

Аннотация: этот документ является практическим руководством к в Coding Bootcamp.

Week 1 Day 3

**Академия Ковалевского**

[**1. Теория**](#_30j0zll) **3**

[**2. Практическая работа**](#_gqxts5hxfb6r) **4**

[Обновленный StdString (теперь банановый)](#_3znysh7) 4

[StdString2(char[] base)](#_2jl5qcbr7cj) 4

[StdString2()](#_jibahdfuneif) 4

[StdString2(StdString that)](#_d7fax38yshhq) 5

[toAsciiLowerCase](#_s2s14r96v8wt) 5

[toAsciiUpperCase](#_dobo3yiu19y) 5

[subString](#_2odgdh38313u) 6

[concat](#_8xdb919tmefd) 6

[split](#_99daf3vycgxg) 7

[trim](#_62z82m75izqp) 7

[removeCharacter](#_2pixg4vdctn5) 8

# 

# 1. Теория

В данном занятии рекомендуется ознакомится с такими топиками как:

* Наследование ([видео](https://www.youtube.com/watch?v=_rxR-UHzFxA&list=PLsQAG1V_t58AKvV5v4NVXxo68OyLdNX3j&index=14))
* Ключевое слово **super** ([пример статьи](https://www.examclouds.com/java/java-core-russian/keyword-super))
* Аргументы переменной длины ([пример статьи](https://www.bestprog.net/ru/2018/12/23/variable-length-arguments-in-parameters-overloading-methods-with-variable-length-arguments-property-length-examples-ru/))

Напоминаем, что ссылки тут на внешние источники просто примеры, крайне рекомендуем самостоятельно поискать материалы по интересующим темам, чтобы все в группе смотрели разные источники по теме.

# 2. Практическая работа

## Обновленный StdString (теперь банановый)

Запрещено: использовать готовые реализации методов.

Разрешено: использовать свои реализации из предыдущих дней, StringBuilder.

Чтобы добавить новые возможности в наш класс, мы создадим класс наследник вот с такой сигнатурой:

|  |
| --- |
| package com.kovalevskyi.academy.codingbootcamp.week1.day3; |

|  |
| --- |
| public class StdString2 extends StdString1 {  // TO DO } |

### StdString2(char[] base)

Конструктор, который создает новую строку на базе переданного массива символов.

|  |
| --- |
| *// сигнатура конструктора*  public StdString2(char[] base) {  // TODO } |

### StdString2()

Пустой конструктор, который создает пустую строку.

|  |
| --- |
| *// сигнатура конструктора*  public StdString2() {  // TODO } |

### StdString2(StdString that)

Конструктор копирования, который создает строку идентичную той, что передана на вход.

|  |
| --- |
| *// сигнатура конструктора*  public StdString2(StdString2 that) {  // TODO } |

### toAsciiLowerCase

Создает копию строки, в которой все символы нижнего регистра. Например:

* *"CaT12" → "cat12"*
* *“ asdT2" → " asdt2".*

Если символ не ASCII, то метод должен кинуть исключение *IllegalArgumentException*.

|  |
| --- |
| *// сигнатура метода*  public StdString2 toAsciiLowerCase() {  // TODO } |

### toAsciiUpperCase

Создает копию строки, в которой все символы верхнего регистра. Например:

* *"CaT12" → "CAT12"*
* *“ asdT2" → " ASDT2"*

Если символ не ASCII, то метод должен кинуть исключение *IllegalArgumentException*.

|  |
| --- |
| *// сигнатура метода*  public StdString2 toAsciiUpperCase() {  // TODO } |

## 

### subString

Возвращает подстроку, начиная с заданного индекса (включительно) по заданный индекс (не включительно).

Например:

* *"ased 79s", 1 5 → "sed "*
* "cat-dog-123", 2 3 *→* "t"

В случае, если какой-либо из индексов выходит за пределы строки, метод кидает *IndexOutOfBoundsException*. В случае, если аргументы неверные (например, начальный индекс больше конечного), метод кидает *IllegalArgumentException.*

|  |
| --- |
| *// сигнатура метода*  public StdString2 subString(int from, int to) {  // TODO } |

### concat

Метод создает новую строку, в которую входят все символы текущей строки, а также все символы каждой строки, которая была передана на вход. Например:

* *"thisStr", "thatStr123", "thatStr222" → "thisStrthatStr123thatStr222"*

|  |
| --- |
| *// сигнатура метода*  public StdString2 concat(StdString2... that) {  // TODO } |

## 

### split

Метод делит текущую строку на подстроки в соответствии с заданным делителем и возвращает массив этих строк. Например:

* *"cat trim dog", ' ' → "cat", "trim", "dog"*
* *"one!str !two ! three", '!' → "one", "str ", "two ", " three"*
* *"cat dog ", ' ' → "cat", "dog", "", ""*
* *"cat dog ", '!' → "cat dog "*

|  |
| --- |
| *// сигнатура метода*  public StdString2[] split(char separator) {  // TODO } |

### trim

Метод создает новую строку, которая похожа на текущую, но только без пробелов в начале и в конце строки (если они там вообще есть). Например:

* *" hi " → "hi"*
* *"hi" → "hi"*
* *" 1 2" → "1 2"*
* *" " → ""*

|  |
| --- |
| *// сигнатура метода*  public StdString2 trim() {  // TODO } |

## 

### removeCharacter

Метод создает новую строку, которая равна текущей, однако без определенного символа, который передан на вход. Например:

* *"new,1,2,3" ',' → "new123"*
* *" 1 2 ", ' ' → "12"*
* *"asdf123", '!' → "asdf123"*

|  |
| --- |
| *// сигнатура метода*  public StdString2 removeCharacter(char toRemove) {  // TODO } |