**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

# Отчет по лабораторной работе № 5

Выполнил: студент группы БВТ2402

Косякова Олеся Дмитриевна

Москва, 2025

Цель работы: Освоение практического применения регулярных выражений в Java для обработки текстовых данных и валидации ввода.

Задание:

1) Поиск всех чисел в тексте
2) Проверка корректности ввода пароля
3) Поиск заглавной буквы после строчной
4) Проверка корректности ввода IP-адреса
5) Поиск всех слов, начинающихся с заданной буквы

Ход работы:

Задание 1.

Вызываем метод класса Pattern и компилируем представление нашего регулярного выражения. Создаем объект класса Matcher, которому передаем текст, к которому будем применять шаблон и сам шаблон. Сам объект хранит в себе текущее состояние поиска. Далее пишем цикл, который ищет совпадение с регулярным выражением и выводит это на экран. Далее возвращаем текс, который был успешно найден

```
public class NumberFinder {
    Run | Debug
    public static void main(String[] args) {
        String exm = "1234 dkfkd 111.233 djfdko 78,78 87&43 22?d23";
        try {
            Pattern pt = Pattern.compile(regex: "-?\\d+([.,]\\d+)?");
            Matcher mt = pt.matcher(exm);
            boolean ans = false;
            while (mt.find()) {
                System.out.println(mt.group());
                ans = true;
            }
            if (!ans) {
                System.out.println(x: "чисел нет");
            }
        } catch (PatternSyntaxException e) {
            System.err.println("ошибка в регулярном выражении: " + e.getMessage());
        }
    }

}
```

Задание 2.

Прописываем паттерн ?= обозначает, что нужно посмотреть вперед и определить есть ли любые символы в количестве 0 и больше

принадлежащие [A-Z], то же самое прописываем для цифр и указываем используемый алфавит и длину нашего пароля. Прописываем проверку на null и, если совпадения нет вызываем метод validate, который укажет в чем ошибка

```
public class Password {
    Run | Debug
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        String exm = console.nextLine();
        try {
            Pattern pt = Pattern.compile(regex: "(?=.*[A-Z])(?=.*\\d)[a-zA-Z\\d]{8,16}");
            Matcher mt = pt.matcher(exm);
            if (exm.isEmpty()) {
                throw new NullPointerException();
            }
            System.out.println(mt.matches());
             if (!mt.matches()) {
                validate(exm);
            }

        } catch (PatternSyntaxException e) {
            System.err.println("Ошибка в регулярном выражении: " + e.getMessage());
        } catch (NullPointerException e) {
            System.err.println(x: "Пароль не может быть null");
        } finally {
            console.close();
        }
    }
```

Задание 3.

Прописываем паттерн, разделяя его на две группы, далее в методе replaceAll обращаемся к первой группе символов и второй через $1, $2 и заменяем все совпадения на пары, между которыми восклицательные знаки.

```
public class UpperCase {
    Run | Debug
    public static void main(String[] args) {
        String exm = "helloWorld thisIsTest пример aBcDeFg приМер";
        Pattern pt = Pattern.compile(regex: "([a-zа-я])([A-ZА-Я])");
        Matcher mt = pt.matcher(exm);
        String ans = mt.replaceAll(replacement: "!$1$2!");
        System.out.println(ans);
    }
}
```

Задание 4.

В паттерне прописываем, что число может быть в диапазоне 250-255, или 200-240, или 100-199, или 10-99, или 0-9. И после каждого числа, кроме последнего должна быть точка.

```java
public class IP {
    Run | Debug
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        String exm = console.nextLine();
        try {
            Pattern pt = Pattern.compile(regex: "(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[0-9]\\.){3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[0-9])");
            Matcher mt = pt.matcher(exm);
            if (exm.isEmpty()) {
                throw new NullPointerException();
            }
            System.out.println(mt.matches());

        } catch (PatternSyntaxException e) {
            System.err.println("Ошибка в регулярном выражении: " + e.getMessage());
        } catch (NullPointerException e) {
            System.err.println(x: "IP не может быть null");
        } finally {
            console.close();
        }
    }
}
```

Задание 5.

В паттерне прописываем начало слова \\b и нашу букву далее указываем, что после нее может быть любое количество букв и пишем флаг игнорирования регистра.

```java
public class CharSearch {
    Run | Debug
    public static void main(String[] args) {
        String text = "Apple, orange, banana, Pineapple, berry, grape, Cherry.";
        Scanner console = new Scanner(System.in);
        String exm = console.nextLine();

        try {
            if (exm.isEmpty()) {
                System.out.println(x: "Буква не может быть null");
                return;
            }
            if (exm.length() > 1) {
                System.out.println(x: "Напишите одну букву");
                return;
            }

            Pattern pt = Pattern.compile("\\b" + exm + "\\w*\\b", Pattern.CASE_INSENSITIVE);
            Matcher mt = pt.matcher(text);

            boolean ans = false;
            while (mt.find()) {
                System.out.println(mt.group());
                ans = true;
            }

            if (!ans) {
                System.out.println(x: "Слов не нашлось");
            }

        } catch (PatternSyntaxException e) {
            System.err.println("ошибка в регулярном выражении: " + e.getMessage());
        } finally {
            console.close();
        }
    }
}
```

Вывод: В ходе выполнения лабораторной работы мы освоили практическое применение регулярных выражений в Java для обработки текстовых данных и валидации ввода.

Ссылка на репозиторий: https://github.com/swydiz/information_technology