Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа, обработки и интерпретации больших данных**

О Т Ч Е Т

по лабораторной работе №7

Название: Строки и регулярные выражения

Дисциплина: Языки программирования для работы с большими данными

Студент	ИУ6-22М		М.А. Зотов
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2023 г.

Цель: получить опыт работы со строками и регулярными выражениями.

Вариант 1 – задание 9

Условие: из текста удалить все слова заданной длины, начинающиеся на согласную букву.

Код:

```
import java.util.Arrays;
import java.util.HashSet;
import java.util.Scanner;

public class var1_ex9 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String text = "Этот текст содержит слова разной длины и
на разных буквах. Некоторые слова начинаются на согласные, а
некоторые на гласные.";

        System.out.print("Введите длину слова: ");
        int length = in.nextInt(); // Заданная длина слова
        HashSet<Character> consonants = new
HashSet<>(Arrays.asList('б', 'в', 'г', 'д', 'ж', 'з', 'й', 'к',
'л', 'м', 'н', 'п', 'р', 'с', 'т', 'ф', 'х', 'ц', 'ч', 'ш',
'щ')); // Согласные буквы

        String[] words = text.split("\\s+"); // Разбить текст на
слова
        StringBuilder result = new StringBuilder(); //
Результирующий текст

        for (String word : words) {
            if (word.length() == length &&
consonants.contains(Character.toLowerCase(word.charAt(0)))) {
                continue; // Пропустить слово
            }
            result.append(word).append(" "); // Добавить слово в
результирующий текст
        }

        System.out.println(result.toString().trim()); // Вывести
результат
    }
}
```

Результат выполнения:

```
Введите длину слова: 5
Этот содержит разной и на разных буквах. Некоторые начинаются на согласные, а некоторые на гласные.
```

Рисунок 1 – Результат выполнения 1.9

Вариант 1 – задание 10

Условие: удалить из текста его часть, заключенную между двумя символами, которые вводятся (например, между скобками ‘(’ и ‘)’ или между звездочками ‘*’ и т.п.).

Код:

```
import java.util.Scanner;

public class var1_ex10 {
    public static void main(String[] args) {
        String text = "Этот текст (содержит) некоторые (части), 
которые нужно удалить.";
        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите первый символ: ");
        char startChar = scanner.next().charAt(0);

        System.out.print("Введите второй символ: ");
        char endChar = scanner.next().charAt(0);

        StringBuilder result = new StringBuilder(); // 
Результирующий текст
        boolean isInside = false; // Флаг, указывающий, 
находимся ли мы внутри блока, который нужно удалить

        for (int i = 0; i < text.length(); i++) {
            char c = text.charAt(i);

            if (c == startChar) {
                isInside = true; // Входим в блок, который нужно 
удалить
                continue;
            }

            if (c == endChar) {
                isInside = false; // Выходим из блока, который 
нужно удалить
                continue;
            }

            if (!isInside) {
                result.append(c); // Добавляем символ в 
результирующий текст, если мы не находимся внутри блока
            }
        }

        System.out.println(result.toString()); // Выводим 
результат
    }
}
```

Результат выполнения:

```
Введите первый символ: (
Введите второй символ: )
Этот текст  некоторые , которые нужно удалить.
```

Рисунок 2 – Результат выполнения 1.10

Вариант 2 – задание 9

Условие: В стихотворении найти одинаковые буквы, которые встречаются во всех словах.

Код:

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class var2_ex9 {
    public static void main(String[] args) {
        String poem = "Морозъ иъ солнцеъ; деньъ чудесныйъ!\nЕщеъ тыъ дремлешьъ, другъ прелестныйъ –ъ\nПораъ, красавицаъ, проснисьъ:\nОткройъ сомкнутыъ негойъ взорыъ\nНавстречуъ северноймъ Аврорыъ,\nЗвездоюъ севераъ явисьъ!";

        Set<Character> commonChars = new HashSet<>(); // Множество общих букв
        boolean isFirstWord = true; // Флаг, указывающий, что это первое слово

        Scanner scanner = new Scanner(poem);
        while (scanner.hasNext()) {
            String word = scanner.next();
            Set<Character> currentChars = new HashSet<>(); // Множество букв в текущем слове

            for (int i = 0; i < word.length(); i++) {
                char c = word.charAt(i);
                if (isFirstWord) {
                    commonChars.add(c); // Если это первое слово, добавляем все его буквы в множество общих букв
                } else {
                    if (commonChars.contains(c)) {
                        currentChars.add(c); // Если общие буквы содержат текущую букву, добавляем ее в множество текущих букв
                    }
                }
            }

            if (!isFirstWord) {
                commonChars.retainAll(currentChars); // Удаляем из множества общих букв те, которые не встречаются в текущем слове
            }
```

```
            isFirstWord = false; // Снимаем флаг первого слова
        }

        System.out.println("Общие буквы: " + commonChars); //
Выводим результат
    }
}
```

Результат выполнения:

```
Общие буквы: [ъ]
```

Рисунок 3 – Результат выполнения 2.9

Вариант 2 – задание 10

Условие: в тексте найти первую подстроку максимальной длины, не содержащую букв.

Код:

```
public class var2_ex10 {
    public static void main(String[] args) {
        String text = "Текст для поиска первой подстроки
максимальной длины, не содержащей буквы.";

        String maxSubstring = "";
        String currentSubstring = "";

        for (int i = 0; i < text.length(); i++) {
            char c = text.charAt(i);

            if (Character.isLetter(c)) {
                // Нашли букву, обновляем текущую подстроку
                currentSubstring = "";
            } else {
                // Нашли не букву, добавляем к текущей подстроке
                currentSubstring += c;

                if (currentSubstring.length() >
maxSubstring.length()) {
                    // Обновляем максимальную подстроку, если
текущая длиннее
                    maxSubstring = currentSubstring;
                }
            }
        }

        System.out.println("Первая подстрока максимальной длины,
не содержащая буквы: '" + maxSubstring + "'");
    }
}
```

Результат выполнения:

```
Первая подстрока максимальной длины, не содержащая буквы: ', '
```

Рисунок 4 – Результат выполнения 2.10

Вариант 3 – задание 9

Условие: напечатать слова русского текста в алфавитном порядке по первой букве. Слова, начинающиеся с новой буквы, печатать с красной строки.

Код:

```
import java.util.Arrays;
import java.util.Comparator;
import java.util.List;

public class var3_ex9 {
    public static void main(String[] args) {
        String text = "Напечатать слова русского текста в
алфавитном порядке по первой букве. Слова, начинающиеся с новой
буквы, печатать с красной строки.";

        // Разбиваем текст на слова и сортируем их по первой
букве в алфавитном порядке
        List<String> words = Arrays.asList(text.split("\\s+"));
        words.sort(Comparator.comparingInt(s -> s.charAt(0)));

        // Печатаем слова, начинающиеся с новой буквы, с красной
строки
        char currentLetter = 0;
        for (String word : words) {
            char firstLetter = word.charAt(0);
            if (firstLetter != currentLetter) {
                System.out.println();
                System.out.print(word + " ");
                currentLetter = firstLetter;
            } else {
                System.out.print(word + " ");
            }
        }
    }
}
```

Результат выполнения:

```
Напечатать
Слова,
алфавитном
букве. буквы,
в
красной
начинающиеся новой
порядке по первой печатать
русского
слова с с строки.
текста
```

Рисунок 5 – Результат выполнения 3.9

Вариант 3 – задание 10

Условие: рассортировать слова русского текста по возрастанию доли гласных букв (отношение количества гласных к общему количеству букв в слове).

Код:

```
import java.util.Arrays;
import java.util.Collections;
import java.util.Comparator;
import java.util.List;

public class var3_ex10 {
    public static void main(String[] args) {
        String text = "Рассортировать слова русского текста по
возрастанию доли гласных букв (отношение количества гласных к
общему количеству букв в слове)";

        // Разбиваем текст на слова
        List<String> words = Arrays.asList(text.split("\\s+"));

        // Сортируем слова по возрастанию доли гласных букв
        words.sort(new Comparator<>() {
            @Override
            public int compare(String s1, String s2) {
                double vowelsFraction1 =
countVowelsFraction(s1);
                double vowelsFraction2 =
countVowelsFraction(s2);
                return Double.compare(vowelsFraction1,
vowelsFraction2);
            }

            private double countVowelsFraction(String word) {
                int vowelCount = 0;
                int totalLetterCount = 0;
                for (char c : word.toCharArray()) {
                    if (isVowel(c)) {
                        vowelCount++;
                    }
                    totalLetterCount++;
                }
                return (double) vowelCount / totalLetterCount;
            }

            private boolean isVowel(char c) {
                return
"аеёиоуыэюя".indexOf(Character.toLowerCase(c)) != -1;
            }
        });
```

```
        // Печатаем отсортированные слова
        for (String word : words) {
            System.out.println(word);
        }
    }
}
```

Результат выполнения:

```
к
в
букв
букв
гласных
гласных
текста
слове)
Рассортировать
русского
слова
количества
количеству
возрастанию
по
доли
(отношение
общему
```

Рисунок 6 – Результат выполнения 3.10

Вариант 4 – задание 9

Условие: Преобразовать каждое слово в тексте, удалив из него все последующие (предыдущие) вхождения первой (последней) буквы этого слова.

Код:

```
public class var4_ex9 {
    public static void main(String[] args) {
        String text = "Преобразовать каждое слово в тексте,
удалив из него все последующие предыдущие вхождения первой
последней буквы этого слова";
        String[] words = text.split("\\s+");
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            char firstChar = word.charAt(0);
            char lastChar = word.charAt(word.length() - 1);
            String transformedWord;
            if (word.length() > 2)
                transformedWord = firstChar
                        + word.substring(1, word.length() - 1)

.replaceAll(String.valueOf(firstChar), "")

.replaceAll(String.valueOf(lastChar), "")
```

```
                        + lastChar;
                else
                    transformedWord = word;
                result.append(transformedWord).append(" ");
            }

            System.out.println(result.toString().trim());
        }
}
```

Результат выполнения:

```
Преобразовать каждое слво в тексе, удалив из него все послдующие прдыдущие вхождения первой последней буквы этго слова
```

Рисунок 7 – Результат выполнения 4.9

Вариант 4 – задание 10

Условие: исключить из текста подстроку максимальной длины, начинающуюся и заканчивающуюся одним и тем же символом.

Код:

```
public class var4_ex10 {
    public static void main(String[] args) {
        String text = "Это некоторый текст, содержащий подстроку
максимальной длины, начинающуюся и заканчивающуюся символом
'о'.";

        String maxSubstring = "";
        int maxLength = 0;

        for (int i = 0; i < text.length(); i++) {
            for (int j = i + 1; j < text.length(); j++) {
                if (text.charAt(i) == text.charAt(j)) {
                    String substring = text.substring(i, j + 1);
                    if (substring.length() > maxLength) {
                        maxSubstring = substring;
                        maxLength = substring.length();
                    }
                }
            }
        }

        String result = text.replace(maxSubstring, "");
        System.out.println(result);
    }
}
```

Результат выполнения:

```
Эт'.
```

Рисунок 8 – Результат выполнения 4.10

Вывод: в ходе выполнения лабораторной работы был получен опыт работы со строками и регулярными выражениями.