МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ
Ордена трудового Красного Знамени федеральное государственное бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математическая кибернетика и информационные технологии

Лабораторная работа №5

Строки и регулярные выражения

Выполнил: студент группы ___БВТ2402___
Голиков Михаил Вячеславович
Руководитель: _Мосева Марина Сергеевна_

Москва, 2077

Цель работы

Целью лабораторной работы является изучение и практическое применение строковых методов и регулярных выражений в языке программирования Java.

В ходе выполнения лабораторной работы студент осваивает основы работы с классами String, Pattern, Matcher, а также принципы составления регулярных выражений для поиска, проверки и обработки текстовой информации. Дополнительно изучаются методы обработки исключений, возникающих при некорректной работе с текстом и шаблонами.

Индивидуальное задание

В рамках лабораторной работы необходимо реализовать пять программ, каждая из которых использует регулярные выражения для решения конкретной задачи.

Задание 1. Поиск всех чисел в тексте

Написать программу, которая будет находить **все числа** в заданном тексте и выводить их на экран.

Для поиска чисел необходимо использовать **регулярные выражения**. Программа должна обрабатывать возможные ошибки, связанные с некорректным текстом или шаблоном.

Задание 2. Проверка корректности ввода пароля

Написать программу, которая будет **проверять корректность введённого пароля**.

Пароль должен соответствовать следующим критериям:

- состоять только из латинских букв и цифр;
- содержать от 8 до 16 символов;
- включать как минимум **одну заглавную букву** и **одну цифру**.

Для проверки необходимо использовать регулярные выражения и обеспечить обработку ошибок, связанных с неверным форматом пароля или отсутствием ввода.

Задание 3. Поиск заглавной буквы после строчной

Написать программу, которая будет находить все случаи, когда **сразу после строчной буквы идёт заглавная**, без пробелов и других символов между ними.
Каждый найденный случай необходимо выделить **знаками "!"** с двух сторон.

Программа должна использовать регулярные выражения для поиска таких сочетаний и корректно обрабатывать ошибки при анализе текста.

Задание 4. Проверка корректности ввода IP-адреса

Написать программу, которая будет **проверять правильность IP-адреса**.
Условия корректного IP:

- состоит из **четырёх чисел**, разделённых точками;
- каждое число находится в диапазоне **от 0 до 255**.

Для проверки использовать регулярное выражение. Необходимо также обработать возможные ошибки — например, при вводе строки, не соответствующей шаблону.

Задание 5. Поиск всех слов, начинающихся с заданной буквы

Написать программу, которая будет **искать все слова, начинающиеся с заданной буквы**, и выводить их на экран.
Буква задаётся пользователем.
Для поиска использовать регулярные выражения, а также реализовать обработку ошибок при вводе и поиске.

(Вариант 6).

Основная часть

В процессе выполнения лабораторной работы были разработаны пять программ, каждая из которых демонстрирует применение регулярных выражений в различных задачах обработки текста.

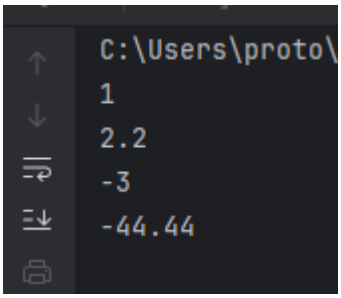
1. **NumberFinder** — выполняет поиск чисел в строке и выводит их на экран. Использует класс Pattern для компиляции шаблона и Matcher для поиска совпадений.
   Реализована обработка ошибок при некорректных входных данных.
2. **PasswordValidator** — проверяет соответствие введённого пароля заданным требованиям по длине, составу и структуре. В случае ошибок выводит поясняющие сообщения.
3. **UpperAfterLowerFinder** — находит и выделяет комбинации символов, где за строчной буквой следует заглавная. Демонстрирует возможности поиска и замены по шаблону.
4. **IPValidator** — проверяет корректность IP-адреса, используя строгое регулярное выражение для чисел от 0 до 255. При неверном вводе сообщает пользователю об ошибке.
5. **WordFinder** — реализует поиск всех слов, начинающихся с определённой буквы, заданной пользователем. Позволяет гибко работать с текстом любой длины.

Все программы используют механизм обработки исключений try-catch-finally, что обеспечивает надёжную работу при ошибках ввода и предотвращает аварийное завершение программ.

```
package Lab_5;

import java.util.regex.*;

public class NumberFinder {
    public static void main(String[] args){
        try {
            String text = "Hi    a12 1 2.2 -3 -44.44    5.6.7    8..9
.0.";
```

```
            Pattern pattern = Pattern.compile("(?<![\\w._])-
?\\d+(\\.\\d+)?(?![\\w.-_])");

            Matcher matcher = pattern.matcher(text);
            while (matcher.find()) {
                String group = matcher.group();
                //if (!group.equals("")){
                System.out.println(matcher.group());
                //}
            }
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Элемент 1 — Код программы 1

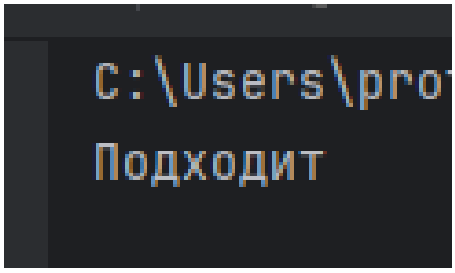


Элемент 2 — Результат программы 1

```
package Lab_5;

import java.util.regex.*;

public class PasswodValidation {

    public static void main(String[] args) {
        try {
            String password = "1234A-56789";
            Check_password(password);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    public static void Check_password(String password) throws Exception {
        if (password.isEmpty()) {
            throw new Exception("Пустой пароль");
        }

        Pattern pattern = Pattern.compile("^(?=.*[A-
Z])(?=.*\\d).[^\\s]{8,16}$");
        Matcher matcher = pattern.matcher(password);

        if (matcher.find()) {
            System.out.println("Подходит");
        } else {
            System.out.println("Не подходит");
        }
```

```
    }
}
```

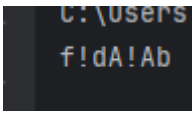
Элемент 3 — Код программы 2



Элемент 4 — Результат программы 2

```
package Lab_5;


import java.util.regex.*;

public class NumberThree {
    public static void main(String[] args){
        String text = "fdAAb";
        try {
            System.out.println(FindCombination(text));
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    public static String FindCombination(String text) throws Exception{
        Pattern pattern = Pattern.compile("([a-z])([A-Z])");
        Matcher matcher = pattern.matcher(text);
        String result = matcher.replaceAll("!$1$2!");
        return result;
    }
}
```

Элемент 5 — Код программы 3

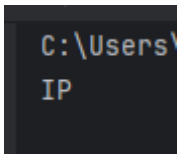


Элемент 6 — Результат программы 3

```
package Lab_5;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class IPDetector {
    public static void main(String[] args){
        String text = "0.0.0.0";
```

```
        try {
            if(FindIP(text)){
                System.out.println("IP");
            }
            else{
                System.out.println("He IP");
            }
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    } // (([0-9])|([1-9][0-9])|(1[0-9][0-9])|(2([0-4][0-9]|5[0-5])))
    // "^((chislo)\\.){3}(chislo)$"
    public static boolean FindIP(String text) throws Exception{
        Pattern pattern = Pattern.compile("^((((\\d)|([1-
9]\\d)|(1\\d{2})|(2([0-4]\\d|5[0-5])))\\.){3}((\\d)|([1-
9]\\d)|(1\\d{2})|(2([0-4]\\d|5[0-5]))))$");
        Matcher matcher = pattern.matcher(text);

        return matcher.find();
    }
}
```

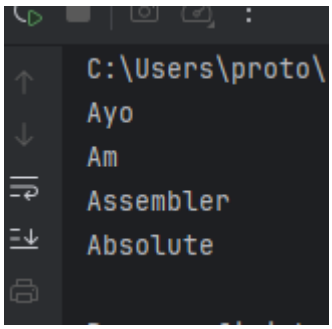
Элемент 7 — Код программы 4



Элемент 8 — Результат программы 4

```
package Lab_5;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class NumberFive {
    public static void main(String[] args){
        String text = "Ayo everyone. Am I the best Progger-Assembler -
absolute";
        String letter = "A";
        try {
            FindAlpha(text, letter);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    public static void FindAlpha(String text, String letter) throws
Exception{
        String regexpression = "\\b[" + letter.toLowerCase() +
letter.toUpperCase() + "]\\w*\\b";
        Pattern pattern = Pattern.compile(regexpression);
        Matcher matcher = pattern.matcher(text);
        while (matcher.find()){
            System.out.println(matcher.group());
        }
    }
}
```

Элемент 9 — Код программы 5



Элемент 10 — Результат программы 5

Заключение

В ходе выполнения лабораторной работы №5 были изучены принципы работы со **строками и регулярными выражениями** в Java.

Реализованные программы продемонстрировали:

- использование классов Pattern и Matcher для анализа текстов;
- применение различных шаблонов для поиска, проверки и замены данных;
- обработку исключений при работе с пользовательским вводом.

Данная работа позволила закрепить навыки текстовой обработки и повысить понимание механизмов регулярных выражений, применяемых при анализе данных, валидации и поиске информации в реальных приложениях.

Github: https://github.com/Prototype721/Java