





## Fuzzing

Подключили библиотеку JQF для проведения случайного тестирования. Подключение проводилось с помощью maven, где мы указали необходимые dependencies и plugins. Конечный maven file можно увидеть в проекте (pom.xml)

Для запуска fuzzing'a мы используем команду `mvn clean verify`, также помимо фаззинга у нас собирается информация с помощью `jacoco` о покрытии кода тестами.

Результаты покрытия мы можем наблюдать ниже:

### Lesson 5

Method	Line Coverage	Branch Coverage	Instruction Coverage	Block Coverage
<a href="#">canBuildFrom(List, String)</a>		100%		100%
<a href="#">containsIn(Map, Map)</a>		100%		50%
Total	641 of 700	9%	59 of 62	10%

В целом фаззинг справился неплохо, покрытие инструкций на 100%, но в случае `containsIn` мы не зашли в одну из веток :-)

### CanbuildFrom

```
17. */
18. fun canBuildFrom(chars: List<Char>, word: String): Boolean =
19.     word.lowercase().toSet().all { letter -> letter in chars.map { it.lowercaseChar() } }
20.
21. /**
22.  * Средняя (4 балла)
```

### ContainsIn

```
0. */
1. fun containsIn(a: Map<String, String>, b: Map<String, String>): Boolean = a.all { it.value == b[it.key] }
2.
```

### Lesson 6

<a href="#">dateDigitToChar(String)</a>		0%		0%
<a href="#">dateStrToDigit(String)</a>		9%		16%
<a href="#">isDayCorrect(int, int, int)</a>		0%		0%
<a href="#">flattenPhoneNumber(String)</a>		0%		0%
<a href="#">plusMinus(String)</a>		45%		43%

Здесь покрытие значительно хуже, методы, которые тестировались с помощью фаззинга выделены красной рамкой. Низкое покрытие обуславливается тем, что с помощью сгенерированного датасета не получилось пройти по всем инструкциям и ветвям.

### DateStrToDigit

```

80. */
81. fun dateStrToDigit(str: String): String {
82.     if (!str.matches(Regex("""\d+ ([a-я])+ \d+"""))) return ""
83.     val list = str.split(' ')
84.     val day = list[0].toInt()
85.     var month = list[1]
86.     val year = list[2].toInt()
87.     val months = mapOf(
88.         "января" to "01",
89.         "февраля" to "02",
90.         "марта" to "03",
91.         "апреля" to "04",
92.         "мая" to "05",
93.         "июня" to "06",
94.         "июля" to "07",
95.         "августа" to "08",
96.         "сентября" to "09",
97.         "октября" to "10",
98.         "ноября" to "11",
99.         "декабря" to "12"
100.    )
101.    if (month !in months) return "" else month = months[month].toString()
102.    return if (isDayCorrect(day, month.toInt(), year))
103.        String.format("%02d.%s.%d", day, month, year) else ""
104. }
105.
106. fun isDayCorrect(day: Int, month: Int, year: Int): Boolean {
107.     val limits = mutableMapOf<Int, Int>()
108.     for (i in listOf(1, 3, 5, 7, 8, 10, 12)) limits += i to 31
109.     for (i in listOf(4, 6, 9, 11)) limits += i to 30
110.     limits += if ((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0) 2 to 29 else 2 to 28
111.     return day in 1..limits[month]!!
112. }
113.
114. /**

```

## PlusMinus

```

101. * При нарушении формата входной строки бросить исключение IllegalArgumentException
102. */
103. fun plusMinus(expression: String): Int {
104.     if (expression.contains(Regex("""[+-] |\d |\d^[+-]""")))
105.         throw IllegalArgumentException()
106.     val stripped = expression.filter { it != ' ' }
107.     val numbers = Regex("[+-]").split(stripped)
108.     val move = stripped.filter { it in "+-" }.toList()
109.     var total = numbers[0].toInt()
110.     for (m in move.indices) {
111.         when (move[m]) {
112.             '+' -> total += numbers[m + 1].toInt()
113.             '-' -> total -= numbers[m + 1].toInt()
114.         }
115.     }
116.     return total
117. }
118.

```

## Lesson7

		88%		85%
Total	1 007 of 1 200	15%	61 of 74	17%

Здесь покрытие кода лучше, чем в lesson 6, соответственно в данном случае fuzzing смог попасть данными в нужные инструкции и разветвления

ChooseLongestChaoticWord

```

258.  * Обратите внимание: данная функция не имеет возвращаемого значения
259.  */
260. fun chooseLongestChaoticWord(inputName: String, outputName: String) {
261.     val chosen = mutableListOf("")
262.     File(inputName).forEachLine { w ->
263.         val wLow = w.lowercase()
264.         if (wLow.all { char -> wLow.count { it == char } == 1 }) {
265.             if (w.length == chosen[0].length) chosen += w
266.             if (w.length > chosen[0].length) {
267.                 chosen.clear()
268.                 chosen += w
269.             }
270.         }
271.     }
272.     File(outputName).bufferedWriter().use { it.write(chosen.joinToString()) }
273. }
274.
275. /**
276.  * Сложная (22 балла)
277.  *

```