Обращение к методам через $this

Через **$this** можно обращаться не только к свойствам объекта, но и к его методам. Думаю это уже более-менее понятно, поэтому давайте сразу посмотрим применение на практическом примере.

Пусть у нас есть класс **User**, а в нем метод **setAge** для изменения возраста юзера:

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

    // Метод для изменения возраста юзера:
    public function setAge($age)
    {
      $this->age = $age;
    }
  }
?>
```

Давайте добавим проверку введенного возраста: если он от 18 до 60, то будем менять возраст на новый, а если нет - то менять не будем:

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

    // Метод для изменения возраста юзера:
    public function setAge($age)
    {
      // Если возраст от 18 до 60:
      if ($age >= 18 and $age <= 60) {
        $this->age = $age;
      }
    }
  }
?>
```

Пусть также нам нужен метод **addAge**, который будет добавлять некоторое количество лет к текущему возрасту:

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

    // Метод для изменения возраста юзера:
    public function setAge($age)
    {
      // Если возраст от 18 до 60:
      if ($age >= 18 and $age <= 60) {
        $this->age = $age;
      }
    }

    // Метод для добавления к возрасту:
```

```
17      public function addAge($years)
18      {
19        $this->age = $this->age + $years;
20      }
21    }
22  ?>
```

В метод **addAge**, конечно же, также необходимо добавить проверку возраста, сделаем это:

```
1   <?php
2     class User
3     {
4       public $name;
5       public $age;
6
7       // Метод для изменения возраста юзера:
8       public function setAge($age)
9       {
10        // Если возраст от 18 до 60:
11        if ($age >= 18 and $age <= 60) {
12          $this->age = $age;
13        }
14      }
15
16      // Метод для добавления к возрасту:
17      public function addAge($years)
18      {
19        $newAge = $this->age + $years; // вычислим нов
            ый возраст
20
21        // Если НОВЫЙ возраст от 18 до 60:
22        if ($newAge >= 18 and $newAge <= 60) {
23          $this->age = $newAge; // обновим, если нов
              ый возраст прошел проверку
24        }
25      }
26    }
27  ?>
```

Получится, что ограничения на возраст теперь задаются в двух местах (в функции **setAge** и в функции **addAge**), что не очень хорошо: если мы захотим поменять ограничение, нам придется сделать это в двух местах - это неудобно, и в каком-то из мест мы можем забыть внести изменения - это опасно, ведь наш код получится с трудноуловимой ошибкой.

Давайте вынесем проверку возраста на корректность в отдельный *вспомогательный* метод **isAgeCorrect**, в который параметром будет передаваться возраст для проверки, и используем его внутри методов **setAge** и **addAge**:

```
1   <?php
2     class User
3     {
4       public $name;
5       public $age;
6
7       // Метод для проверки возраста:
8       public function isAgeCorrect($age)
9       {
10        if ($age >= 18 and $age <= 60) {
11          return true;
12        } else {
13          return false;
14        }
15      }
16
17      // Метод для изменения возраста юзера:
18      public function setAge($age)
19      {
20        // Проверим возраст на корректность:
21        if ($this->isAgeCorrect($age)) {
```

```
      $this->age = $age;
    }
  }

  // Метод для добавления к возрасту:
  public function addAge($years)
  {
    $newAge = $this->age + $years; // вычислим нов
      ый возраст

    // Проверим возраст на корректность:
    if ($this->isAgeCorrect($newAge)) {
      $this->age = $newAge; // обновим, если нов
        ый возраст прошел проверку
    }

  }
}
?>
```

Теперь любое изменения в условиях проверки можно будет легко сделать в одном месте.

Кстати, более изящно будет переписать метод **isAgeCorrect** на короткую форму работы с логическими выражениями, вот так:

```
<?php
  public function isAgeCorrect($age)
  {
    return $age >= 18 and $age <= 60;
  }
?>
```

Исправим наш код:

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

    // Метод для проверки возраста:
    public function isAgeCorrect($age)
    {
      return $age >= 18 and $age <= 60;
    }

    // Метод для изменения возраста юзера:
    public function setAge($age)
    {
      // Проверим возраст на корректность:
      if ($this->isAgeCorrect($age)) {
        $this->age = $age;
      }
    }

    // Метод для добавления к возрасту:
    public function addAge($years)
    {
      $newAge = $this->age + $years; // вычислим нов
        ый возраст

      // Проверим возраст на корректность:
      if ($this->isAgeCorrect($newAge)) {
        $this->age = $newAge; // обновим, если нов
          ый возраст прошел проверку
      }

    }
  }
?>
```

Проверим, что все работает как надо:

```php
<?php
  $user = new User;

  $user->setAge(30); // установим возраст в 30
  echo $user->age; // выведет 30

  $user->setAge(10); // установим некорректный возраст
  echo $user->age; // не выведет 10, а выв
    едет 30
?>
```

Задача 5.1

Не подсматривая в мой код создайте такой же класс **User** с такими же методами.

Задача 5.2

Создайте объект этого класса **User**, проверьте работу методов **setAge** и **addAge**.

Задача 5.3

Добавьте также метод **subAge**, который будет выполнять уменьшение текущего возраста на переданное количество лет.