**МАССОВЫХ КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Информационные технологии»

Лабораторная работа по ИТП №5

Выполнил: Студент группы

БВТ2204

Сарыков Эрбол

Москва

2023

Задание 1: Поиск всех чисел в тексте

Необходимо 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.

Пример реализации метода:

import java.util.regex.\*;

public class NumberFinder {

public static void main(String[] args) {

String text = "The price of the product is $19.99";

Pattern pattern = Pattern.compile("\\d+\\ .\\d+");

Matcher matcher = pattern.matcher(text);

while (matcher.find()) {

System.out.println(matcher.group());

}

}

}

Задание 2: Проверка корректности ввода пароля

Необходимо 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: Замена всех ссылок на гиперссылки

Необходимо написать программу, которая будет заменять все ссылки в

заданном тексте на гиперссылки. При этом программа должна

использовать регулярные выражения для поиска ссылок и замены и

обрабатывать возможные ошибки.

Задание 4: Проверка корректности ввода IP-адреса

Необходимо 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

Необходимо 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.

Задание 1

import java.util.regex.\*;

public class NumberFinder {

    public static void main(String[] args) {

        String text1 = "The price of the product is $19.99";

        String text2 = "abc";

        String text3 = "";

        try {

            findAndPrintNumbers(text1);

            findAndPrintNumbers(text2);

            findAndPrintNumbers(text3);

        } catch (Exception e) {

            System.err.println("Произошла ошибка: " + e.getMessage());

        }

    }

    private static void findAndPrintNumbers(String text) {

        Pattern pattern = Pattern.compile("\\b\\d+(\\.\\d{1,2})?\\b");

        Matcher matcher = pattern.matcher(text);

        System.out.println("Предложение: " + text);

        if (!matcher.find()) {

            System.out.println("В данном предложении нет чисел.\n");

            return;

        }

        System.out.print("Найденные числа: ");

        do {

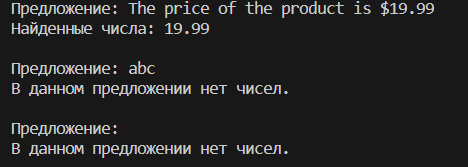
            System.out.print(matcher.group() + " ");

        } while (matcher.find());

        System.out.println("\n");

    }

}



1. Метод findAndPrintNumbers(String text)

Этот метод принимает текстовую строку text, в которой будет выполняться поиск чисел.

Метод выполняет следующие действия:

Создает объект Pattern, который представляет собой регулярное выражение для поиска чисел. В данном случае, регулярное выражение \\b\\d+(\\.\\d{1,2})?\\b используется для поиска чисел с возможной десятичной точкой и до двух знаков после запятой.

Создает объект Matcher, который будет использоваться для поиска соответствий регулярному выражению в тексте.

Выводит исходную строку текста на экран.

Проверяет, содержит ли текст числа, используя метод find() объекта Matcher. Если числа не найдены, выводит сообщение о том, что в данной строке нет чисел.

Если числа найдены, метод выполняет цикл do-while, который выводит все найденные числа, используя метод group() объекта Matcher.

2. Метод main

Метод main представляет точку входа в программу.

В этом методе определены три текстовые строки: text1, text2 и text3, каждая из которых содержит разные текстовые данные.

Затем метод вызывает findAndPrintNumbers для каждой из этих строк, а также обрабатывает исключение Exception, которое может возникнуть при выполнении метода findAndPrintNumbers. В случае ошибки, программа выводит сообщение об ошибке.

Задание 2

import java.util.regex.\*;

public class PasswordValidator {

    public static void main(String[] args) {

        String password = "Pod13242323";

        System.out.println(isValidPassword(password));

    }

    public static boolean isValidPassword(String password) {

        if (password.isEmpty()) {

            System.out.println("Вы не ввели пароль.");

            return false;

        }

        if (password.length() < 8 || password.length() > 16) {

            System.out.println("Пароль должен быть от 8 до 16 символов.");

            return false;

        }

        String regex = "^(?=.\*[A-Z])(?=.\*\\d)[A-Za-z\\d]{8,16}$";

        Pattern pattern = Pattern.compile(regex);

        Matcher matcher = pattern.matcher(password);

        if (!matcher.matches()) {

            System.out.println("Пароль должен содержать хотя бы одну заглавную букву и одну цифру.");

            return false;

        }

        return true;

    }

}



Основные компоненты

1. Метод isValidPassword(String password)

Этот метод принимает строку password в качестве аргумента и выполняет проверку на соответствие пароля заданным критериям безопасности.

Метод выполняет следующие проверки:

Проверяет, не является ли строка password пустой. Если строка пуста, метод выводит сообщение "Вы не ввели пароль." и возвращает false.

Проверяет, что длина строки password находится в диапазоне от 8 до 16 символов. Если длина не соответствует этому диапазону, метод выводит сообщение "Пароль должен быть от 8 до 16 символов." и возвращает false.

Использует регулярное выражение ^(?=.\*[A-Z])(?=.\*\\d)[A-Za-z\\d]{8,16}$ для проверки соответствия пароля заданным критериям. Это регулярное выражение требует наличие хотя бы одной заглавной буквы и хотя бы одной цифры в пароле. Если пароль не соответствует критериям, метод выводит сообщение "Пароль должен содержать хотя бы одну заглавную букву и одну цифру." и возвращает false.

Если все проверки пройдены успешно, метод возвращает true, указывая на то, что пароль соответствует заданным критериям безопасности.

2. Метод main

Метод main представляет точку входа в программу.

В этом методе определена строка password, которая представляет собой пароль для проверки.

Затем метод вызывает isValidPassword(password) для проверки пароля и выводит результат проверки на экран.

Задание 3

import java.util.regex.\*;

public class HyperlinkReplacer {

    public static void main(String[] args) {

        String text = "https://www.youtube.com/";

        String replacedText = replaceLinksWithHyperlinks(text);

        System.out.println(replacedText);

    }

    public static String replaceLinksWithHyperlinks(String text) {

        if (text.isEmpty()) {

            return "Строка пуста.";

        }

        String regex = "\\bhttps?://[-A-Za-z0-9+&@#/%?=~\_|!:,.;]\*[-A-Za-z0-9+&@#/%=~\_|]";

        try {

            Pattern pattern = Pattern.compile(regex);

            Matcher matcher = pattern.matcher(text);

            if (!matcher.find()) {

                return "Строка не содержит ссылок.";

            }

            matcher.reset();

            StringBuffer result = new StringBuffer();

            while (matcher.find()) {

                String replacement = "<a href=\"" + matcher.group() + "\">" + matcher.group() + "</a>";

                matcher.appendReplacement(result, replacement);

            }

            matcher.appendTail(result);

            return result.toString();

        } catch (PatternSyntaxException e) {

            System.err.println("Ошибка в синтаксисе регулярного выражения: " + e.getMessage());

            return null;

        }

    }

}



1. Метод replaceLinksWithHyperlinks(String text)

Этот метод принимает текстовую строку text, в которой будет выполняться поиск и замена ссылок.

Метод выполняет следующие действия:

Проверяет, не является ли строка text пустой. Если строка пуста, метод возвращает строку "Строка пуста."

Определяет регулярное выражение regex для поиска текстовых ссылок в тексте. Это регулярное выражение ищет строки, начинающиеся с "http://" или "https://", и содержащие различные символы, соответствующие URL.

Создает объект Pattern и Matcher для выполнения поиска ссылок в тексте.

Если в тексте не найдены ссылки, метод возвращает строку "Строка не содержит ссылок."

Если найдены ссылки, метод создает объект StringBuffer с именем result, который будет использоваться для создания результирующей строки.

В цикле while, метод выполняет замену каждой найденной ссылки на HTML-гиперссылку с использованием методов matcher.appendReplacement() и matcher.appendTail().

Затем метод возвращает строку result.toString(), которая содержит исходный текст с замененными ссылками на гиперссылки.

В случае ошибки синтаксиса регулярного выражения (класс PatternSyntaxException), метод выводит сообщение об ошибке и возвращает null.

2. Метод main

Метод main представляет точку входа в программу.

В этом методе определена строка text, которая содержит текст, в котором нужно выполнить поиск и замену ссылок.

Затем метод вызывает replaceLinksWithHyperlinks(text) для выполнения замены ссылок и выводит результат на экран.

Задание 4

import java.util.regex.\*;

public class IPAddressValidator {

    public static void main(String[] args) {

        String ipAddress = "192.168.1.1";

        System.out.println(isValidIPAddress(ipAddress));

    }

    public static boolean isValidIPAddress(String ipAddress) {

        if (ipAddress.isEmpty()) {

            System.out.println("Строка пуста.");

            return false;

        }

        String regex = "^(25[0-5]|2[0-4]\\d|[0-1]?\\d?\\d)(\\.(25[0-5]|2[0-4]\\d|[0-1]?\\d?\\d)){3}$";

        try {

            Pattern pattern = Pattern.compile(regex);

            Matcher matcher = pattern.matcher(ipAddress);

            if (!matcher.matches()) {

                System.out.println("Неверный формат IP-адреса.");

                return false;

            }

            return true;

        } catch (PatternSyntaxException e) {

            System.err.println("Ошибка в синтаксисе регулярного выражения: " + e.getMessage());

            return false;

        }

    }

}



1. Метод isValidIPAddress(String ipAddress)

Этот метод принимает строку ipAddress, которая представляет собой IP-адрес, который нужно проверить.

Метод выполняет следующие действия:

Проверяет, не является ли строка ipAddress пустой. Если строка пуста, метод выводит сообщение "Строка пуста." и возвращает false.

Определяет регулярное выражение regex, которое соответствует правильному формату IP-адреса. Регулярное выражение позволяет IP-адресу быть в формате "X.X.X.X", где X - это число от 0 до 255.

Создает объект Pattern и Matcher для выполнения проверки строки ipAddress на соответствие регулярному выражению.

Если строка ipAddress не соответствует регулярному выражению, метод выводит сообщение "Неверный формат IP-адреса." и возвращает false.

В случае ошибки синтаксиса регулярного выражения (класс PatternSyntaxException), метод выводит сообщение об ошибке и возвращает false.

Если строка ipAddress соответствует формату IP-адреса, метод возвращает true, указывая на то, что IP-адрес действителен.

2. Метод main

Метод main представляет точку входа в программу.

В этом методе определена строка ipAddress, которая содержит IP-адрес для проверки.

Затем метод вызывает isValidIPAddress(ipAddress) для проверки IP-адреса и выводит результат на экран.

Задание 5

import java.util.regex.\*;

public class WordFinder {

    public static void main(String[] args) {

        String text = "Apple apricot banana cherry date.";

        char startingLetter = 'b';

        findWords(text, startingLetter);

    }

    public static void findWords(String text, char startingLetter) {

        if (text.isEmpty()) {

            System.out.println("Строка пуста.");

            return;

        }

        String regex = "\\b" + Pattern.quote(String.valueOf(startingLetter)) + "\\w\*\\b";

        try {

            Pattern pattern = Pattern.compile(regex, Pattern.CASE\_INSENSITIVE);

            Matcher matcher = pattern.matcher(text);

            boolean found = false;

            while (matcher.find()) {

                System.out.println(matcher.group());

                found = true;

            }

            if (!found) {

                System.out.println("Слова, начинающиеся с буквы '" + startingLetter + "', не найдены.");

            }

        } catch (PatternSyntaxException e) {

            System.err.println("Ошибка в синтаксисе регулярного выражения: " + e.getMessage());

        }

    }

}



1. Метод findWords(String text, char startingLetter)

Этот метод принимает текстовую строку text и символ startingLetter, который представляет собой букву, с которой должны начинаться найденные слова.

Метод выполняет следующие действия:

Проверяет, не является ли строка text пустой. Если строка пуста, метод выводит сообщение "Строка пуста." и завершает выполнение.

Определяет регулярное выражение regex для поиска слов, начинающихся с заданной буквы startingLetter. Регулярное выражение использует \b для обозначения начала слова, Pattern.quote(String.valueOf(startingLetter)) для безопасной обработки символа startingLetter, \w\* для поиска слова и \b для обозначения конца слова. Также используется флаг Pattern.CASE\_INSENSITIVE для выполнения поиска без учета регистра букв.

Создает объекты Pattern и Matcher для выполнения поиска слов в тексте.

В цикле while, метод выполняет поиск и выводит каждое найденное слово с заданной буквы. Переменная found используется для отслеживания наличия найденных слов.

Если в тексте не найдено ни одного слова, начинающегося с заданной буквы, метод выводит сообщение "Слова, начинающиеся с буквы 'startingLetter', не найдены."

В случае ошибки синтаксиса регулярного выражения (класс PatternSyntaxException), метод выводит сообщение об ошибке.

2. Метод main

Метод main представляет точку входа в программу.

В этом методе определена строка text, которая содержит текст, в котором нужно выполнить поиск слов.

Также определен символ startingLetter, который представляет собой букву, с которой должны начинаться найденные слова.

Затем метод вызывает findWords(text, startingLetter) для выполнения поиска и вывода найденных слов.