Начальные значения свойств при объявлении

Рассмотрим следующий класс:

```php
<?php
  class Test
  {
    public $prop1;
    public $prop2;

    public function __construct()
    {
      $this->prop1 = 'value1'; // начальное значение свойства prop1
      $this->prop2 = 'value2'; // начальное значение свойства prop2
    }
  }

  $test = new Test;
  echo $test->prop1; // выведет 'value1'
  echo $test->prop2; // выведет 'value2'
?>
```

Как вы видите, в этом коде в конструкторе объекта мы задаем начальные значения свойств.

На самом деле можно сократить лишний код, задав начальные значения свойств прямо при их объявлении:

```php
<?php
  class Test
  {
    public $prop1 = 'value1'; //!! начальное значение свойства prop1
    public $prop2 = 'value2'; //!! начальное значение свойства prop2
  }

  $test = new Test;
  echo $test->prop1; // выведет 'value1'
  echo $test->prop2; // выведет 'value2'
?>
```

Замечания

Конечно же, не обязательно задавать начальные значения всем свойствам:

```php
<?php
  class Test
  {
    public $prop1 = 'value1'; // задаем начальное значение
    public $prop2; // не задаем
  }
?>
```

При задании начальных значений свойств можно выполнять некоторые операции:

```php
<?php
  class Test
  {
    public $prop = 1 + 2; //!! найдем сумму чисел
  }

  $test = new Test;
  echo $test->prop; // выведет 3
?>
```

> Можно выполнять только самые примитивные операции, к примеру, нельзя написать вызов функции и тп.

Применение

Пусть у нас есть вот такой класс **Student**, в конструкторе которого задается начальное значение свойства **course**:

```php
<?php
  class Student
  {
    private $name;
    private $course;

    public function __construct($name)
    {
      $this->name = $name;
      $this->course = 1; // начальное значение курса
    }

    // Геттер имени:
    public function getName()
    {
      return $this->name;
    }

    // Геттер курса:
    public function getCourse()
    {
      return $this->course;
    }

    // Перевод студента на новый курс:
    public function transferToNextCourse()
    {
      $this->course++;
    }
  }
?>
```

Давайте вынесем начальное значение курса в объявление свойства:

```php
<?php
  class Student
  {
    private $name;
    private $course = 1; // начальное значение курса

    public function __construct($name)
    {
      $this->name = $name;
    }

    // Геттер имени:
    public function getName()
    {
      return $this->name;
    }

```

```
    // Геттер курса:
    public function getCourse()
    {
      return $this->course;
    }

    // Перевод студента на новый курс:
    public function transferToNextCourse()
    {
      $this->course++;
    }
  }
?>
```

Применение

Пусть у нас есть вот такой класс **Arr**, у которого есть метод **add** для добавления чисел и метод **getSum** для получения суммы всех добавленных чисел:

```
<?php
  class Arr
  {
    // Массив для хранения чисел:
    private $numbers;

    // Добавляет число в набор:
    public function add($num)
    {
      $this->numbers[] = $num;
    }

    // Находит сумму чисел набора:
    public function getSum()
    {
      return array_sum($this->numbers);
    }
  }
?>
```

Давайте воспользуемся нашим классом **Arr** - добавим несколько чисел и найдем их сумму:

```
<?php
  $arr = new Arr;

  $arr->add(1);
  $arr->add(2);
  $arr->add(3);

  echo $arr->getSum(); // выведет 6
?>
```

Все вроде работает, но что будет, если сразу после создания вызвать метод **getSum**? Вот таким образом:

```
<?php
  $arr = new Arr;
  echo $arr->getSum(); // будет ошибка!
?>
```

Такой код вызовет ошибку! Почему: потому что функция **array_sum** пытается найти сумму массива **$this->numbers** - но там еще нет элементов.

А так как в свойство **numbers** мы еще ничего не записали, то там лежит значение **null** и фактически мы вызываем код **array_sum(null)**, что, конечно же, выдает ошибку, так как **array_sum** ожидает, что его параметром будет именно массив.

Давайте исправим проблему, объявив свойство **numbers** пустым массивом:

```
<?php
  class Arr
```

```
3   {
4     private $numbers = []; // задаем начальное зна
        чение свойства как []
5
6     public function add($num)
7     {
8       $this->numbers[] = $num;
9     }
10
11    public function getSum()
12    {
13      return array_sum($this->numbers);
14    }
15  }
16 ?>
```

Проверим:

```
1  <?php
2    $arr = new Arr;
3    echo $arr->getSum(); // выведет 0
4  ?>
```

Задача 13.1

Реализуйте класс **Arr**, похожий на тот, который я реализовал выше.

В отличие от моего класса метод **add** вашего класса параметром должен принимать массив чисел. Все числа из этого массива должны добавляться в конец массива **$this->numbers**.

Задача 13.2

Вместо метода **getSum** реализуйте метод **getAvg**, который будет находить среднее арифметическое переданных чисел.