НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Алгоритмизация и программирование

Программирование на Kotlin (Строки)

Беркунский Е.Ю., кафедра ИУСТ, НУК
eugeny.berkunsky@gmail.com
http://www.berkut.mk.ua

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Лекция – набор примеров

Строки в Kotlin

- Класс String представляет символьные строки.
- Все строковые литералы в программах Kotlin, такие как «abc», реализованы как экземпляры этого класса.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Длина строки

- У строк есть свойство «length» - длина

```
val s = "Admiral Makarov National University of Shipbuilding"
val len = s.length
println("Длина строки = $len")
```

```
Длина строки = 51
```

Функции строк

```
fun compareTo(other: String): Int
```

Сравнивает этот объект с указанной строкой:

- Возвращает ноль, если эта строка равна указанной другой строке,
- отрицательное число, если она «меньше» другой,
- или положительное число, если она «больше» другой.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Функции строк

```
fun equals(other: Any?): Boolean
```

Указывает, равен ли какой-либо другой объект этому.
Для строк (и большинства других объектов в Kotlin) можно использовать ==

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Функции строк

```
fun get(index: Int): Char
```

Возвращает символ этой строки по указанному индексу. Если индекс выходит за пределы этой строки, выдает исключение IndexOutOfBoundsException

Можно использовать в форме […]

Строковые шаблоны

Поздороваемся с котом по имени «Барсик»

```
val catName: String = "Барсик"
println("Привет $catName! Как дела?")
```

Естественно, можно и так:

```
val catName = "Барсик"
```

Функции-расширения

- Разработчики JetBrains добавили множество готовых функций-расширений для многих классов, в том числе и для строк. Найти их можно в файле String.kt (в IntelliJ IDEA дважды нажмите клавишу Shift и в поисковой строке наберите имя данного файла для просмотра исходника).

- Некоторые примеры функций-расширений будут рассмотрены далее. На самом деле их гораздо больше, изучайте их самостоятельно.

Строка как массив

- Строку можно рассматривать как массив символов.

```
val cat = "Барсик"
val character = cat[2]
println(character) // выводит р
```

Пробежаться по строке

- Пробежаться по всей строке без использования индекса

```
val cat = "Барсик"

for(char in cat){
    println(char)
}
```

- Пробежаться по всей строке с использованием индекса

```
val cat = "Барсик"

for (char in cat.indices){
    print(cat[char] + "\n")
}
```

Пробежаться по строке

- Саму строку можно предварительно явно преобразовать в массив

```
val cat = "Барсик"
for(char in cat.toCharArray){
    println(char)
}
```

- Можно вызывать forEach

```
cat.forEach { char -> println(char) }
```

- Если вам нужен не только символ строки, но и его индекс, то вызываем forEachIndexed

```
cat.forEachIndexed {
   index, char -> println("Index $index Character $char")
}
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Встроенные функции

```
    val blank = "   ".isBlank()
// true, если пустая строка или содержит только символы
// пробела, табуляции и т.п.

// индекс последнего символа
    val lastIndex = "Кот Мурзик".lastIndex // 9

// переводим в верхний регистр первый символ строки
// decapitalize() выполняем обратную задачу
    val capitalize = "кот Мурзик".capitalize()

// добавляем пробел перед строкой
    val withSpaces = "1".padStart(2)

// "100"  добавляем нули в конец
    val endZeros = "1".padEnd(3, '0')
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Встроенные функции

```
    val dropStart = "Kotlin".drop(2) // "tlin"
// убираем первые символы в указанном количестве
    val dropEnd = "Kotlin".dropLast(3) // "Kot"
// убираем последние символы в указанном количестве

// возвращаем строку без первого символа,
// который удовлетворяет условию
    val string = "Мурзик"
    val result = string.dropWhile{
        it == 'М'
    }
    println(result) // урзик

// возвращаем строку без последнего символа,
// который удовлетворяет условию
    val string = "Мурзик"
    val result = string.dropLastWhile{
        it == 'к'
    }
    println(result) // Мурзи
```

Встроенные функции

```
    // разбиваем на массив строк
    "A\nB\nC".lines() // [A, B, C]
    "ABCD".zipWithNext() // [(A, B), (B, C), (C, D)]

// удаляем символы из заданного диапазона
    val string = "Кот, который гулял сам по себе"
    val result = string.removeRange(
        3..28 // range
    )
```

Встроенные функции

```
// удаляем префикс из строки
    val string = "Кот, который гулял сам по себе"
    val result = string.removePrefix("Кот")

// удаляем суффикс из строки
    val string = "Кот, который гулял сам по себе"
    val result = string.removeSuffix("себе")

// удаляем заданный разделитель,
// который должен окружать строку с начала и с конца
    val string = "та, тра-та-та, мы везём с собой кота"

    val result = string.removeSurrounding(
        "та" // delimiter
    )

    println(result) // , тра-та-та, мы везём с собой ко
```

Встроенные функции

```
// Также можно указать разные начало и конец,
// которые окружают строку
val string = "Тра-та-та, тра-та-та, мы везём с собой кота"

val result = string.removeSurrounding(
    "Тра-", // prefix
    " кота" // suffix
)

println(result) // та-та, тра-та-та, мы везём с собой
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Встроенные функции

```
// Добавляем отступы при явном переводе на новую строку
val string =
   "Какой-то длинный текст, \nсостоящий из имён котов: " +
            "\nВаська" +
            "\nБарсик" +
            "\nРыжик"

val result = string.prependIndent(
    "     " // indent
)
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Встроенные функции

```
// Разбиваем символы на две группы.
// В первую группу попадут символы в верхнем регистре,
// во вторую - символы в нижнем регистре
val string = "Кот Васька и кот Мурзик - Друзья!"

val result: Pair<String, String> = string.partition {
    it.isUpperCase()
}

println(result.first + " : " + result.second)
//КВМД : от аська и кот урзик - рузья!
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Встроенные функции

```
// Разбиваем строку на список строк.
// В качестве разделителя - перевод на новую строку
val s1 = "Кот Васька\nКотМурзик\nКот Мурзик"

// Split string into lines (CRLF, LF or CR)
val lines: List<String> = s1.lines()

println("Кол-во строк: ${lines.size}")

// В качестве разделителя - пробел
val s2 = "Кот Васька Кот Мурзик Кот Мурзик"
val lines2: List<String> = s2.split(" ")

println("Кол-во строк: ${lines2.size}")
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Использование предикатов

```
// Содержит ли строка только цифры (используем предикат)
val string = "09032020"

// Returns true if all characters match the given predicate
val result: Boolean = string.all{
    it.isDigit()
}
```

```
// Содержит ли строка хотя бы одну цифру
// (используем предикат)
val string = "3 кота"
// Returns true if at least one character matches the
// given predicate
val result: Boolean = string.any() {
    it.isDigit()
}
```

- Для замены отдельных символов или строк используется функция **replace()**

```
val s1 = "Кіт Кішка"

val r1 = s1.replace(
    'i', // old char
    'o', // new char
    true // ignore case Boolean = false
)
```

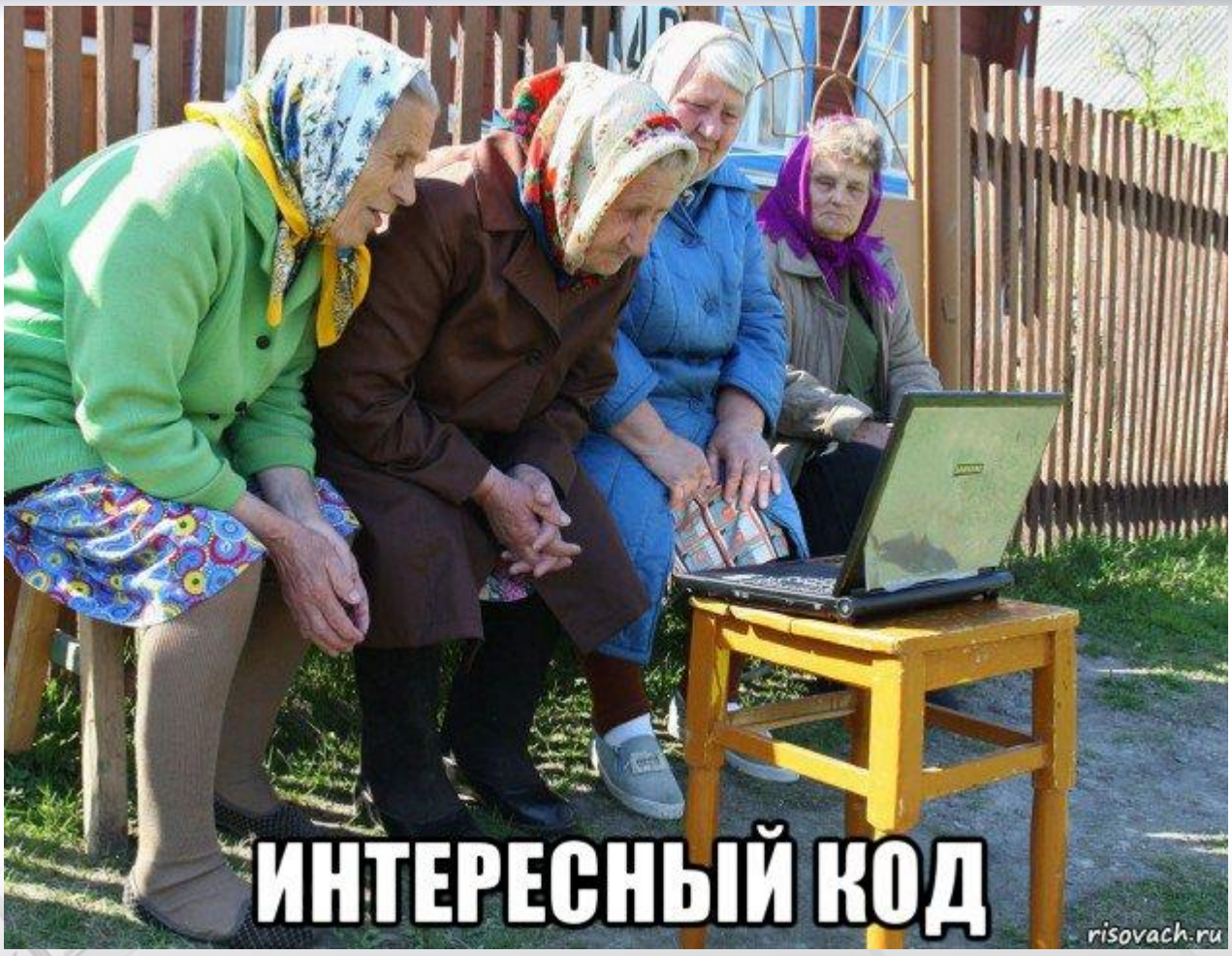
```
val str = "Собака - друг человека"
val res = string.replace(
    "Собака", // old value
    "Кот", // new value
    true // ignore case
)
```

```
fun toDragonSpeak(phrase: String) =
    phrase.replace(Regex("[aeiou]")) {
        when (it.value) {
            "a" -> "4"
            "e" -> "3"
            "i" -> "1"
            "o" -> "0"
            "u" -> "|_|"
            else -> it.value
        }
    }
```

```
println(toDragonSpeak("Kitten")) // K1tt3n
```

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Демонстрация



НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ ІМЕНІ АДМІРАЛА МАКАРОВА

Алгоритмизация и программирование

Программирование на Kotlin (Строки)

Беркунский Е.Ю., кафедра ИУСТ, НУК
eugeny.berkunsky@gmail.com
http://www.berkut.mk.ua