Наследование классов

Представьте, что у вас есть класс **User**. Он нужен вам для каких-то целей и в общем-то полностью вас устраивает - доработки этому классу в не нужны.

Вот этот класс:

```php
<?php
  class User
  {
    private $name;
    private $age;

    public function getName()
    {
      return $this->name;
    }

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

    public function getAge()
    {
      return $this->age;
    }

    public function setAge($age)
    {
      $this->age = $age;
    }
  }
?>
```

А теперь представим себе ситуацию, когда нам понадобился еще и класс **Employee** (работник). Работник очень похож на юзера, имеет те же свойства и методы, но еще и добавляет свои - свойство **salary** (зарплата), а также геттер и сеттер для этого свойства.

Вот этот класс **Employee**:

```php
<?php
  class Employee
  {
    private $name;
    private $age;
    private $salary; // зарплата

    // Геттер зарплаты
    public function getSalary()
    {
      return $this->salary;
    }

    // Сеттер зарплаты
    public function setSalary($salary)
    {
      $this->salary = $salary;
    }

```

```
    public function getName()
    {
      return $this->age;
    }

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

    public function getAge()
    {
      return $this->age;
    }

    public function setAge($age)
    {
      $this->age = $age;
    }
  }
?>
```

Как мы видим, код классов **User** и **Employee** практически полностью совпадает.

Было бы намного лучше сделать так, чтобы общая часть была записана только в одном месте.

Если обдумать ситуацию, то получается, что класс **Employee** - это тот же класс **User**, но более расширенный.

Для решения проблемы существует такая вещь, как *наследование*.

С помощью наследования мы можем заставить наш класс **Employee** позаимствовать (*унаследовать*) методы и свойства класса **User** и просто дополнить их своими методами и свойствами.

Наследование реализуется с помощью ключевого слова **extends** (переводится как *расширяет*).

Чтобы класс **Employee** унаследовал от класса **User** следует при объявлении класса **Employee** вместо *class Employee* написать так: *class Employee extends User*.

Класс, от которого наследуют называется *родителем* (англ. parent), а класс, который наследует - *потомком*.

Класс-потомок наследует только публичные методы и свойства, но не приватные.

Итак, давайте перепишем наш класс **Employee** так, чтобы он наследовал от **User**.

Код намного сократится:

```
<?php
  class Employee extends User
  {
    private $salary;

    public function getSalary()
    {
      return $this->salary;
    }

    public function setSalary($salary)
    {
      $this->salary = $salary;
    }

  }
?>
```

Проверим работу нового класса **Employee**:

```
<?php
  $employee = new Employee;

  $employee->setSalary(1000); // метод кла
    сса Employee
  $ employee->setName('Коля'); // метод унаследован от
    родителя
  $ employee->setAge(25); // метод унаследован от
```

```
    родителя

echo $employee->getSalary(); // метод класса Emp
    loyee
e cho $employee->getName(); // метод унаследован от
    родителя
e cho $employee->getAge(); // метод унаследован от
    родителя
?>
```

Обратите внимание на следующее: класс-потомок не унаследовал от своего родителя приватные свойства **name** и **age** - попытка обратится к ним вызовет ошибку.

При этом, однако, в классе-потомке доступны геттеры и сеттеры свойств **name** и **age**, так как эти геттеры и сеттеры являются публичными.

Задача 19.1

Не подсматривая в мой код реализуйте такие же классы **User**, **Employee**.

Несколько потомков

Преимущества наследования в том, что каждый класс может несколько потомков.

Давайте посмотрим на примере.

Пусть кроме работника нам нужен еще и класс **Student** - давайте также унаследуем его от **User**:

```
<?php
  class Student extends User
  {
    private $course; // курс

    public function getCourse()
    {
      return $this->course;
    }

    public function setCourse($course)
    {
      $this->course = $course;
    }
  }
?>
```

```
<?php
  $student = new Student;

  $student->setCourse(3); // метод класса Stu
    dent
  $ student->setName('Коля'); // метод унаследован от
    родителя
  $ student->setAge(25); // метод унаследован от
    родителя

  echo $student->getCourse(); // метод класса Stu
    dent
  e cho $student->getName(); // метод унаследован от
    родителя
  e cho $student->getAge(); // метод унаследован от
    родителя
?>
```

Задача 19.2

Не подсматривая в мой код реализуйте такой же класс **Student**, наследующий от класса **User**.

Наследование от наследников

Пусть у нас есть класс-родитель и класс-потомок. От этого потомка также могут наследовать другие классы, от его потомков другие и так далее.

Посмотрим на примере.

Пусть от класса **Student** наследует класс **StudentBSU** (студент БГУ):

```php
<?php
  class StudentBSU extends Student
  {
    // добавляем свои свойства и методы
  }
?>
```

Получится, что от класса **User** наследует **Student**, а от него в свою очередь наследует класс **StudentBSU**. От **StudentBSU** также может кто-то наследовать и так далее.

Задача 19.3

Сделайте класс **Programmer**, который будет наследовать от класса **Employee**. Пусть новый класс имеет свойство **langs**, в котором будет хранится массив языков, которыми владеет программист. Сделайте также геттер и сеттер для этого свойства.

Задача 19.4

1 Сделайте класс **Driver** (водитель), который будет наследовать от класса **Employee**. Пусть новый класс добавляет следующие свойства: водительский стаж, категория вождения (A, B, C, D), а также геттеры и сеттеры к ним.