



Министерство науки и высшего образования Российской Федерации  
Калужский филиал  
федерального государственного бюджетного  
образовательного учреждения высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(КФ МГТУ им. Н.Э. Баумана)

**ФАКУЛЬТЕТ ИУ-КФ «Информатика и управление»**

**КАФЕДРА ИУ4-КФ «Программное обеспечение ЭВМ, информационные технологии»**

## **ЛАБОРАТОРНАЯ РАБОТА №5**

**«Объектное программирование»**

**ДИСЦИПЛИНА: «Кроссплатформенная разработка программного обеспечения»**

Выполнил: студент гр. ИТД.Б-62 \_\_\_\_\_ Паксеваткин А. С.  
(Подпись)

Проверил: \_\_\_\_\_ Овсиенко О.С.  
(Подпись)

Дата сдачи (защиты):

Результаты сдачи (защиты):

— Балльная оценка:

— Оценка:

Калуга, 2020

**Цель:** приобретение практических навыков работы с объектным программированием в языке PHP.

**Задачи:**

1. Изучить способы создания классов;
2. Ознакомиться со способами вывода данных в формате таблицы;
3. Изучить статические и "волшебные" методы классов;
4. Изучить способ документирования с помощью @property phpDoc

**Задание:**

1. Переработать класс Table <http://htmlweb.ru/php/php6.php> пример 3 для вывода в формате таблицы с тегами table, th, tr, td
2. Добавить закрытие тегов tr, th, td в примере 6 <http://htmlweb.ru/php/php6.php>
3. Создать класс с именем **baseClass**, в свойствах которого сохраняются два числа. Написать к нему метод **calculate()**, который выводит эти числа на экран.
4. Создать класс с именем **addCalc**, производный от класса **baseClass**. Переопределить его метод **calculate()** так, чтобы он выводил на экран сумму чисел.
5. Создать класс с именем **minusCalc**, производный от класса **baseClass**. Переопределить его метод **calculate()** так, чтобы он выводил на экран разность первого и второго чисел.
6. Создать класс содержащий static метод **\_GetVar(id, свойство)**, который создает объект класса и возвращает значение свойства переданного во втором параметре. Это задание на понимание различий статических и динамических методов. При обращении к static методу нужно создать(new) новый экземпляр класса получить у него свойство, переданное в качестве параметра и вернуть его.
7. Создать класс в котором будут производиться запись и чтение любых свойств этого класса. Использовать массив со свойствами и методы-перехватчики **\_\_get**, **\_\_set**, **\_\_unset**, **\_\_isset**  
Отдокументировать 3 свойства с помощью @property phpDoc
8. Определить волшебный метод **Sum(a,b)** - суммирования двух "волшебных" свойств переданных в параметре с помощью **\_\_call**. Отдокументировать этот метод, определенный с помощью phpDoc

**Исходный код:**

```
<!-- task5.php -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
```

```

<title>Лабораторная работа №5</title>
<link rel="stylesheet" href="styles.css"/>
</head>
<body>
<main>
<?php
    // 1 Переработать класс Table пример 3 для вывода в формате таблицы с
    тегами table, th, tr, td
    class Table
    {
        protected $headers = [];
        protected $data = [];
        function Table ( $headers )
        {
            $this->headers = $headers;
        }
        function addRow ( $row )
        {
            $tmp = [];
            foreach ( $this->headers as $header ) {
                if ( ! isset( $row[$header] ) ) $row[$header] = "";
                $tmp[] = $row[$header];
            }
            array_push ( $this->data, $tmp );
        }
        function output ( )
        {
            echo "<table><tr>";
            foreach ( $this->headers as $header )
                echo "<th>$header</th>";
            echo "</tr>";

            foreach ( $this->data as $y )
            {
                echo "<tr>";
                foreach ( $y as $x )
                    echo "<td>$x</td>";
                echo "</tr>";
            }
            echo "</table>";
        }
    }

    $test1 = new Table(["a","b","c"]);
    $test1->addRow(["a"=>1,"b"=>3,"c"=>2]);
    $test1->addRow(["a"=>1,"c"=>3]);
    $test1->addRow(["c"=>1,"b"=>3,"a"=>4]);
    $test1->output();

    echo '<br />';

    // 2 Добавить закрытие тегов tr, th, td в примере 6
    class HTMLTable extends Table
    {
        public $cellpadding = "2";
        public $bgcolor;
        function HTMLTable ( $headers, $bg="FFFFFF" )
        {
            Table::Table( $headers );
            $this->bgcolor = $bg;
        }
        function setCellpadding ( $padding )
        {

```

```

        $this->cellpadding = $padding;
    }
    function output ()
    {
        echo "<table cellpadding='". $this->cellpadding. "'><tr>";

        foreach ( $this->headers as $header )
            echo "<th bgcolor='". $this->bgcolor. "'>$header</th>";

        echo '</tr>';

        foreach ( $this->data as $y ) {
            echo "<tr>";
            foreach ( $y as $x )
                echo "<td bgcolor='". $this->bgcolor. "'>$x</td>";
            echo "</tr>";
        }
        echo "</table>";
    }
}

$test2 = new HTMLTable ( array("a","b","c"), "#00FFFF" );
$test2->setCellpadding ( 7 );

$test2->addRow(["a"=>1,"b"=>3,"c"=>2]);
$test2->addRow(["a"=>1,"c"=>3]);
$test2->addRow(["c"=>1,"b"=>3,"a"=>4]);

$test2->output();

echo '<br />';

// 3 Создать класс с именем baseClass, в свойствах которого сохраняются
два числа. Написать к нему метод calculate(), который выводит эти числа на
экран.
class BaseClass
{
    protected $a;
    protected $b;

    function BaseClass(float $a, float $b)
    {
        $this->a = $a;
        $this->b = $b;
    }

    public function calculate()
    {
        echo '('.$this->a.', '.$this->b.')';
    }
}

echo 'new BaseClass(1, 2).calculate(): ', (new BaseClass(1, 2))-
>calculate();
echo '<br />';

// 4 Создать класс с именем addCalc, производный от класса baseClass.
Переопределить его метод calculate() так, чтобы он выводил на экран сумму
чисел.
class AddCalc extends BaseClass {
    public function calculate()
    {
        echo $this->a + $this->b;
    }
}

```

```

    }
}
echo 'new AddCalc(1, 2).calculate(): ', (new AddCalc(1, 2))->calculate();
echo '<br />';

// 5 Создать класс с именем minusCalc, производный от класса baseClass.
Переопределить его метод calculate() так, чтобы он выводил на экран разность
первого и второго чисел.
class MinusCalc extends BaseClass {
    public function calculate()
    {
        echo $this->a - $this->b;
    }
}
echo 'new MinusCalc(1, 2).calculate(): ', (new MinusCalc(1, 2))-
>calculate();
echo '<br />';

// 6 Создать класс содержащий static метод _GetVar(id, свойство), который
создает объект класса и возвращает значение свойства переданного во втором
параметре. Это задание на понимание различий статических и динамических
методов. При обращении к (static* методу нужно создать(new) новый экземпляр
класса получить у него свойство, переданное в качестве параметра и вернуть
его.

class TestClass
{
    private $id;
    private $ten = 10;

    public function TestClass($id)
    {
        $this->id = $id;
    }

    public static function _GetVar($id, $prop)
    {
        $obj = new TestClass($id);
        return $obj->$prop;
    }
}

echo 'TestClass::_GetVar(5, "id"): ', TestClass::_GetVar(5, "id"),
'<br/>';
echo 'TestClass::_GetVar(5, "ten"): ', TestClass::_GetVar(5, "ten"),
'<br/>';

// 7 Создать класс в котором будут производиться запись и чтение любых
свойств этого класса. Использовать массив со свойствами и методы-перехватчики
__get, __set, __unset, __isset. Отдокументировать 3 свойства с помощью
@property phpDoc
// 8 Определить волшебный метод Sum(a,b) - суммирование двух "волшебных"
свойств переданных в параметре с помощью __call. Отдокументировать этот метод,
определенный с помощью phpDoc
/**
 * @property int $myProperty1
 * @property int $myProperty2
 * @property int $myProperty3
 * @method int Sum(string $prop1, string $prop1)
 */
class MagicClass
{

```

```

private $map = [];

function __get($prop)
{
    return $this->map[$prop];
}

function __set($prop, $val)
{
    $this->map[$prop] = $val;
}

function __unset($prop)
{
    unset( $this->map[$prop] );
}

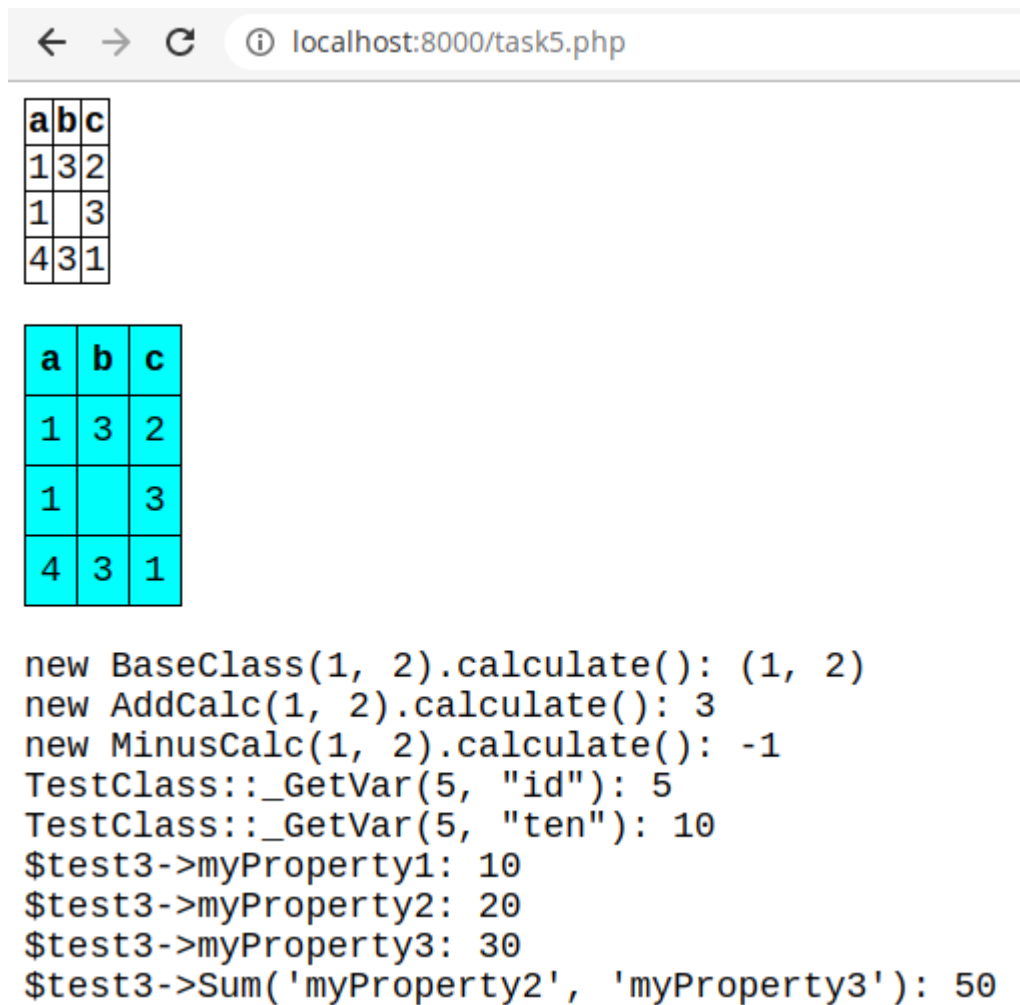
function __isset($prop)
{
    return $this->map[$prop] !== null;
}

function __call($method, $args)
{
    if($method === 'Sum') {
        return $this->map[ $args[0] ] + $this->map[ $args[1] ];
    }
    return $this->$method(...$args);
}
}

$test3 = new MagicClass();
$test3->myProperty1 = 10;
$test3->myProperty2 = 20;
$test3->myProperty3 = 30;
echo '$test3->myProperty1: ', $test3->myProperty1, '<br/>';
echo '$test3->myProperty2: ', $test3->myProperty2, '<br/>';
echo '$test3->myProperty3: ', $test3->myProperty3, '<br/>';
echo '$test3->Sum(\'myProperty2\', \'myProperty3\'): ', $test3-
>Sum('myProperty2', 'myProperty3'), '<br/>';
?>
</main>
</body>
</html>

```

## Результаты выполнения программы:



← → ↻ ⓘ localhost:8000/task5.php

a	b	c
1	3	2
1		3
4	3	1

a	b	c
1	3	2
1		3
4	3	1

```
new BaseClass(1, 2).calculate(): (1, 2)
new AddCalc(1, 2).calculate(): 3
new MinusCalc(1, 2).calculate(): -1
TestClass::_GetVar(5, "id"): 5
TestClass::_GetVar(5, "ten"): 10
$test3->myProperty1: 10
$test3->myProperty2: 20
$test3->myProperty3: 30
$test3->Sum('myProperty2', 'myProperty3'): 50
```

Рис. 1.

**Вывод:** в ходе выполнения лабораторной работы были приобретены практические навыки работы с объектным программированием в языке PHP.