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 информационным технологиям и программированию

на тему: **«Строки и регулярные выражения»**

Выполнил: студент группы БПИ2403

ФИО: Сон Владимир Сергеевич

Руководитель: Рыбаков Егор Дмитриевич

Москва, 2025

Цель работы: Изучить особенности работы со строками в Java, в частности свойства неизменяемости класса String, научиться использовать классы StringBuilder и StringBuffer для эффективной работы с изменяемыми последовательностями символов, а также познакомиться с основами применения регулярных выражений для обработки текста.

Задание 1. Поиск всех чисел в тексте

Написать программу, которая будет искать все числа в заданном тексте и выводить их на экран. При этом программа должна использовать регулярные выражения для поиска чисел и обрабатывать возможные ошибки.

Задание 2. Проверка корректности ввода пароля

Написать программу, которая будет проверять корректность ввода пароля. Пароль должен состоять из латинских букв и цифр, быть длиной от 8 до 16 символов и содержать хотя бы одну заглавную букву и одну цифру. При этом программа должна использовать регулярные выражения для проверки пароля и обрабатывать возможные ошибки.

Задание 3. Поиск заглавной буквы после строчной

Написать программу, которая будет находить все случаи в тексте, когда сразу после строчной буквы идет заглавная без какого-либо символа между ними и выделять их знаками «!» с двух сторон.

Задание 4. Проверка корректности ввода IP-адреса

Написать программу, которая будет проверять корректность ввода IP-адреса. IP-адрес должен состоять из 4 чисел, разделенных точками, и каждое число должно быть в диапазоне от 0 до 255. При этом программа должна использовать регулярные выражения для проверки IP-адреса и обрабатывать возможные ошибки.

Задание 5. Поиск всех слов, начинающихся с заданной буквы

Написать программу, которая будет искать все слова в заданном тексте, начинающиеся с заданной буквы, и выводить их на экран. При этом программа должна использовать регулярные выражения для поиска слов и обрабатывать возможные ошибки.

Ход работы:

```java
import java.util.regex.*;

public class NumberFinder {
    public static void main(String[] args) {
        String text = "БПС Challenger имел пробитие 230 мм" +
                      "Swererpanzer spahwagen 7.5 cm sondercraftfarzeug 234/4" +
                      "shdfklhsklfh s;ljdfpojsof sdjfoji;sdifj";

        Pattern pattern = Pattern.compile("\\d+\\.?\\d*");
        Matcher matcher = pattern.matcher(text);

        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

```java
import java.util.regex.*;
import java.util.Scanner;

public class PasswordValidator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String password = scanner.nextLine();

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

        if (matcher.matches()) {
            System.out.println("✓");
        } else {
            if (password.length() < 8 || password.length() > 16) {
                System.out.println("- Длина должна быть от 8 до 16 символов");
            }
            if (!password.matches(".*[A-Z].*")) {
                System.out.println("- Должна быть хотя бы одна заглавная буква");
            }
            if (!password.matches(".*\\d.*")) {
                System.out.println("- Должна быть хотя бы одна цифра");
            }
            if (!password.matches("[A-Za-z\\d]*")) {
                System.out.println("- Допустимы только латинские буквы и цифры");
            }
        }
        scanner.close();
    }
}
```

```java
import java.util.regex.*;

public class CapitalAfterLowerFinder {
    public static void main(String[] args) {
        String text = "Зачем мНе брИтанский Дырокол если еСть совЕтский КаморниК?";

        Pattern pattern = Pattern.compile("[а-я][А-Я]");
        Matcher matcher = pattern.matcher(text);
        StringBuffer result = new StringBuffer();

        while (matcher.find()) {
            String found = matcher.group();
            matcher.appendReplacement(result, "!" + found + "!");
        }
        matcher.appendTail(result);
        System.out.println(result.toString());
    }
}
```

```java
import java.util.regex.*;
import java.util.Scanner;

public class IPAddressValidator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String ipAddress = scanner.nextLine();

        if (isValidIP(ipAddress)) {
            System.out.println("✓");
        } else {
            System.out.println("✗");
            System.out.println("- Правильный формат: xxx.xxx.xxx.xxx");
            System.out.println("- где xxx - число от 0 до 255");
        }
        scanner.close();
    }

    public static boolean isValidIP(String ip) {
        String intPattern = "(25[0-5]|2[0-4]\\d|1\\d{2}|[1-9]?\\d)";
        String ipPattern = "^" + intPattern + "\\." + intPattern + "\\." +
                           intPattern + "\\." + intPattern + "$";

        Pattern pattern = Pattern.compile(ipPattern);
        Matcher matcher = pattern.matcher(ip);
        return matcher.matches();
    }
}
```

```java
import java.util.regex.*;
import java.util.ArrayList;
import java.util.List;

public class WordsStartingWithLetter {
    public static void main(String[] args) {
        String text = "Мам купи Abrams!" +
                      "У нас есть abrams дома" +
                      "ababs дома: 🚜 ";
        String letter = "A";

        List<String> words = findWordsStartingWith(text, letter);
        System.out.println(words);
    }

    public static List<String> findWordsStartingWith(String text, String letter) {
        List<String> words = new ArrayList<>();

        String patternStr = "\\b" + letter + "\\w*";
        Pattern pattern = Pattern.compile(patternStr, Pattern.CASE_INSENSITIVE);
        Matcher matcher = pattern.matcher(text);

        while (matcher.find()) {
            words.add(matcher.group());
        }
        return words;
    }
}
```

Вывод:

В ходе работы были рассмотрены ключевые аспекты работы со строками в Java. Было показано, что класс String является неизменяемым, что обеспечивает безопасность в многопоточных средах, но может приводить к избыточному созданию объектов. Для частых модификаций строк рекомендуется использовать StringBuilder (для однопоточных приложений) или StringBuffer (для многопоточных). Также затронуты основы интернирования строк и начала работы с регулярными выражениями для гибкого поиска и обработки текстовых данных.