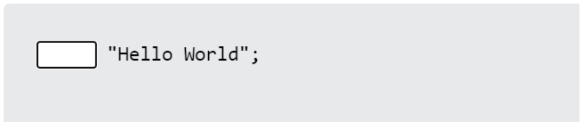
PHP syntax:

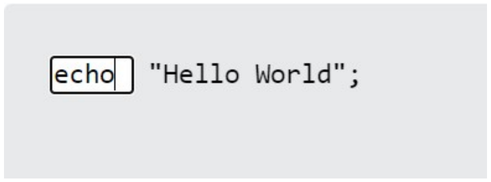
Exercise 1:

Insert the missing part of the code below to output "Hello World".



```
"Hello World";
```

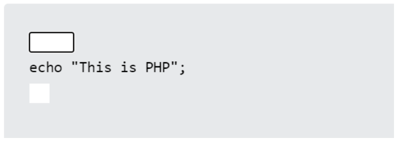
Solution:



```
echo "Hello World";
```

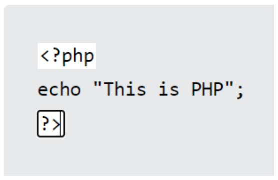
Exercise 2:

Write the correct opening tag and close tag for PHP scripts.



```
echo "This is PHP";
```

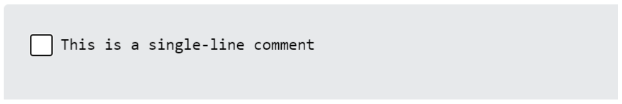
Solution:



```
<?php
echo "This is PHP";
?>
```

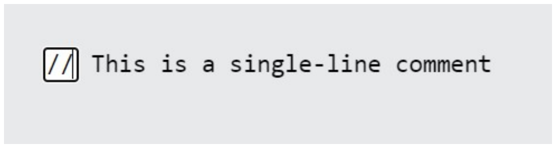
Exercise 3:

Single-line comments in PHP can be written using two different prefixes, write one of them.



```
This is a single-line comment
```

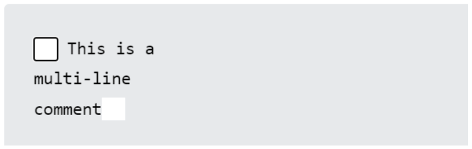
Solution:



```
// This is a single-line comment
```

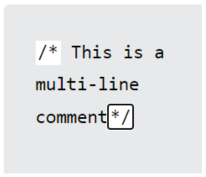
Exercise 4:

Insert the correct characters to write a multi-line comment.



```
[ ] This is a
multi-line
comment[ ]
```

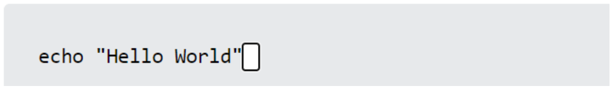
Solution:



```
/* This is a
multi-line
comment*/
```

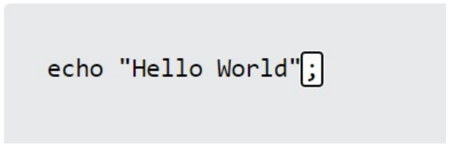
Exercise 5:

Statements in PHP have to end with a special character, which one?



```
echo "Hello World"[ ]
```

Solution:

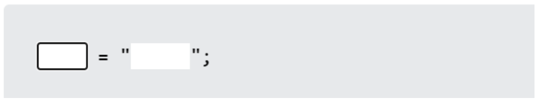


```
echo "Hello World";
```

PHP variables:

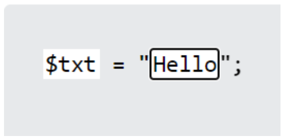
Exercise 1:

Create a variable named `txt` and assign the value `"Hello"`.



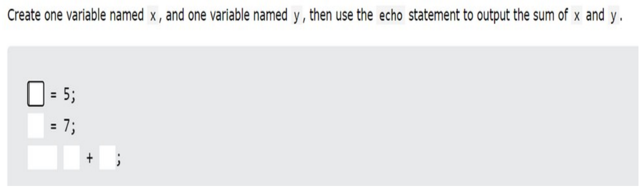
```
[ ] = "[ ]";
```

Solution:



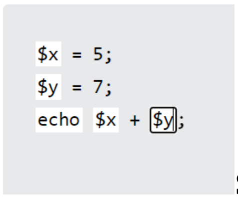
```
$txt = "Hello";
```

Exercise 2:

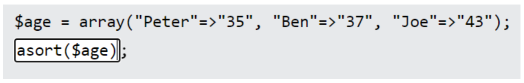

Create one variable named x, and one variable named y, then use the echo statement to output the sum of x and y.

```
 = 5;
 = 7;
   + ;
```

Solution:

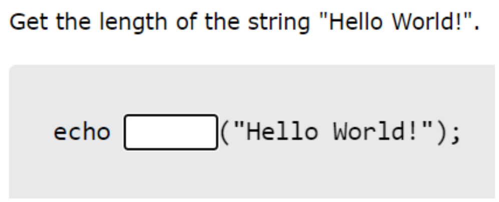

```
$x = 5;
$y = 7;
echo $x + $y;
```

Solution:


```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
asort($age);
```

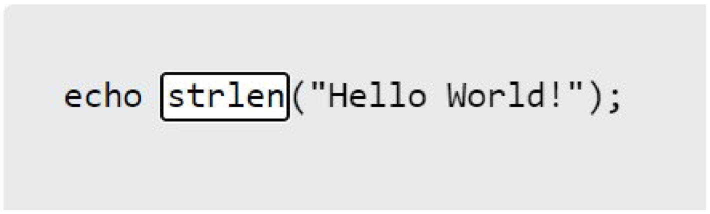
PHP strings:

Exercise 1:

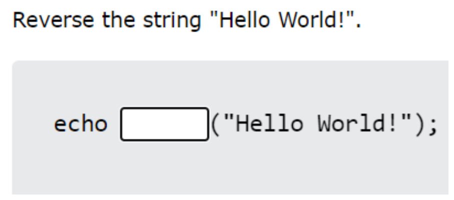

Get the length of the string "Hello World!".

```
echo ("Hello World!");
```

Solution:

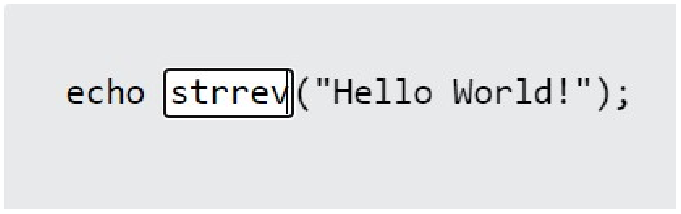

```
echo strlen("Hello World!");
```

Exercise 2:

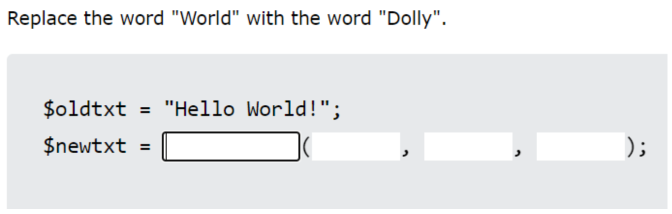

Reverse the string "Hello World!".

```
echo       ("Hello World!");
```

Solution:

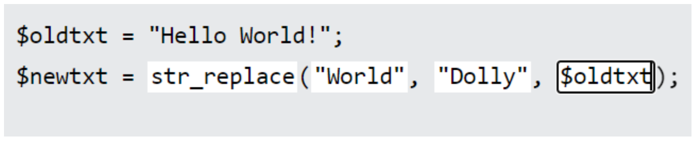

```
echo strrev("Hello World!");
```

Exercise 3:


Replace the word "World" with the word "Dolly".

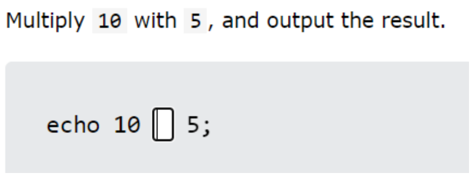
```
$oldtxt = "Hello World!";
$newtxt =          (      ,      ,      );
```

Solution:


```
$oldtxt = "Hello World!";
$newtxt = str_replace("World", "Dolly", $oldtxt);
```

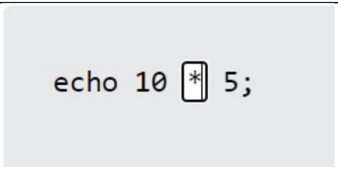
PHP operators:

Exercise 1:


Multiply `10` with `5`, and output the result.

```
echo 10   5;
```

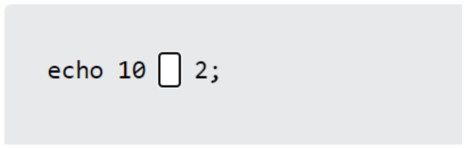
Solution:



```
echo 10 * 5;
```

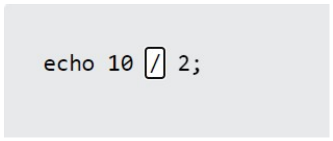
Exercise 2:

Divide `10` by `2`, and output the result.



```
echo 10 ☐ 2;
```

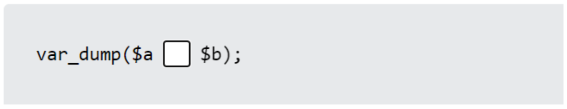
Solution:



```
echo 10 / 2;
```

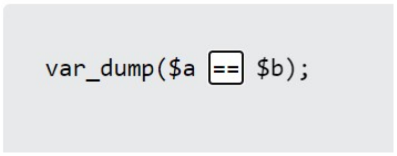
Exercise 3:

Use the correct comparison operator to check if `$a` is equal to `$b`.



```
var_dump($a ☐ $b);
```

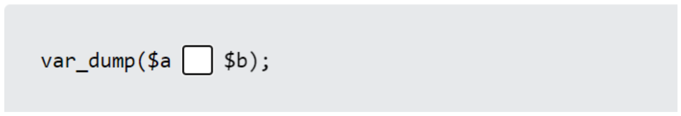
Solution:



```
var_dump($a == $b);
```

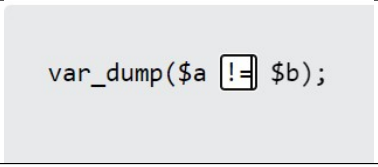
Exercise 4:

Use the correct comparison operator to check if `$a` is NOT equal to `$b`.



```
var_dump($a ☐ $b);
```

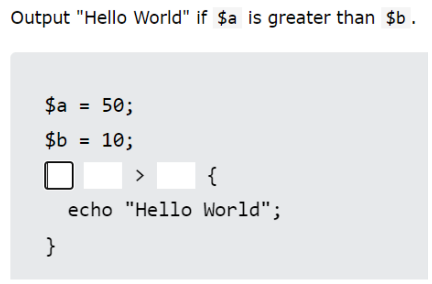
Solution:



```
var_dump($a != $b);
```

PHP if-else

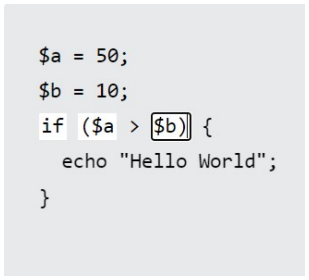
Exercise 1:



Output "Hello World" if `$a` is greater than `$b`.

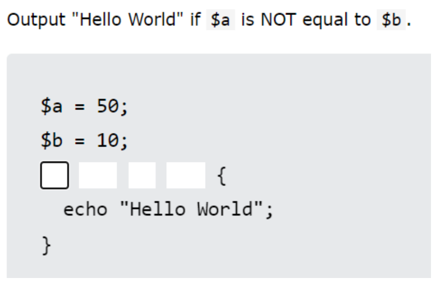
```
$a = 50;
$b = 10;
□      >      {
  echo "Hello World";
}
```

Solution:



```
$a = 50;
$b = 10;
if ($a > $b) {
  echo "Hello World";
}
```

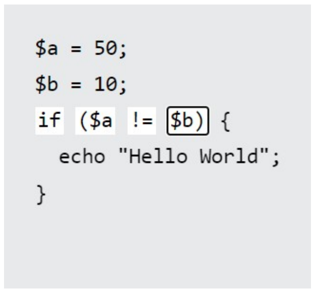
Exercise 2:



Output "Hello World" if `$a` is NOT equal to `$b`.

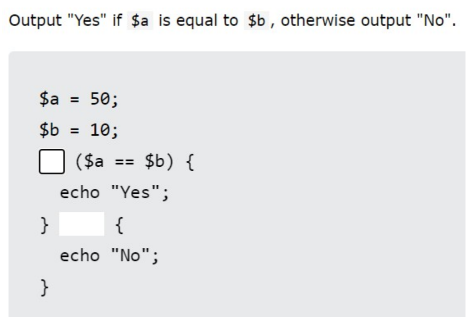
```
$a = 50;
$b = 10;
□              {
  echo "Hello World";
}
```

Solution:



```
$a = 50;
$b = 10;
if ($a != $b) {
  echo "Hello World";
}
```

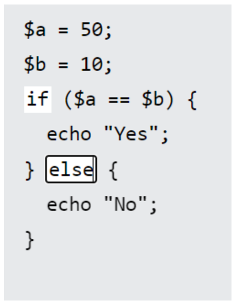
Exercise 3:



Output "Yes" if `$a` is equal to `$b`, otherwise output "No".

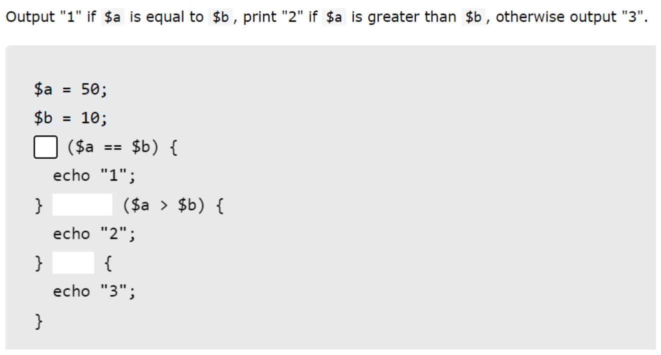
```
$a = 50;
$b = 10;
   ($a == $b) {
  echo "Yes";
}       {
  echo "No";
}
```

Solution:



```
$a = 50;
$b = 10;
if ($a == $b) {
  echo "Yes";
} else {
  echo "No";
}
```

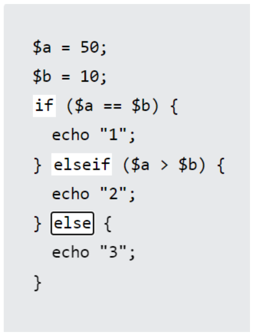
Exercise 4:



Output "1" if `$a` is equal to `$b`, print "2" if `$a` is greater than `$b`, otherwise output "3".

```
$a = 50;
$b = 10;
   ($a == $b) {
  echo "1";
}          ($a > $b) {
  echo "2";
}       {
  echo "3";
}
```

Solution:

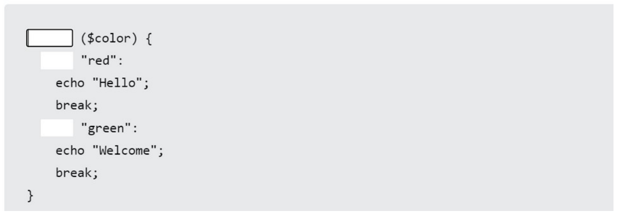


```
$a = 50;
$b = 10;
if ($a == $b) {
  echo "1";
} elseif ($a > $b) {
  echo "2";
} else {
  echo "3";
}
```

PHP switch:

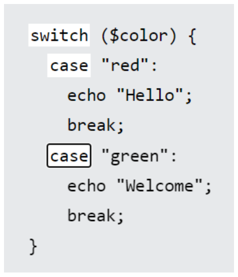
Exercise 1:

Create a `switch` statement that will output "Hello" if `$color` is "red", and "welcome" if `$color` is "green".



```
       ($color) {
        "red":
    echo "Hello";
    break;
        "green":
    echo "Welcome";
    break;
}
```

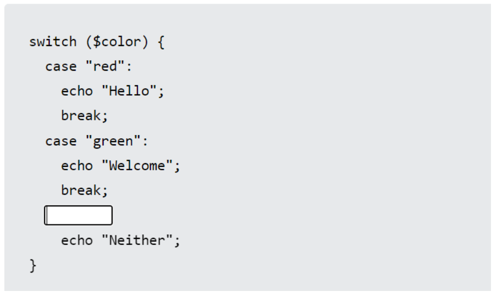
Solution:



```
switch ($color) {
  case "red":
    echo "Hello";
    break;
  case "green":
    echo "Welcome";
    break;
}
```

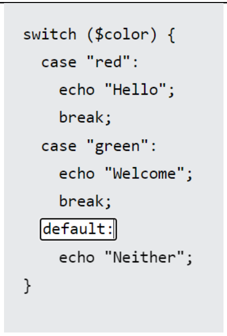
Exercise 2:

Add a section that will output "Neither" if `$color` is neither "red" nor "green".



```
switch ($color) {
  case "red":
    echo "Hello";
    break;
  case "green":
    echo "Welcome";
    break;
  
    echo "Neither";
}
```

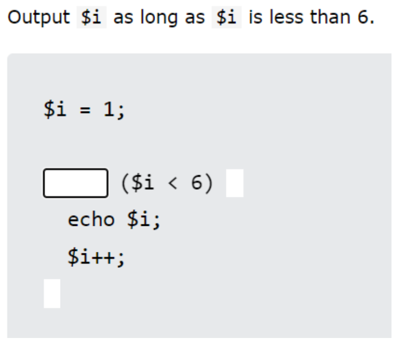
Solution:



```
switch ($color) {
  case "red":
    echo "Hello";
    break;
  case "green":
    echo "Welcome";
    break;
  default:
    echo "Neither";
}
```

PHP loops:

Exercise 1:



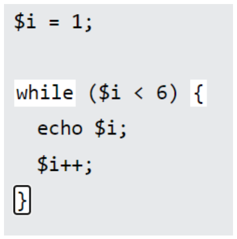
Output `$i` as long as `$i` is less than 6.

```
$i = 1;

      ($i < 6)
  echo $i;
  $i++;

```

Solution:

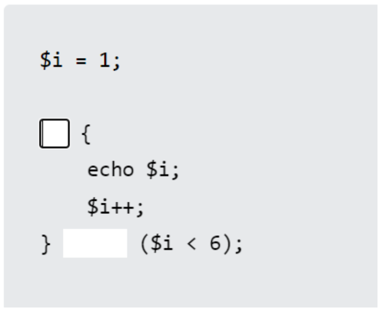


```
$i = 1;

while ($i < 6) {
  echo $i;
  $i++;
}
```

Exercise 2:

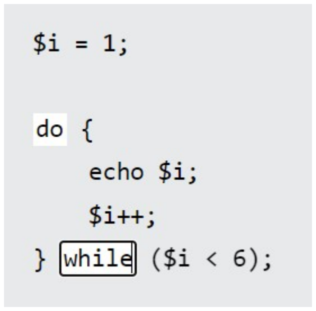
Output `$i` as long as `$i` is less than 6.



```
$i = 1;

    {
    echo $i;
    $i++;
}        ($i < 6);
```

Solution:

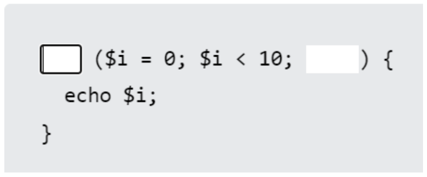


```
$i = 1;

do {
    echo $i;
    $i++;
} while ($i < 6);
```

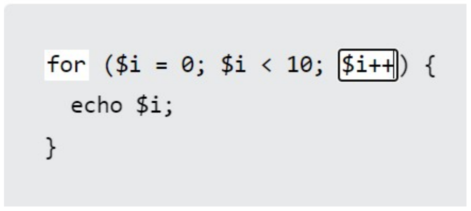
Exercise 3:

Create a loop that runs from 0 to 9.



```
    ($i = 0; $i < 10;     ) {
  echo $i;
}
```

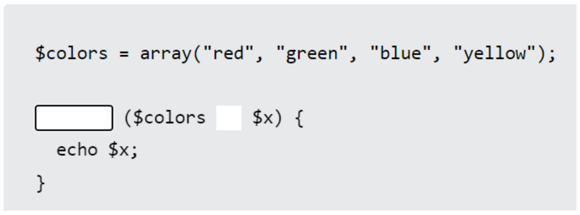
Solution:



```
for ($i = 0; $i < 10; $i++) {
  echo $i;
}
```

Exercise 4:

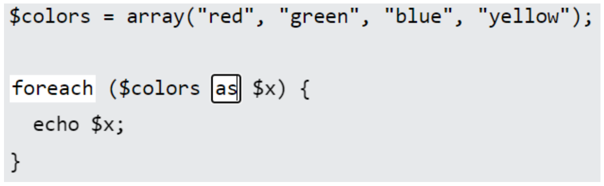
Loop through the items in the `$colors` array.



```
$colors = array("red", "green", "blue", "yellow");

      ($colors   $x) {
  echo $x;
}
```

Solution:

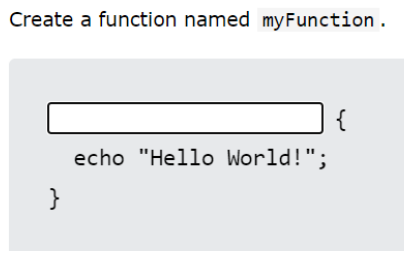


```
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $x) {
  echo $x;
}
```

PHP functions:

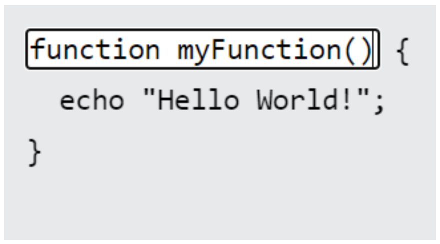
Exercise 1:



Create a function named `myFunction`.

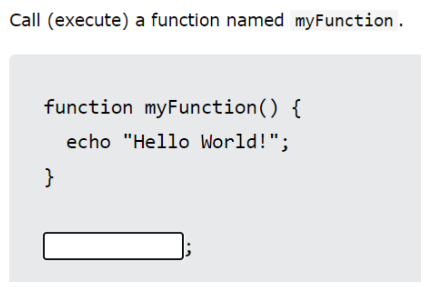
```
                {
  echo "Hello World!";
}
```

Solution:



```
function myFunction() {
  echo "Hello World!";
}
```

Exercise 2:

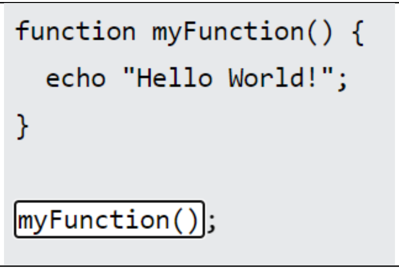


Call (execute) a function named `myFunction`.

```
function myFunction() {
  echo "Hello World!";
}

          ;
```

Solution:

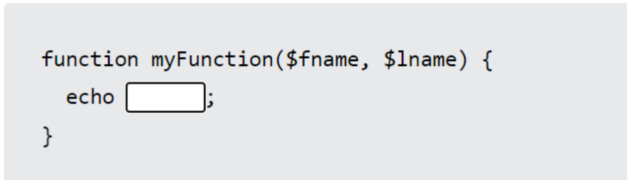


```
function myFunction() {
  echo "Hello World!";
}

myFunction();
```

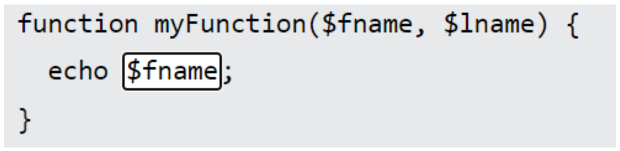
Exercise 3:

Inside a function with two parameters, print the first parameter.



```
function myFunction($fname, $lname) {
  echo        ;
}
```

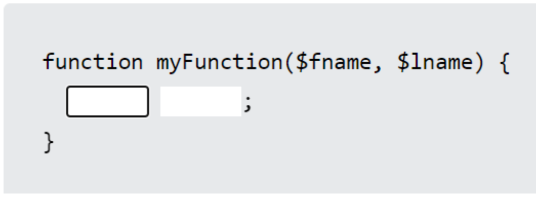
Solution:



```
function myFunction($fname, $lname) {
  echo $fname;
}
```

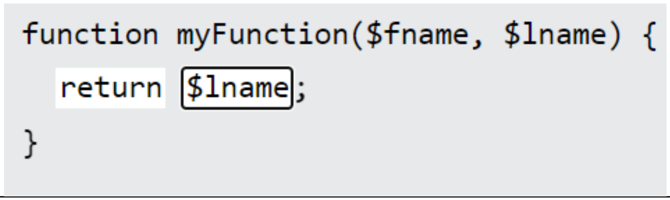
Exercise 4:

Let the function return the second value.



```
function myFunction($fname, $lname) {
              ;
}
```

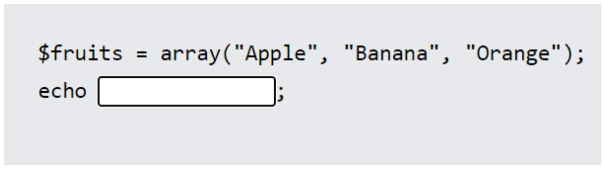
Solution:



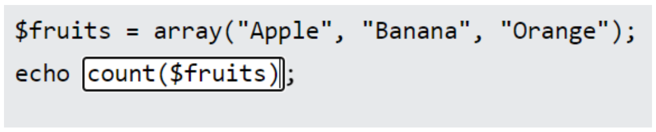
```
function myFunction($fname, $lname) {
  return $lname;
}
```

PHP arrays:

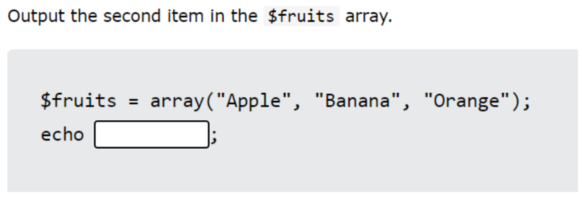
Exercise 1:


```
$fruits = array("Apple", "Banana", "Orange");
echo            ;
```

Solution:

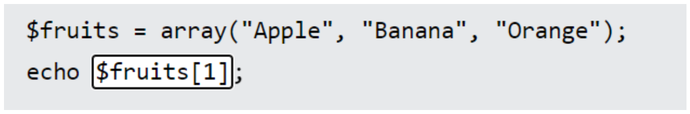

```
$fruits = array("Apple", "Banana", "Orange");
echo count($fruits);
```

Exercise 2:

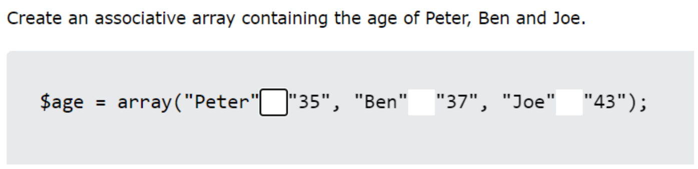

Output the second item in the `$fruits` array.

```
$fruits = array("Apple", "Banana", "Orange");
echo         ;
```

Solution:

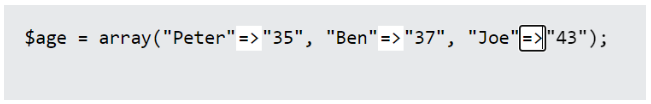

```
$fruits = array("Apple", "Banana", "Orange");
echo $fruits[1];
```

Exercise 3:

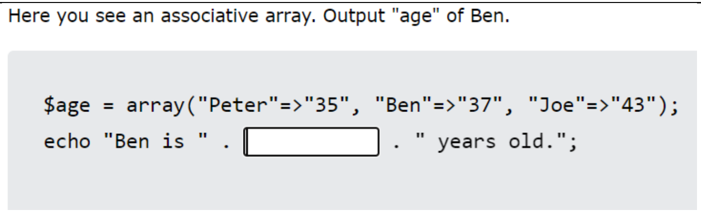

Create an associative array containing the age of Peter, Ben and Joe.

```
$age = array("Peter"  "35", "Ben"  "37", "Joe"  "43");
```

Solution:

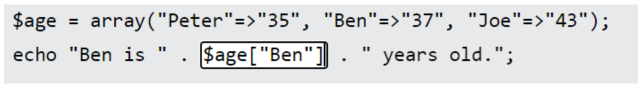

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

Exercise 4:


Here you see an associative array. Output "age" of Ben.

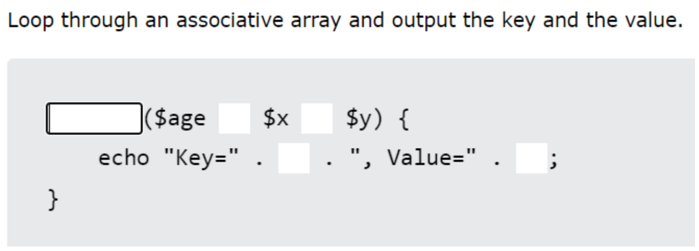
```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Ben is " .            . " years old.";
```

Solution:



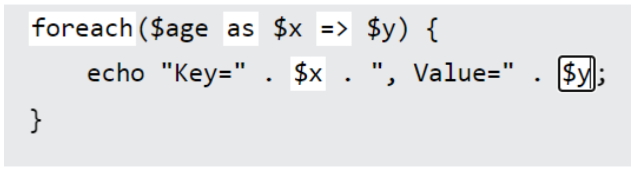
```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Ben is " . $age["Ben"] . " years old.";
```

Exercise 5:


Loop through an associative array and output the key and the value.

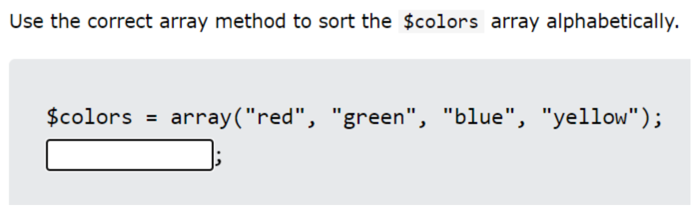
```
        ($age    $x    $y) {
    echo "Key=" .    . ", Value=" .    ;
}
```

Solution:



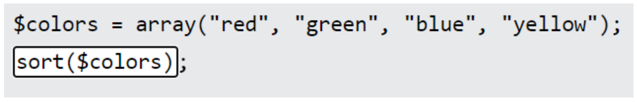
```
foreach($age as $x => $y) {
    echo "Key=" . $x . ", Value=" . $y;
}
```

Exercise 6:


Use the correct array method to sort the `$colors` array alphabetically.

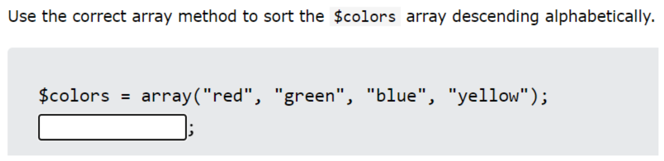
```
$colors = array("red", "green", "blue", "yellow");
            ;
```

Solution:



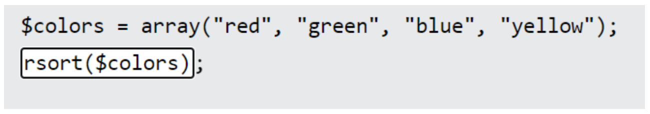
```
$colors = array("red", "green", "blue", "yellow");
sort($colors);
```

Exercise 7:


Use the correct array method to sort the `$colors` array descending alphabetically.

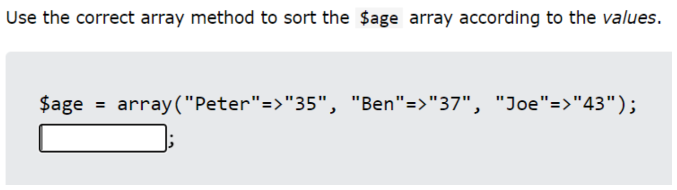
```
$colors = array("red", "green", "blue", "yellow");
;
```

Solution:



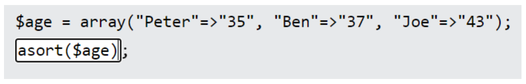
```
$colors = array("red", "green", "blue", "yellow");
rsort($colors);
```

Exercise 8:


Use the correct array method to sort the `$age` array according to the *values*.

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
;
```

Solution:



```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
asort($age);
```