

МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Кафедра 319 «Системы интеллектуального мониторинга»

**ЛАБОРАТОРНАЯ РАБОТА №2**

по дисциплине «Программирование»

Тема: регулярные выражения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | | |  | Андрианова Е.А. |
| Группа | | |  |  |
| Проверил | | |  | Шилов В.В. |
| Оценка |  | Дата защиты « \_ » 2021 г. | | |

**Москва 2021**

МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Кафедра 319 «Системы интеллектуального мониторинга»

**З А Д А Н И Е**

на лабораторную работу №3 по дисциплине

«Программирование»

Студент \_\_М3О-135Б-20, Андрианова Елена Андреевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(№ группы, Ф. И. О.)

Тема \_\_\_\_\_\_\_\_\_\_\_\_ регулярные выражения \_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень вопросов, подлежащих разработке в лабораторной работе

Изучить механизм и принципы регулярного выражения.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Решить задачи с помощью регулярных выражений. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Разработать пояснительную записку\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рекомендуемая литература

1. Герберт Шилдт. Java. Полное руководство, 10-е изд. : Пер. с англ. – СПб. ООО "Альфакнига', 2018. – 1488 с. ISBN 978-5-6040043-6-4

2. Бертран Мейер. Объектно-ориентированное конструирование программных системю: Русская Редакция; 2015. – 768 с. ISBN 5-7502-0255-0\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Билл Смит Методы и алгоритмы вычислений на строках (regexp) = Computing Patterns in Strings. – М.: «Вильямс», 2006. – С. 496. – ISBN 0-201-39839-7 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Задание выдано «27» февраль 2021 г.

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*Шилов В.В* \_\_\_\_

(Ф. И. О., подпись)

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

## СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc71926048)

[ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 6](#_Toc71926049)

[ПРОЦЕСС РАЗРАБОТКИ 11](#_Toc71926050)

[ПРОЦЕСС ТЕСТИРОВАНИЯ 14](#_Toc71926051)

[ЗАКЛЮЧЕНИЕ 15](#_Toc71926052)

[СПИСОК ЛИТЕРАТУРЫ 16](#_Toc71926053)

## ВВЕДЕНИЕ

В этой лабораторной работе мы знакомимся с правилами написания регулярных выражений, классами Pattern и Matcher. Для изучения данного материала мы пишем несколько регулярных выражений на языке Java.

## ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Регулярное выражение (Regular Expression - RegEX) — это специальная последовательность символов, которая помогает вам сопоставлять или находить другие строки или наборы строк, используя специальный синтаксис, содержащийся в шаблоне. В Java исходным представлением этого шаблона всегда является строка, то есть объект класса String. Их можно использовать для поиска, редактирования или манипулирования текстом и данными.

Шаблон поиска может быть любым из простого символа, фиксированной строки или сложного выражения, содержащего специальные символы, описывающие шаблон. Шаблон, определенный выражением, может совпадать один или несколько раз, или не совпадать для данной строки.

Специальные символы, предназначенные для написания регулярных выражений, называются метасимволами. Вот список метасимволов в Java:

*^ начало строки*

*$ конец строки*

*\b граница слова*

*\B не граница слова*

*\A начало ввода*

*\G конец предыдущего совпадения*

*\Z конец ввода*

*\z конец ввода*

*\d цифровой символ*

*\D нецифровой символ*

*\s символ пробела*

*\S непробельный символ*

*\w буквенно-цифровой символ или знак подчёркивания*

*\W любой символ, кроме буквенного, цифрового или знака подчёркивания*

*. любой символ*

*\t символ табуляции*

*\n символ новой строки*

*\r символ возврата каретки*

*\f переход на новую страницу*

*\u 0085 символ следующей строки*

*\u 2028 символ разделения строк*

*\u 2029 символ разделения абзацев*

*[абв] любой из перечисленных (а,б, или в)*

*[^абв] любой, кроме перечисленных (не а,б, в)*

*[a-zA-Z] слияние диапазонов (латинские символы от a до z без учета регистра )*

*[a-d[m-p]] объединение символов (от a до d и от m до p)*

*[a-z&&[def]] пересечение символов (символы d,e,f)*

*[a-z&&[^bc]] вычитание символов (символы a, d-z)*

*? один или отсутствует*

*\* ноль или более раз*

*+ один или более раз*

*{n} n раз*

*{n,} n раз и более*

*{n,m} не менее n раз и не более m раз*

Эти символы являются основными инструментами для написания регулярных выражений на языке программирования Java.

Регулярные выражения в Java обеспечиваются пакетом *java.util.regex*. В пакете *java.util.regex* есть два наиболее используемых класса:

Класс Pattern

Класс Pattern используется для простой обработки строк. Для более сложной обработки строк используется класс Matcher, рассматриваемый ниже.

В классе Pattern объявлены следующие методы:

*compile*(String regex) – возвращает Pattern, который соответствует regex;

*matcher*(CharSequence input) – возвращает Matcher, с помощью которого можно находить соответствия в строке input;

*matches*(String regex, CharSequence input) – проверяет на соответствие строки input шаблону regex;

*pattern*() – возвращает строку, соответствующую шаблону;

*split*(CharSequence input) – разбивает строку input, учитывая, что разделителем является шаблон;

*split*(CharSequence input, int limit) – разбивает строку input на не более чем limit частей.

С помощью метода matches() класса Pattern можно проверять на соответствие шаблону целой строки, но если необходимо найти соответствия внутри строки, например, определять участки, которые соответствуют шаблону, то класс Pattern не может быть использован. Для таких операций необходимо использовать класс Matcher.

Класс Matcher

Начальное состояние объекта типа Matcher не определено. Попытка вызвать какой-либо метод класса для извлечения информации о найденном соответствии приведет к возникновению ошибки IllegalStateException. Для того чтобы начать работу с объектом Matcher нужно вызвать один из его методов:

*matches*() – проверяет, соответствует ли вся строка шаблону;

*lookingAt*() – пытается найти последовательность символов, начинающуюся с начала строки и соответствующую шаблону;

*find*() или *find*(int start) – пытается найти последовательность символов, соответствующих шаблону, в любом месте строки. Параметр start указывает на начальную позицию поиска.

Иногда необходимо сбросить состояние объекта класса Matcher в исходное, для этого применяется метод *reset*() или *reset*(CharSequence input), который также устанавливает новую последовательность символов для поиска.

Для замены всех подпоследовательностей символов, удовлетворяющих шаблону, на заданную строку можно применить метод *replaceAll*(String replacement).

Для того, чтобы ограничить поиск границ входной последовательности применяется метод *region*(int start, int end), а для получения значения этих границ – *regionEnd*() и *regionStart*(). С регионами связано несколько методов:

*useAnchoringBounds*(boolean b) – если установлен в true, то начало и конец региона соответствуют символам ^ и $ соответственно;

*hasAnchoringBounds*() – проверяет закрепленность границ.

В регулярном выражении для более удобной обработки входной последовательности применяются группы, которые помогают выделить части найденной подпоследовательности. В шаблоне они обозначаются скобками «(» и «)». Номера групп начинаются с единицы. Нулевая группа совпадает со всей найденной подпоследовательностью. Далее приведены методы для извлечения информации о группах:

*end*() – возвращает индекс последнего символа подпоследовательности, удовлетворяющей шаблону;

*end*(int group) – возвращает индекс последнего символа указанной группы;

*group*() – возвращает всю подпоследовательность, удовлетворяющую шаблону;

*group*(int group) – возвращает конкретную группу;

*groupCount*() – возвращает количество групп;

*start*() – возвращает индекс первого символа подпоследовательности, удовлетворяющей шаблону;

*start*(int group) – возвращает индекс первого символа указанной группы;

*hitEnd*() – возвращает истину, если был достигнут конец входной последовательности.

## ПРОЦЕСС РАЗРАБОТКИ

Коды программ: «Регулярка число длиной n», «Регулярка пароль», «Регулярка дата и время», «Регулярка цветов», «Регулярка URL», «Регулярка IPv4» представлены соответственно на рисунках 1, 2, 3, 4, 5, 6.

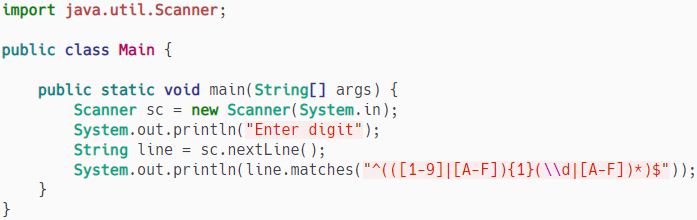


Рис. 1 – Число длиной n

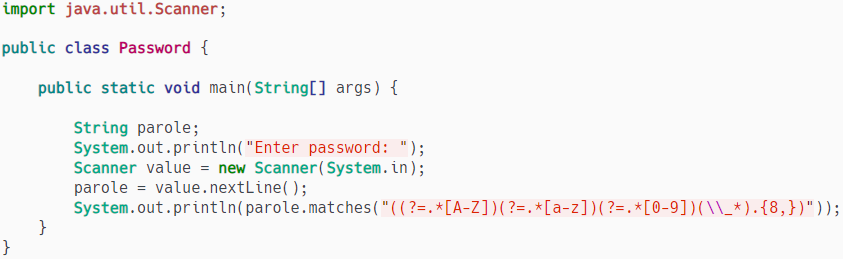


Рис. 2 – Пароль

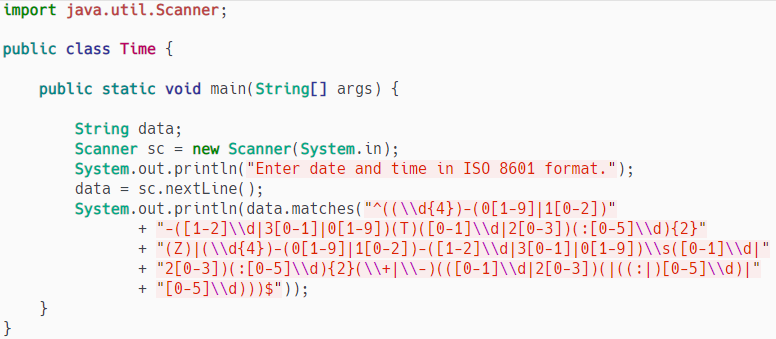


Рис.3 – Дата и время

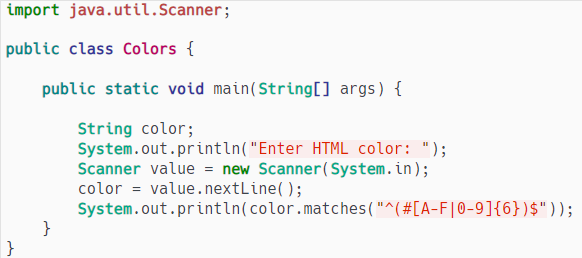


Рис. 4 – Цвета по HTML

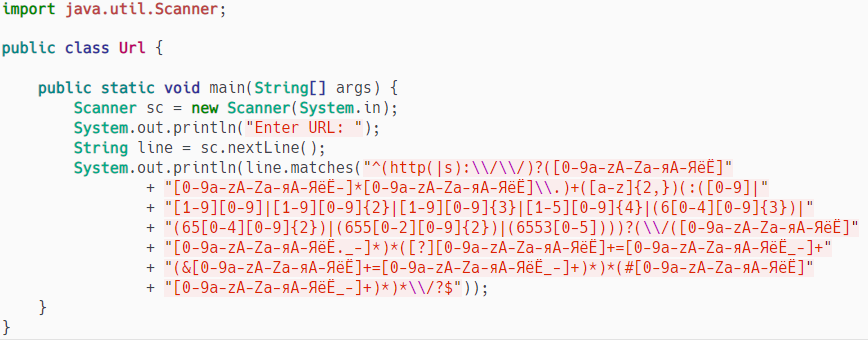


Рис. 5 – URL

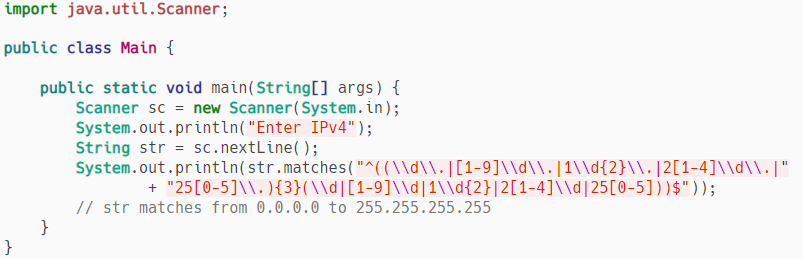


Рис. 6 - IPv4

## ПРОЦЕСС ТЕСТИРОВАНИЯ

Тесты регулярного выражения, определяющего является ли данная строчка числом в 16 системе (Таблица 1)

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | 123456F | True |
| 2 | Проверка правильности регулярного выражения | a234567 | True |
| 3 | Проверка правильности регулярного выражения | 123456w | False |
| 4 | Проверка правильности регулярного выражения | 012345 | False |

Тесты регулярного выражения, определяющего является ли пароль надежным (Таблица 2):

Таблица 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | Cool\_pass | False |
| 2 | Проверка правильности регулярного выражения | C00l | False |
| 3 | Проверка правильности регулярного выражения | coolpass123 | False |
| 4 | Проверка правильности регулярного выражения | SupperPas1 | True |

Тесты регулярного выражения, определяющего является ли строчка временем по формату ISO 8601.

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | 2021-01-00 12:12:12+10 | False |
| 2 | Проверка правильности регулярного выражения | 2021-01-11 12:12:12+1076 | False |
| 3 | Проверка правильности регулярного выражения | 2021-01-11 12:12:12+10:1 | False |
| 4 | Проверка правильности регулярного выражения | 2021-01-11 12:12:12+10 | True |

Тесты регулярного выражения, определяющего является ли данная строчка цветом в HTML (Таблица 4):

Таблица 4

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | 123 | False |
| 2 | Проверка правильности регулярного выражения | #0000000000 | False |
| 3 | Проверка правильности регулярного выражения | #Yt68EA | False |
| 4 | Проверка правильности регулярного выражения | #FFFFFF | True |

Тесты регулярного выражения, определяющего является ли данная строка валидным URL адресом (Таблица 5):

Таблица 5.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | 123 | False |
| 2 | Проверка правильности регулярного выражения | https://123 | False |
| 3 | Проверка правильности регулярного выражения | 123.com | False |
| 4 | Проверка правильности регулярного выражения | https://www.hostedredmine.com/issues | True |

Тесты регулярного выражения, определяющего является ли данная строка правильным IPv4-адресом (Таблица 6):

Таблица 6.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назначение теста | Входные данные | Реакция программы |
| 1 | Проверка правильности регулярного выражения | 123 | False |
| 2 | Проверка правильности регулярного выражения | 174.K.103.2E6 | False |
| 3 | Проверка правильности регулярного выражения | 074.2.103.246 | False |
| 4 | Проверка правильности регулярного выражения | 174.2.103.246 | True |

## ЗАКЛЮЧЕНИЕ

Создание своих регулярных выражений позволили нам узнать основные функции, принципы их написания на языке Java, показали нам возможность их применения.

## СПИСОК ЛИТЕРАТУРЫ

1. Герберт Шилдт. Java. Полное руководство, 10-е изд. : Пер. с англ. – СПб. ООО "Альфакнига', 2018. – 1488 с. ISBN 978-5-6040043-6-4

2. Бертран Мейер. Объектно-ориентированное конструирование программных системю: Русская Редакция; 2015. – 768 с. ISBN 5-7502-0255-0

3. Билл Смит Методы и алгоритмы вычислений на строках (regexp) = Computing Patterns in Strings. – М.: «Вильямс», 2006. – С. 496. – ISBN 0-201-39839-7