Перезапись методов родителя в классе потомке

Пусть дан класс **User** с приватными свойствами **name** и **age**, а также геттерами и сеттерами этих свойств. При этом в сеттере возраста выполняется проверка возраста на то, что он равен или больше 18 лет:

```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($age)
    {
      // Выполняем проверку возраста:
      if ($age >= 18) {
        $this->age = $age;
      }
    }
  }
?>
```

От класса **User** наследует класс **Student** с приватным свойством **course**, его геттером и сеттером:

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

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

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

Предположим теперь, что метод **setAge**, который **Student** наследует от **User** нам чем-то не подходит, например, нам нужна также проверка того, что возраст студента до 25 лет.

То есть проверка на то, что возраст более 18 лет нас устраивает, но хотелось бы иметь добавочную проверку на то, что он меньше 25.

Для решения проблемы мы используем то, что в PHP в классе-потомке разрешено сделать метод с таким же именем, как и у родителя, таким образом переопределив этот метод родителя на свой.

Как раз то, что нам нужно.

Итак, давайте напишем студент свой **setAge** в классе **Student**. Наш setAge будет проверять то, что возраст студента меньше от 18 до 25 лет:

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

    // Перезаписываем метод родителя:
    public function setAge($age)
    {
      if ($age >= 18 and $age <= 25) {
        $this->age = $age;
      }
    }

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

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

Так как наш метод **setAge** использует свойство **age** от родителя, то в родителе это свойство надо объявить как защищенное:

```php
<?php
  class User
  {
    private $name;
    protected $age; // изменим модификатор доступа на
      protected

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

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

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

    public function setAge($age)
    {
      if ($age >= 18) {
        $this->age = $age;
      }
    }
  }
?>
```

Давайте проверим работу переопределенного метода **setAge**:

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

  $student->setAge(24); // укажем корректный воз
    раст
  e cho $student->getAge(); // выведет 24 - воз
    раст поменялся

  $student->setAge(30); // укажем НЕкорректный воз
    раст
  e cho $student->getAge(); // выведет 24 - воз
    раст НЕ поменялся
?>
```

Работа с parent

Сейчас в нашем новом методе **setAge** мы выполняем проверку того, что возраст от 18 до 25 лет.

Однако, проверку, того, что возраст от 18 лет выполняет и метод **setAge**.

Это значит, что если мы захотим изменить нижнюю границу возраста - нам придется сделать это в двух местах: в коде класса родителя и в коде класса потомка.

Было бы удобно, если мы метод **setAge** потомка мог использовать метод **setAge** от родителя, ведь в методе родителя расположена часть кода, которая нам подходит и мы не хотим ее дублировать в методе потомка.

Такое можно сделать с помощью ключевого слова **parent**, указывающего на родителя.

С его помощью к переопределенному методу родителя можно обратиться так: **parent::setAge()**, то есть ключевое слово **parent**, затем два двоеточия и сам метод.

Давайте доработаем наш класс **Student** так, чтобы использовался метод **setAge** родителя:

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

    public function setAge($age)
    {
      // Если возраст меньше или равен 25:
      if ($age <= 25) {
        // Вызываем метод родителя:
        parent::setAge($age); // в родителе вып
          олняется проверка age >= 18
      }
    }

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

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

Мы добились того, что хотели. Более того, теперь метод **setAge** потомка не использует свойство **age** напрямую. Это значит, что в классе-родителе можно поменять его модификатор доступа с **protected** обратно на **private**.

Задача 21.1

Модифицируйте код класса **User** так, чтобы в методе **setName** выполнялась проверка того, что длина имени более 3-х символов.

Задача 21.2

Добавьте в класс **Student** метод **setName**, в котором будет выполняться проверка того, что длина имени более 3-х символов и менее 10 символов.

Пусть метод **setName** класса **Student** использует метод **setName** своего родителя, чтобы выполнить часть проверки.