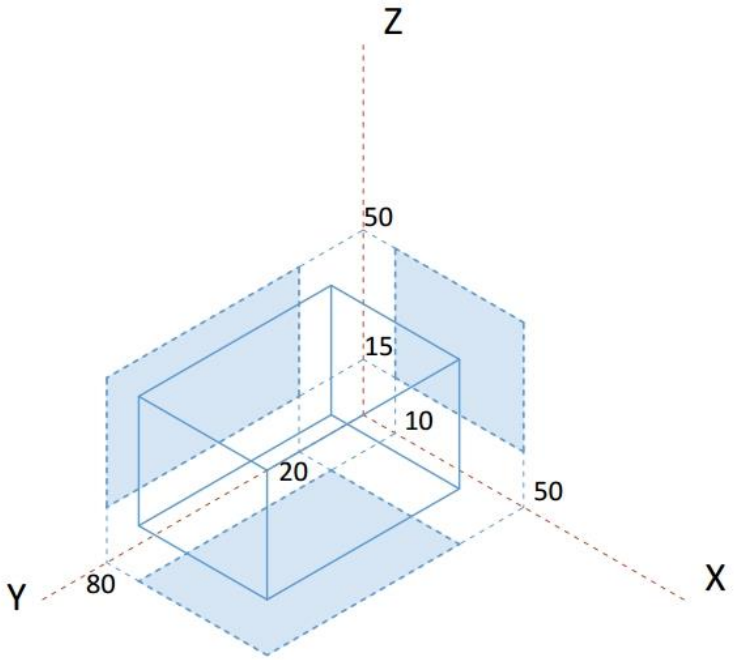
Write a function to return the day number by day of week in text.

Examples

Input	Output
Monday	1
Sunday	7

Inside Volume

Write a function that determines whether a point is inside the volume, defined by the box, shown on the right.



The **input** comes as a string representing the coordinates that needs to be split and parsed as numbers. Each set of 3 elements are the x, y and z coordinates of a point.

The **output** should be printed to the console on a new line for each point. Print **`inside`** if the point lies inside the volume and **`outisde`** otherwise.

Examples

Input	Output
8, 20, 22	outside
13.1, 50, 31.5,	inside

50, 80, 50, -5, 18, 43	inside outsid e

Hints

This task is very similar to previous assignments where a point might lie inside an area in 2D space, with just an extra dimension added. If we look at a classic conditional statement, which checks whether a point is inside a rectangle:

```
if ($x >= $x1 && $x <= $x2) {
     if ($y >= $y1 && $y <= $y2) {
         return true;
     }
}
```

It checks whether a coordinate is greater than the minimum and at the same time less than the maximum bounding value for both axes (x and y). All we have to do is to include an additional check for a coordinate to be within the vertical limits of the volume (z-axis):

```
if ($x >= $x1 && $x <= $x2) {
     if ($y >= $y1 && $y <= $y2) {
             if ($z >= $z1 && $z <= $z2) {
             return true;
             }
     }
}
```

We can then wrap this whole statement in a function and as we process each set of coordinates, pass them to see if they are inside the volume and print the correct message to the console. Since the volume is the same every time, we can hardcode the values, but it's generally good practice to pass them as function arguments, so that the function may work with any arbitrary volume. Later in the course we'll learn how to shorten this with the use of objects.

```php
function isVolume($x, $y, $z)
{
    $x1 = 10; $x2 = 50;
    $y1 = 20; $y2 = 80;
    $z1 = 15; $z2 = 50;

    if ($x >= $x1 && $x <= $x2) {
        if ($y >= $y1 && $y <= $y2) {
            if ($z >= $z1 && $z <= $z2) {
                return true;
            }
        }
    }

    return false;
}
```

We can extract the sets of coordinates with a loop that skips 3 elements at a time and assigns them to temporary variables:

```php
$inputNum = count($input);
  for ($i = 0; $i < $inputNum; $i += 3) {
      $x = $input[$i];
      $y = $input[$i + 1];
      $z = $input[$i + 2];

      if (isVolume($x, $y, $z)) {
          echo 'inside';
      } else {
          echo 'outside';
      }
  }
```

We know from the problem description that the input array will contain sets to three coordinates. Starting at 0, the current element (denoted by index ***i*** inside the loop) is the x-coordinate, the element after the current (***i + 1***) is the y-coordinate, and the element two indices after the current (***i + 2***) is the z-coordinate. At the end of the cycle, the index is increased by 3 and we can obtain the coordinates of the next point, using the same arithmetic (instead of 0, 1 and 2 we will get 3, 4 and 5) and so on, until there are no more elements in the array. The three coordinates are passed into our function and we get a Boolean value as a result. If it's true, we print **`inside`** for the current point and otherwise we print **`outside`**.

Road Radar

Write a function that determines whether a driver is within the speed limit. You will receive his speed and the area where he's driving. Each area has a different limit: on the **motorway** the limit

is **130** km/h, on the **interstate** the limit is **90**, inside a **city** the limit is **50** and within a **residential** area the limit is **20** km/h. If the driver is within the limits, your function prints nothing. If he's over the limit however, your function prints the severity of the infraction. For speeds up to **20** km/h over the limit, he's speeding; for speeds up to **40** over the limit, the infraction is **excessive speeding** and for anything else, **reckless driving**.

The **input** comes in two rows. On the first row you will receive the current speed as a string and needs to be parsed as a number. On the second row you will be given the second element which is the area.

The **output** should be printed to the console. Note in certain cases there will be no output.

Examples

Input	Output
40 city	
20 residential	speeding
120 interstate	excessive speeding
200 motorway	reckless driving

Hints

We can divide the task in two functions – one that determines what the current speed limit is, depending on zone, and another which tells us if an infraction is being made, depending on current speed and current limit. Determining the limit is achieved with a **switch** statement on the input:

```php
function getLimit($zone) {
    switch ($zone) {
        case 'motorway':
            $speedLimit = 130;
            break;
        case 'interstate':
            $speedLimit = 90;
            break;
        case 'city':
            $speedLimit = 50;
            break;
        case 'residential:';
            $speedLimit = 20;
            break;
        default:
            echo "Not a Valid Input";
            $speedLimit = 1000;
    }
    return $speedLimit;
}
```

This function takes a string as an argument and returns a number, depending on what that string is. We can take this directly from the input, pass it to this function and save the return value in a variable. In our second function, we pass the current speed and the limit, which we just saved.

```php
function getInfraction($speed, $limit) {
    $overSpeed = $speed - $limit;
    if ($overSpeed <= 0) {
        $result = false;
    } else {
        //TODO:
    }
    return $result;
}
```

We calculate the difference between the current speed and the limit – if it's negative or zero, this means the driver is within the rules and we return **`false`**, and in any other case, return the infraction as a string and store the result of the operation in a variable.

```
$limit = getLimit($zone);
$infraction = getInfraction($speed, $limit);
$overSpeed = $speed-$limit;
if ($infraction) {
    //TODO:
}
```

UNIVERSITY of GREENWICH

Alliance with FPT Education

MÃ SINH VIÊN