ASSOCIATIVE AND MULTI-DIMENSIONAL ARRAYS

```php
<?php
$myarray = array(
  "one" => "one",
  "two" => 2,
  "three" => array(
    "sub_one" => "sub_one",
    "sub_two" => 2,
    "sub_three" => array(1, 2, 3)
  )
);
?>
```

WORKING WITH ASSOCIATIVE ARRAYS

```php
<?php
$car = array(2014, "black", 4);
$car = array(
  "year" => 2012,
  "color" => "black",
  "doors" => 4
);

$car["make"] = "BMW";
unset($car["year"]);

array_key_exists("year", $car); // false
array_key_exists("doors", $car); // true

foreach ($car as $key => $value) {
  echo $key . ": " . $value;
  echo $car[$key];
}
```

WORKING WITH MULTI-DIMENSIONAL ARRAYS

```
<?php
?>
```

OBJECT ORIENTED PROGRAMMING

```php
<?php
public class Bicycle
{
  public $size;
  public $chain;
  public $tireSize;

  public function __construct($size, $chain, $tireSize)
  {
    $this->size = $size;
    $this->chain = $chain;
    $this->tireSize = $tireSize;
  }

  public function tireSize()
  {
    return "The tire size is " . $this->tireSize . " inches";
  }
```

YOUR FIRST CLASS

```php
<?php
?>
```

MAGIC METHODS

```php
<?php
__construct();
__destruct();
?>
```

VISIBILITY AND INHERITANCE

```php
<?php
public class Bicycle
{
  public $size;
  public $chain;
  public $tireSize;

  public function __construct($size, $chain, $tireSize)
  {
    $this->size = $size;
    $this->chain = $chain;
    $this->tireSize = $tireSize;
  }

  public function tireSize()
  {
    return "The tire size is " . $this->tireSize . " inches";
  }
```