|  |
| --- |
| Министерство образования Республики Беларусь  УО «Полоцкий государственный университет им. Ефросинии Полоцкой» |

|  |  |
| --- | --- |
|  | Факультет информационных технологий  Кафедра технологий программирования |

|  |
| --- |
| ОТЧЕТ |
| **к лабораторной работе №12(3)**  **Дисциплина**: «Программирование для Интернет» |

|  |  |
| --- | --- |
| Выполнил: | Студент группы 19-ИТ-2  Гилимович А.Н. |

|  |  |
| --- | --- |
| Проверила: | Преподователь  Виноградова А.Д. |

|  |
| --- |
| Полоцк, 2022 г. |

**Объектно-ориентированное программирование в PHP.**

**Цель работы:** ознакомиться с базовыми понятиями ООП: объекты и классы, конструкторы и деструкторы, статические методы и свойства, интерфейсы.

**Задание 1**

Сделайте класс Worker, в котором будут следующие public поля - name (имя), age (возраст), salary (зарплата). Создайте объект этого класса, затем установите поля в следующие значения (не в \_\_construct, а для созданного объекта) – имя 'Иван', возраст 25, зарплата 1000. Создайте второй объект этого класса, установите поля в следующие значения - имя 'Вася', возраст 26, зарплата 2000. Выведите на экран сумму зарплат Ивана и Васи. Выведите на экран сумму возрастов Ивана и Васи.

**Задание 2**

Сделайте класс Worker, в котором будут следующие private поля - name

(имя), age (возраст), salary (зарплата) и следующие public методы setName,

getName, setAge, getAge, setSalary, getSalary. Создайте 2 объекта этого класса:

'Иван', возраст 25, зарплата 1000 и 'Вася', возраст 26, зарплата 2000. Выведите на экран сумму зарплат Ивана и Васи. Выведите на экран сумму возрастов Ивана и Васи.

**Задание 3**

Дополните класс Worker из предыдущей задачи private методом checkAge,

который будет проверять возраст на корректность (от 1 до 100 лет). Этот метод

должен использовать метод setAge перед установкой нового возраста (если возраст не корректный - он не должен меняться).

**Задание 4**

Сделайте класс Worker, в котором будут следующие private поля – name (имя), salary (зарплата). Сделайте так, чтобы эти свойства заполнялись в методе \_\_construct при создании объекта (вот так: new Worker(имя, возраст) ). Сделайте также public методы getName, getSalary. Создайте объект этого класса 'Дима', возраст 25, зарплата 1000. Выведите на экран произведение его возраста и зарплаты.

**Задание 5**

Сделайте класс User, в котором будут следующие protected поля – name (имя), age (возраст), public методы setName, getName, setAge, getAge. Сделайте класс Worker, который наследует от класса User и вносит дополнительное private поле salary (зарплата), а также методы public getSalary и setSalary. Создайте объект этого класса 'Иван', возраст 25, зарплата 1000. Создайте второй объект этого класса 'Вася', возраст 26, зарплата 2000. Найдите сумму зарплата Ивана и Васи. Сделайте класс Student, который наследует от класса User и вносит дополнительные private поля стипендия, курс, а также геттеры и сеттеры для них.

**Задание 6**

Сделайте класс Driver (Водитель), который будет наследоваться от класса Worker из предыдущей задачи. Этот метод должен вносить следующие private поля: водительский стаж, категория вождения (A, B, C).

**Задание 8**

Создать HTML-документ, реализующий следующую задачу: Список содержит следующие операции: четные, нечетные, простые, составные. Вывести все

числа из диапазона от 1 до N, согласно выбранному из списка действию (N вводится пользователем в текстовое поле).

**Ход работы**

Изучив теоритический материал, преступил к выполнению заданий.

**Листинг 1** – задания 1

<?php

class Worker

{

public $name, $age, $salary;

}

$Ivan = new Worker();

$Ivan -> name = "Ivan";

$Ivan -> age = 25;

$Ivan -> salary = 1000;

$Vasya = new Worker();

$Vasya -> name = "Vasya";

$Vasya -> age = 26;

$Vasya -> salary = 2000;

$sumOfSalaryes = $Ivan->salary + $Vasya->salary;

$sumOfAges = $Ivan->age + $Vasya->age;

echo "sum of salaryes ".$sumOfSalaryes." <br>";

echo "sum of ages ".$sumOfAges." <br>";

?>

**Листинг 2** – задания 2

<?php

class Worker2{

private $name, $age, $salary;

function setName($name){

$this->name = $name;

}

function getName(){

return $this->name;

}

function setAge($age){

$this->age = $age;

}

function getAge(){

return $this->age;

}

function setSalary($salary){

$this->salary = $salary;

}

function getSalary(){

return $this->salary;

}

}

$Ivan = new Worker2();

$Ivan->setName("Ivan");

$Ivan->setAge(25);

$Ivan->setSalary(1000);

$Vasya = new Worker2();

$Vasya->setName("Vasya");

$Vasya->setAge(26);

$Vasya->setSalary(2000);

$sumOfSalaryes = $Ivan->getSalary() + $Vasya->getSalary();

$sumOfAges = $Ivan->getAge() + $Vasya->getAge();

echo "sum of salaryes ".$sumOfSalaryes." <br>";

echo "sum of ages ".$sumOfAges." <br>";

?>

**Листинг 3** – задание 3

<?php

class Worker3{

private $name, $age, $salary;

function setName($name){

$this->name = $name;

}

function getName(){

return $this->name;

}

function setAge($age){

if ($this->checkAge($age)) $this->age = $age;

}

function getAge(){

return $this->age;

}

function setSalary($salary){

$this->salary = $salary;

}

function getSalary(){

return $this->salary;

}

private function checkAge($age){

if($age <= 100 && $age >= 1) return true;

return false;

}

}

$Ivan = new Worker3();

$Ivan->setName("Ivan");

$Ivan->setAge(25);

$Ivan->setSalary(1000);

$Vasya = new Worker3();

$Vasya->setName("Vasya");

$Vasya->setAge(26);

$Vasya->setSalary(2000);

$sumOfSalaryes = $Ivan->getSalary() + $Vasya->getSalary();

$sumOfAges = $Ivan->getAge() + $Vasya->getAge();

echo "sum of salaryes ".$sumOfSalaryes." <br>";

echo "sum of ages ".$sumOfAges." <br>";

?>

**Листинг 4** – задание 4

<?php

class ConstructWorker {

private $name;

private $age;

private $salary;

public function \_\_construct($name, $age, $salary) {

$this->name = $name;

$this->age = $age;

$this->salary = $salary;

}

public function getName() {

return $this->name;

}

public function getAge() {

return $this->age;

}

public function getSalary() {

return $this->salary;

}

}

$Ivan = new ConstructWorker("Ivan",25,1000);

$Vasya = new ConstructWorker("Vasya",26,2000);

$Dima = new ConstructWorker("Dima",25,1000);

echo $Dima->getAge() \* $Dima->getSalary()." <br>";

?>

**Листинг 5** – задание 5

<?php

class User{

protected $name, $age;

function setName($name){

$this->name = $name;

}

function getName(){

return $this->name;

}

function setAge($age){

$this->age = $age;

}

function getAge(){

return $this->age;

}

}

class Worker4 extends User{

private $salary;

public function setSalary($salary){

$this->salary = $salary;

}

public function getSalary(){

return $this->salary;

}

}

class Student extends User{

private $step, $course;

public function setStep($step){

$this->step = $step;

}

public function getStep(){

return $this->step;

}

public function setCourse($course){

$this->course = $course;

}

public function getCourse(){

return $this->course;

}

}

$Ivan = new Worker4();

$Ivan->setName("Ivan");

$Ivan->setAge(25);

$Ivan->setSalary(1000);

$Vasya = new Worker4();

$Vasya->setName("Vasya");

$Vasya->setAge(26);

$Vasya->setSalary(2000);

$sumOfSalaryes = $Ivan->getSalary() + $Vasya->getSalary();

$sumOfAges = $Ivan->getAge() + $Vasya->getAge();

echo "sum of salaryes ".$sumOfSalaryes." <br>";

?>

**Листинг 6** – задание 6

<?php

class Driver extends Worker4 {

private $exp, $category;

public function setExp($exp){

$this->exp = $exp;

}

public function getExp(){

return $this->exp;

}

public function setCategory($category){

$this->category = $category;

}

public function getCategory(){

return $this->category;

}

}

?>

**Вывод:** ознакомились с базовыми понятиями ООП: объекты и классы, конструкторы и деструкторы, статические методы и свойства, интерфейсы.