МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Полоцкий государственный университет»

Факультет информационных технологий

Кафедра технологий программирования

**Лабораторная работа №4**

по дисциплине: «Веб-технологии»

на тему: «Объектно-ориентированное программирование в PHP»

Вариант №12

|  |  |
| --- | --- |
| Выполнил | Студент группы 21-ИТ-1  Чиникайло А.П. |
|  |  |
| Проверил | Преподаватель  Сергеев М. А. |

Полоцк 2023

**Цель работы:**

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

**Ход работы**

**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).

Листинг 1 – первое задание

<?php

    class Worker {

        public $name;

        public $age;

        public $salary;

    }

    $work1 = new Worker();

    $work1->name = "Иван";

    $work1->age = 25;

    $work1->salary = 1000;

    echo $work1->name." ";

    echo $work1->age." ";

    echo $work1->salary."<br>";

    $work2 = new Worker();

    $work2->name = "Вася";

    $work2->age = 26;

    $work2->salary = 2000;

    echo $work2->name." ";

    echo $work2->age." ";

    echo $work2->salary."<br>";

    echo "Сумма возрастов и зарплат: ".$work1->age + $work2->age." ".$work1->salary + $work2->salary;

?>

Листинг 2 – второе и третье задание

<?php

    class Worker {

        private $name;

        private $age;

        private $salary;

        public function setName($newName) {

            $this->name = $newName;

        }

        public function setAge($newName) {

            if ($this->checkAge($newName)) {

                $this->age = $newName;

            }

        }

        public function setSalary($newName) {

            $this->salary = $newName;

        }

        public function getName() {

            return $this->name;

        }

        public function getAge() {

            return $this->age;

        }

        public function getSalary() {

            return $this->salary;

        }

        private function checkAge($newAge) {

            echo "Сработал checkAge<br>";

            return $newAge > 0 && $newAge < 100;

        }

    }

    $work1 = new Worker();

    $work1->setName("Иван");

    $work1->SetAge(25);

    $work1->setSalary(1000);

    $work2 = new Worker();

    $work2->setName("Вася");

    $work2->setAge(26);

    $work2->setSalary(2000);

    echo "Сумма возрастов и зарплат: ".$work1->getAge() + $work2->getAge()." ".$work1->getSalary() + $work2->getSalary();

?>

Листинг 3 – четвертое задание

<?php

    class Worker {

        private $name;

        private $age;

        private $salary;

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

            $this->name = $name;

            $this->age = $age;

        }

        public function setSalary($newName) {

            $this->salary = $newName;

        }

        public function getName() {

            return $this->name;

        }

        public function getAge() {

            return $this->age;

        }

        public function getSalary() {

            return $this->salary;

        }

    }

    $work1 = new Worker("Дима", 25);

    $work1->setSalary(1000);

    echo "Произведение возраста и зарплаты: ".$work1->getAge() \* $work1->getSalary();

?>

Листинг 4 – пятое задание

<?php

    class User {

        protected $name;

        protected $age;

        public function setName($newName) {

            $this->name = $newName;

        }

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

            $this->name = $name;

            $this->age = $age;

        }

        public function setAge($newName) {

            $this->age = $newName;

        }

        public function getName() {

            return $this->name;

        }

        public function getAge() {

            return $this->age;

        }

    }

    class Worker extends User {

        private $salary;

        public function setSalary($newName) {

            $this->salary = $newName;

        }

        public function getSalary() {

            return $this->salary;

        }

    }

    class Student extends User {

        private $stipendia;

        private $course;

        public function setCousre($newName) {

            $this->course = $newName;

        }

        public function getCourse() {

            return $this->course;

        }

        public function setStipendia($newName) {

            $this->stipendia = $newName;

        }

        public function getStipendia() {

            return $this->stipendia;

        }

    }

    $work1 = new Worker("Иван", 25);

    $work1->setSalary(1000);

    $work2 = new Worker("Вася", 26);

    $work2->setSalary(2000);

    echo "Сумма зарплат: ".$work1->getSalary() + $work2->getSalary();

?>

Листинг 5 – шестое задание

<?php

    class User {

        protected $name;

        protected $age;

        public function setName($newName) {

            $this->name = $newName;

        }

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

            $this->name = $name;

            $this->age = $age;

        }

        public function setAge($newName) {

            $this->age = $newName;

        }

        public function getName() {

            return $this->name;

        }

        public function getAge() {

            return $this->age;

        }

    }

    class Worker extends User {

        private $salary;

        public function setSalary($newName) {

            $this->salary = $newName;

        }

        public function getSalary() {

            return $this->salary;

        }

    }

    class Driver extends Worker {

        private $expirience;

        private $drive;

        public function setExpirience($newName) {

            $this->expirience = $newName;

        }

        public function getExpirience() {

            return $this->expirience;

        }

        public function setDrive($newName) {

            $this->drive = $newName;

        }

        public function getDrive() {

            return $this->drive;

        }

    }

    $Drive1 = new Worker("Иван", 25);

    $Drive1->setSalary(1000);

    $Drive1->setExpirience(10);

    $Drive1->setDrive("C");

?>

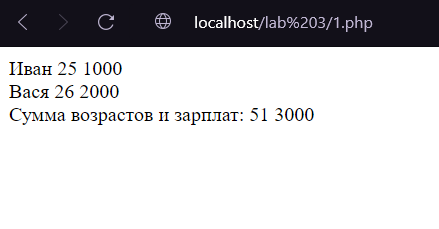


Рисунок 1 – Результат работы первого задания

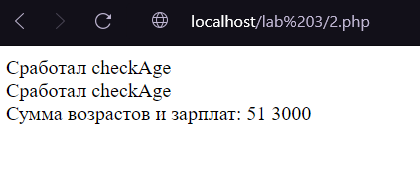


Рисунок 2 – Результат работы второго и третьего задания

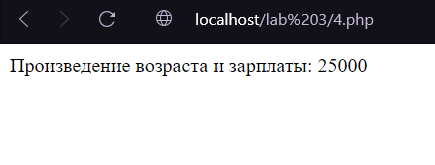


Рисунок 3 – Результат работы четвертого задания

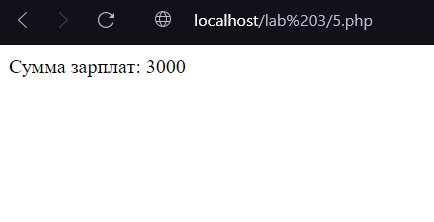


Рисунок 4 – Результат работы пятого задания

**Вывод:**

В результате лабораторной работы было ознакомление с базовыми понятиями ООП: объекты и классы, конструкторы и деструкторы, статические методы и свойства, интерфейсы.