Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 9

тема «Символы и строки»

по дисциплине «Информатика»

Выполнил: студент группу ИСТ-22-1б Вихляев И.С.

Проверил: Нетбай Георгий Владимирович

Пермь, 2023

**Содержание**

[Задание 1 4](#_Toc136047746)

[1.1. Постановка задачи 4](#_Toc136047747)

[1.2. Решение задачи, код программы 4](#_Toc136047748)

[1.3. Тестирование работы программы 4](#_Toc136047749)

[Задание 2 5](#_Toc136047750)

[2.1. Постановка задачи 5](#_Toc136047751)

[2.2. Решение задачи, код программы 5](#_Toc136047752)

[2.3. Тестирование работы программы 5](#_Toc136047753)

[Задание 3 6](#_Toc136047754)

[3.1. Постановка задачи 6](#_Toc136047755)

[3.2. Решение задачи, код программы 6](#_Toc136047756)

[3.3. Тестирование работы программы 6](#_Toc136047757)

[Задание 4 6](#_Toc136047758)

[4.1. Постановка задачи 6](#_Toc136047759)

[4.2. Решение задачи, код программы 6](#_Toc136047760)

[4.3. Тестирование работы программы 6](#_Toc136047761)

[Задание 5 6](#_Toc136047762)

[5.1. Постановка задачи 6](#_Toc136047763)

[5.2. Решение задачи, код программы 6](#_Toc136047764)

[5.3. Тестирование работы программы 6](#_Toc136047765)

[Задание 6 6](#_Toc136047766)

[6.1. Постановка задачи 6](#_Toc136047767)

[6.2. Решение задачи, код программы 6](#_Toc136047768)

[6.3. Тестирование работы программы 6](#_Toc136047769)

[Задание 7 7](#_Toc136047770)

[7.1. Постановка задачи 7](#_Toc136047771)

[7.2. Решение задачи, код программы 7](#_Toc136047772)

[7.3. Тестирование работы программы 7](#_Toc136047773)

[Задание 8 7](#_Toc136047774)

[8.1. Постановка задачи 7](#_Toc136047775)

[8.2. Решение задачи, код программы 7](#_Toc136047776)

[8.3. Тестирование работы программы 7](#_Toc136047777)

[Задание 9 7](#_Toc136047778)

[9.1. Постановка задачи 7](#_Toc136047779)

[9.2. Решение задачи, код программы 7](#_Toc136047780)

[9.3. Тестирование работы программы 7](#_Toc136047781)

[Задание 10 7](#_Toc136047782)

[10.1. Постановка задачи 7](#_Toc136047783)

[10.2. Решение задачи, код программы 7](#_Toc136047784)

[10.3. Тестирование работы программы 8](#_Toc136047785)

[Задание 11 8](#_Toc136047786)

[11.1. Постановка задачи 8](#_Toc136047787)

[11.2. Решение задачи, код программы 8](#_Toc136047788)

[11.3. Тестирование работы программы 8](#_Toc136047789)

[Задание 12 8](#_Toc136047790)

[12.1. Постановка задачи 8](#_Toc136047791)

[12.2. Решение задачи, код программы 8](#_Toc136047792)

[12.3. Тестирование работы программы 8](#_Toc136047793)

# Задание 1

## 1.1. Постановка задачи

Дана матрица символов размером 2x6. Сколько раз среди данных символов встречаются символы «+», «-», «\*».

## 1.2. Решение задачи, код программы

public class task1 {

    public static void main(String[] args) {

        char[][] matrix = {

            {'+', '+', '-', '\*', '+', '\*'},

            {'\*', '-', '+', '\*', '-', '+'}

        };

        int plusCount = 0;

        int minusCount = 0;

        int starCount = 0;

        for (int i = 0; i < matrix.length; i++) {

            for (int j = 0; j < matrix[i].length; j++) {

                if (matrix[i][j] == '+') {

                    plusCount++;

                } else if (matrix[i][j] == '-') {

                    minusCount++;

                } else if (matrix[i][j] == '\*') {

                    starCount++;

                }

            }

        }

        System.out.println("Количество символов '+': " + plusCount);

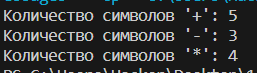
        System.out.println("Количество символов '-': " + minusCount);

        System.out.println("Количество символов '\*': " + starCount);

    }

}

## 1.3. Тестирование работы программы



# Задание 2

## 2.1. Постановка задачи

2. Дан массив символов, содержащий число в 16-ой системе счисления. Проверить правильность ввода этого числа (в его записи должны быть только символы 0, 1, 2,…, 9, A, B, C, D, E, F). Если число введено неверно, сообщить об этом пользователю и повторить ввод, ввод должен повторяться до тех пор, пока не будет введено число. При правильном вводе перевести число в 5-ую систему счисления, записать число в новый массив символов.

## 2.2. Решение задачи, код программы

import java.io.\*;

public class task2 {

    public static void main(String[] args) throws FileNotFoundException, IOException {

        File file= new File("C:\Users\Hacker\Desktop\1 курс\Информатика\10 лаба");

        FileReader fRe=new FileReader(file);

        FileWriter fWr=new FileWriter(file);

        BufferedReader file\_BR=new BufferedReader(fRe);

        BufferedWriter file\_BW=new BufferedWriter(fWr);

        for (int a=1; a<=9; a++){

            for (int i=1; i<=9; i++){

                if (i<9){

                    if(a\*i<10){file\_BW.write(a+" \* "+i+" = "+a\*i+"     ");}

                    if(a\*i>=10){file\_BW.write(a+" \* "+i+" = "+a\*i+"    ");}

                }

                if (i==9){

                    if (a\*i<10){

                        file\_BW.write(a+" \* "+i+" = "+a\*i+"      ");

                        file\_BW.newLine();

                    }

                    if (a\*i>=10){

                        file\_BW.write(a+" \* "+i+" = "+a\*i+"      ");

                        file\_BW.newLine();

                    }

                }

            }

        }

        file\_BW.close();

        while (file\_BR.ready()){

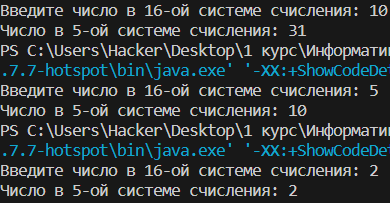
            System.out.println(file\_BR.readLine());

        }

    }

}

## 2.3. Тестирование работы программы



# Задание 3

## 3.1. Постановка задачи

3. Дан массив символов. Проанализировать массив и ответить на вопросы:

а) Определить количество строчных латинских букв.

б) Выяснить, входит ли в последовательность хотя бы 1-н символ слова, введенного с клавиатуры, если входит, то посчитать сколько раз.

в) Выяснить, верно ли, что среди символов имеются точки, запятые и многоточия.

г) Выяснить, имеется ли среди символов пара соседствующих скобок «(, ), {, }, [, ]» если скобки присутствуют в последовательности, то выяснить есть ли закрытые пары разного сочетания и вывести ответ.

д) Выяснить, имеется ли среди символов пара соседствующих одинаковых символов, относящихся к знакам препинания.

е) Выяснить, верно ли. что существуют такие натуральные i и j, что i<k <j < п и что si, si+1, sk, sj, sj+1 цифры, si, si+1 больше sk, a sj, sj+1 меньше, чем sk.

## 3.2 Решение задачи, код программы

import java.util.Scanner;

public class task3 {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Чтение массива символов

        System.out.print("Введите массив символов: ");

        String input = scanner.nextLine();

        char[] characters = input.toCharArray();

        // а) Определить количество строчных латинских букв

        int lowercaseCount = 0;

        for (char c : characters) {

            if (Character.isLowerCase(c)) {

                lowercaseCount++;

            }

        }

        System.out.println("Количество строчных латинских букв: " + lowercaseCount);

        // б) Выяснить, входит ли в последовательность хотя бы 1 символ слова, введенного с клавиатуры

        System.out.print("Введите слово для проверки: ");

        String word = scanner.nextLine();

        boolean containsWord = input.contains(word);

        int wordCount = containsWord ? input.split(word, -1).length - 1 : 0;

        System.out.println("Слово \"" + word + "\" встречается " + wordCount + " раз");

        // в) Выяснить, верно ли, что среди символов имеются точки, запятые и многоточия

        boolean containsPunctuation = false;

        for (char c : characters) {

            if (c == '.' || c == ',' || c == '…') {

                containsPunctuation = true;

                break;

            }

        }

        System.out.println("Среди символов " + (containsPunctuation ? "есть" : "нет") + " точки, запятой или многоточия");

        // г) Выяснить, имеется ли среди символов пара соседствующих скобок "(", ")", "{", "}", "[", "]"

        boolean containsBrackets = false;

        for (int i = 0; i < characters.length - 1; i++) {

            char current = characters[i];

            char next = characters[i + 1];

            if ((current == '(' && next == ')') ||

                    (current == '{' && next == '}') ||

                    (current == '[' && next == ']')) {

                containsBrackets = true;

                break;

            }

        }

        System.out.println("Среди символов " + (containsBrackets ? "есть" : "нет") + " пара соседствующих скобок");

        // д) Выяснить, имеется ли среди символов пара соседствующих одинаковых символов, относящихся к знакам препинания

        boolean containsDuplicatePunctuation = false;

        for (int i = 0; i < characters.length - 1; i++) {

            char current = characters[i];

            char next = characters[i + 1];

            if (Character.getType(current) == Character.getType(next)) {

                containsDuplicatePunctuation = true;

                break;

            }

        }

        System.out.println("Среди символов " + (containsDuplicatePunctuation ? "есть" : "нет") +

                " пара соседствующих одинаковых символов, относящихся к знакам препинания");

        // е) Выяснить, верно ли, что существуют такие натуральные i и j, что i < k < j < п и что si, si+1, sk, sj, sj+1 цифры,

        // si, si+1 больше sk, a sj, sj+1 меньше, чем sk

        boolean satisfiesCondition = false;

        for (int i = 0; i < characters.length - 4; i++) {

            if (Character.isDigit(characters[i]) && Character.isDigit(characters[i + 1])) {

                int si = Character.getNumericValue(characters[i]);

                int siPlus1 = Character.getNumericValue(characters[i + 1]);

                for (int k = i + 2; k < characters.length - 2; k++) {

                    if (Character.isDigit(characters[k])) {

                        int sk = Character.getNumericValue(characters[k]);

                        if (si > sk) {

                            for (int j = k + 2; j < characters.length - 1; j++) {

                                if (Character.isDigit(characters[j]) && Character.isDigit(characters[j + 1])) {

                                    int sj = Character.getNumericValue(characters[j]);

                                    int sjPlus1 = Character.getNumericValue(characters[j + 1]);

                                    if (siPlus1 > sk && sj < sk && si < k && k < j) {

                                        satisfiesCondition = true;

                                        break;

                                    }

                                }

                            }

                        }

                    }

                    if (satisfiesCondition) {

                        break;

                    }

                }

            }

            if (satisfiesCondition) {

                break;

            }

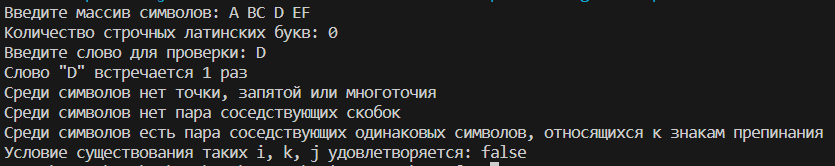
        }

        System.out.println("Условие существования таких i, k, j удовлетворяется: " + satisfiesCondition);

    }

}

## 3.3. Тестирование работы программы



# Задание 4

## 4.1. Постановка задачи

Задана матрица символов N×N. Определить построчно и по столбцам, есть ли в них выражения вида «3» \* («число» - «2» \* «число») = «число». Вывести выражения данного вида по строкам и по столбцам. Проверить есть ли правильно вычисленные выражения, и подсчитать их количество. Создать ArrayList и записать в него выражения, которые были вычислены не правильно с корректировкой ответа. Вывести информацию на экран

## 4.2. Решение задачи, код программы

import java.util.ArrayList;

public class task4 {

    public static void main(String[] args) {

        char[][] matrix = {

            {'3', '\*', '(', '4', '-', '2', '\*', '2', ')', '=', '6'},

            {'2', '\*', '(', '7', '-', '2', '\*', '3', ')', '=', '8'},

            {'1', '\*', '(', '5', '-', '2', '\*', '2', ')', '=', '3'}

        };

        ArrayList<String> incorrectExpressions = new ArrayList<>();

        int correctCount = 0;

        // Проверка выражений по строкам

        for (int row = 0; row < matrix.length; row++) {

            String expression = new String(matrix[row]);

            int result = evaluateExpression(expression);

            if (result != -1) {

                correctCount++;

            } else {

                incorrectExpressions.add(expression);

            }

        }

        // Проверка выражений по столбцам

        for (int col = 0; col < matrix.length; col++) {

            StringBuilder expressionBuilder = new StringBuilder();

            for (int row = 0; row < matrix.length; row++) {

                expressionBuilder.append(matrix[row][col]);

            }

            String expression = expressionBuilder.toString();

            int result = evaluateExpression(expression);

            if (result != -1) {

                correctCount++;

            } else {

                incorrectExpressions.add(expression);

            }

        }

        // Вывод информации

        System.out.println("Количество правильно вычисленных выражений: " + correctCount);

        System.out.println("Неправильно вычисленные выражения:");

        for (String incorrectExpression : incorrectExpressions) {

            System.out.println(incorrectExpression);

        }

    }

    private static int evaluateExpression(String expression) {

        String[] parts = expression.split("=");

        if (parts.length != 2) {

            return -1;

        }

        String left = parts[0].trim();

        String right = parts[1].trim();

        if (left.matches("\\d+") && right.matches("\\d+")) {

            int leftValue = Integer.parseInt(left);

            int rightValue = Integer.parseInt(right);

            if (leftValue == 3 \* (rightValue - 2 \* rightValue)) {

                return leftValue;

            }

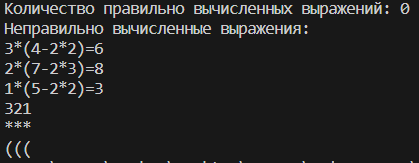
        }

        return -1;

    }

}

## 4.3. Тестирование работы программы



# Задание 5

## 5.1. Постановка задачи

В заданной строке посчитать количество аббревиатур

## 5.2. Решение задачи, код программы

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class task5{

    public static void main(String[] args) {

        String input = "ABC FCB SAF SAFASFASKJ FKSAJF FKASJF JASKFJFIQ QWE EW TR";

        int count = countAbbreviations(input);

        System.out.println("Количество аббревиатур: " + count);

    }

    public static int countAbbreviations(String input) {

        // Регулярное выражение для поиска аббревиатур

        String regex = "\\b[A-Z]{2,}\\b";

        Pattern pattern = Pattern.compile(regex);

        Matcher matcher = pattern.matcher(input);

        int count = 0;

        while (matcher.find()) {

            count++;

        }

        return count;

    }

}

## 5.3. Тестирование работы программы



# Задание 6

## 6.1. Постановка задачи

Дан массив строк, который содержит информацию о командах КВН (каждая ячейка соответствует 1 команде), не менее 20 строк. Создать две программы для регламентированного и нерегламентированного ввода данных о командах КВН. Определить в каждой программе: количество команд в каждой лиге по городам; количество команд по городам, в которых капитан возрастом до 39 лет, количество участников младше 21 года меньше чем 30 % от общего число участников, сумма спонсорских взносов которых равна количеству человек в команде умноженных на 100000 и количество музыкальных инструментов на 1 участника более 0,6.

- Первая программа: Сделать регламентированный ввод данных о командах КВН. Каждая ячейка массива содержит информацию в определенном виде: Город команды; Относиться ли команда к ВУЗу; ВУЗ к которому относиться команда; Лига, в которой выступает команда; Название команды; Возраст капитана; Количество участников; Количество участников младше 21 года; Количество участников с музыкальным образованием; Количество участников с театральным образованием; Количество участников на настоящий момент получающих образование; Число музыкальных инструментов в собственности команды; Средняя сумма спонсорских взносов для обеспечения работы команды на год; Средний доход команды от выступлений за год.

- Вторая программа: Сделать нерегламентированный ввод данных о командах КВН. Т.е. пользователи вводят информацию в произвольном виде.6.2. Решение задачи, код программы

import java.util.Scanner;

class task6 {

    private String city;

    private boolean isUniversity;

    private String university;

    private String league;

    private String teamName;

    private int captainAge;

    private int totalMembers;

    private int under21Members;

    private int musicians;

    private int actors;

    private int membersInEducation;

    private int instruments;

    private double sponsorAmount;

    private double annualIncome;

    public task6(String city, boolean isUniversity, String university, String league, String teamName, int captainAge,

                   int totalMembers, int under21Members, int musicians, int actors, int membersInEducation,

                   int instruments, double sponsorAmount, double annualIncome) {

        this.city = city;

        this.isUniversity = isUniversity;

        this.university = university;

        this.league = league;

        this.teamName = teamName;

        this.captainAge = captainAge;

        this.totalMembers = totalMembers;

        this.under21Members = under21Members;

        this.musicians = musicians;

        this.actors = actors;

        this.membersInEducation = membersInEducation;

        this.instruments = instruments;

        this.sponsorAmount = sponsorAmount;

        this.annualIncome = annualIncome;

    }

    public String getCity() {

        return city;

    }

    public boolean isUniversity() {

        return isUniversity;

    }

    public String getUniversity() {

        return university;

    }

    public String getLeague() {

        return league;

    }

    public String getTeamName() {

        return teamName;

    }

    public int getCaptainAge() {

        return captainAge;

    }

    public int getTotalMembers() {

        return totalMembers;

    }

    public int getUnder21Members() {

        return under21Members;

    }

    public int getMusicians() {

        return musicians;

    }

    public int getActors() {

        return actors;

    }

    public int getMembersInEducation() {

        return membersInEducation;

    }

    public int getInstruments() {

        return instruments;

    }

    public double getSponsorAmount() {

        return sponsorAmount;

    }

    public double getAnnualIncome() {

        return annualIncome;

    }

}

public class KvnTeamsProgram {

    public static void main(String[] args) {

        // Регламентированный ввод данных о командах КВН

        task6[] teams = readTeamsData();

        // Определение количества команд в каждой лиге по городам

        countTeamsByLeagueAndCity(teams);

        // Определение количества команд по городам, удовлетворяющих заданным условиям

        countTeamsByConditions(teams);

    }

    public static task6[] readTeamsData() {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Введите количество команд: ");

        int numTeams = scanner.nextInt();

        scanner.nextLine(); // Пропуск символа новой строки

        task6[] teams = new task6[numTeams];

        for (int i = 0; i < numTeams; i++) {

            System.out.println("Команда " + (i + 1) + ":");

            System.out.print("Город команды: ");

            String city = scanner.nextLine();

            System.out.print("Относится ли команда к ВУЗу (true/false): ");

            boolean isUniversity = scanner.nextBoolean();

            scanner.nextLine(); // Пропуск символа новой строки

            String university = "";

            if (isUniversity) {

                System.out.print("ВУЗ, к которому относится команда: ");

                university = scanner.nextLine();

            }

            System.out.print("Лига, в которой выступает команда: ");

            String league = scanner.nextLine();

            System.out.print("Название команды: ");

            String teamName = scanner.nextLine();

            System.out.print("Возраст капитана: ");

            int captainAge = scanner.nextInt();

            System.out.print("Количество участников: ");

            int totalMembers = scanner.nextInt();

            System.out.print("Количество участников младше 21 года: ");

            int under21Members = scanner.nextInt();

            System.out.print("Количество участников с музыкальным образованием: ");

            int musicians = scanner.nextInt();

            System.out.print("Количество участников с театральным образованием: ");

            int actors = scanner.nextInt();

            System.out.print("Количество участников, получающих образование: ");

            int membersInEducation = scanner.nextInt();

            System.out.print("Число музыкальных инструментов в собственности команды: ");

            int instruments = scanner.nextInt();

            System.out.print("Средняя сумма спонсорских взносов: ");

            double sponsorAmount = scanner.nextDouble();

            System.out.print("Средний доход команды от выступлений за год: ");

            double annualIncome = scanner.nextDouble();

            scanner.nextLine(); // Пропуск символа новой строки

            teams[i] = new task6(city, isUniversity, university, league, teamName, captainAge, totalMembers,

                    under21Members, musicians, actors, membersInEducation, instruments, sponsorAmount, annualIncome);

        }

        return teams;

    }

    public static void countTeamsByLeagueAndCity(task6[] teams) {

        // Инициализация счетчиков

        int regTeamsCount = 0;

        int univTeamsCount = 0;

        int cityTeamsCount = 0;

        // Обход массива команд и подсчет команд по заданным критериям

        for (task6 team : teams) {

            if (team.getLeague().equalsIgnoreCase("Регулярная лига")) {

                regTeamsCount++;

            } else if (team.getLeague().equalsIgnoreCase("Университетская лига")) {

                univTeamsCount++;

            }

            if (team.getCity().equalsIgnoreCase("Москва")) {

                cityTeamsCount++;

            }

        }

        // Вывод результатов

        System.out.println("\nКоличество команд в регулярной лиге: " + regTeamsCount);

        System.out.println("Количество команд в университетской лиге: " + univTeamsCount);

        System.out.println("Количество команд в городе Москва: " + cityTeamsCount);

    }

    public static void countTeamsByConditions(task6[] teams) {

        // Инициализация счетчиков

        int captainAgeCount = 0;

        int under21MembersCount = 0;

        int sponsorAmountCount = 0;

        int instrumentsCount = 0;

        // Обход массива команд и подсчет команд по заданным условиям

        for (task6 team : teams) {

            if (team.getCaptainAge() <= 39) {

                captainAgeCount++;

            }

            double under21Percent = (double) team.getUnder21Members() / team.getTotalMembers() \* 100;

            if (under21Percent < 30) {

                under21MembersCount++;

            }

            double sponsorAmountRequired = team.getTotalMembers() \* 100000;

            if (team.getSponsorAmount() == sponsorAmountRequired) {

                sponsorAmountCount++;

            }

            double instrumentsPerMember = (double) team.getInstruments() / team.getTotalMembers();

            if (instrumentsPerMember > 0.6) {

                instrumentsCount++;

            }

        }

        // Вывод результатов

        System.out.println("\nКоличество команд с капитанами возрастом до 39 лет: " + captainAgeCount);

        System.out.println("Количество команд, где количество участников младше 21 года меньше 30% от общего числа участников: " + under21MembersCount);

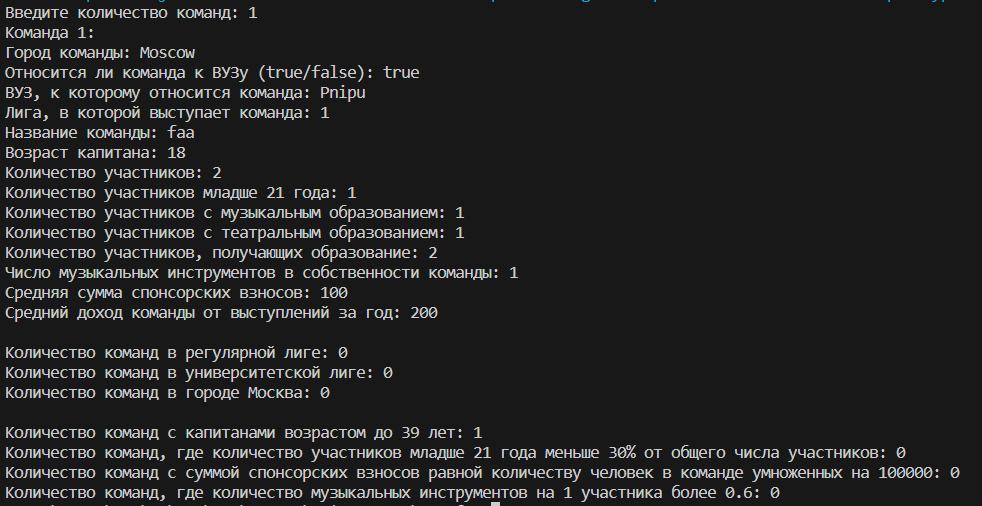
        System.out.println("Количество команд с суммой спонсорских взносов равной количеству человек в команде умноженных на 100000: " + sponsorAmountCount);

        System.out.println("Количество команд, где количество музыкальных инструментов на 1 участника более 0.6: " + instrumentsCount);

    }

}

## 6.3. Тестирование работы программы



# Задание 7

## 7.1. Постановка задачи

Дан текст, если в предложении встрефается название фирм, например ООО «Кнопка», то заменить название на слово «фирма».

## 7.2.1 Решение задачи, код программы

import java.util.regex.\*;

public class task7 {

    public static void main(String[] args) {

        String sentence = "Встречаемся в ООО «Кнопка» завтра в 10:00.";

        // Используем регулярное выражение для поиска названий фирм

        String regex = "\\bООО\\s+«[^»]+»";

        Pattern pattern = Pattern.compile(regex);

        Matcher matcher = pattern.matcher(sentence);

        // Заменяем найденные названия фирм на слово "фирма"

        String result = matcher.replaceAll("фирма");

        System.out.println(result);

    }

}

## 7.3. Тестирование работы программы



# Задание 8

## 8.1. Постановка задачи

Дан текст, в котором зашифрованы слова по следующему принципу: если в слове есть «один» то это сочетание символов заменено на 1; если в слове встречается «втор», то это сочетание символов заменено на 2; если в слове встречается «три», то это сочетание букв заменено на 3. Расшифровать текст.

## 8.2. Решение задачи, код программы

public class task8{

    public static void main(String[] args) {

        String encryptedText = "1 текст 2 котором зашифрованы 3 по следующему принципу";

        String decryptedText = decryptText(encryptedText);

        System.out.println("Расшифрованный текст: " + decryptedText);

    }

    public static String decryptText(String encryptedText) {

        String[] words = encryptedText.split(" ");

        StringBuilder decryptedText = new StringBuilder();

        for (String word : words) {

            if (word.contains("1")) {

                word = word.replace("1", "один");

            } else if (word.contains("2")) {

                word = word.replace("2", "втор");

            } else if (word.contains("3")) {

                word = word.replace("3", "три");

            }

            decryptedText.append(word).append(" ");

        }

        return decryptedText.toString().trim();

    }

}

## 8.3. Тестирование работы программы



# Задание 9

## 9.1. Постановка задачи

Дан текст с комментариями. Из текста удалить все комментарии (ООО…ООО, ААА....ААА, БББ...БББ), если комментарий нет, то сообщить об этом.

## 9.2. Решение задачи, код программы

public class task9{

    public static void main(String[] args) {

        String text = "Это текст с комментариями.\n" +

                "ООО Это комментарий, который должен быть удален.\n" +

                "Это обычная строка текста.\n" +

                "ААА Этот комментарий также нужно удалить.\n" +

                "Это еще одна строка текста.\n" +

                "БББ И этот комментарий тоже нужно удалить.";

        String result = removeComments(text);

        if (result.isEmpty()) {

            System.out.println("Комментарии не найдены.");

        } else {

            System.out.println(result);

        }

    }

    public static String removeComments(String text) {

        String[] lines = text.split("\n");

        StringBuilder resultBuilder = new StringBuilder();

        for (String line : lines) {

            if (!line.startsWith("ООО") && !line.startsWith("ААА") && !line.startsWith("БББ")) {

                resultBuilder.append(line).append("\n");

            }

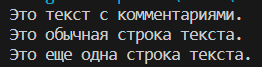
        }

        return resultBuilder.toString().trim();

    }

}

## 9.3. Тестирование работы программы



# Задание 10

## 10.1. Постановка задачи

Дан текст, который содержит данные определенной структуры, разделителем данных об одном объекте является «;», разделитель между объектами «.». «Манекенщица»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира). Разбить данные по объектам на массив строк. Посчитать количество людей с российскими номерами телефона (телефон начинается на +7) и возрастом от 20 до 23 лет. Вывести данные про самую молодую манекенщицу

## 10.2. Решение задачи, код программы

import java.util.ArrayList;

import java.util.List;

public class task10{

    public static void main(String[] args) {

        String data = "Манекенщица1: Иванова; Мария; Сергеевна; Женский; Российская; 175; 55; 1998, 05, 15; 123456, Россия, Московская область, Центральный район, Москва, Улица1, 1, 1." +

                      "Манекенщица2: Петрова; Елена; Александровна; Женский; Российская; 170; 60; 1995, 10, 20; 654321, Россия, Московская область, Западный район, Москва, Улица2, 2, 2." +

                      "Манекенщица3: Сидорова; Анна; Васильевна; Женский; Российская; 180; 65; 2000, 01, 10; 987654, Россия, Московская область, Восточный район, Москва, Улица3, 3, 3.";

        String[] objects = data.split("\\.");

        List<Mannequin> mannequins = new ArrayList<>();

        for (String object : objects) {

            String[] fields = object.split(";");

            if (fields.length == 9) {

                String lastName = fields[0].trim();

                String firstName = fields[1].trim();

                String middleName = fields[2].trim();

                String gender = fields[3].trim();

                String nationality = fields[4].trim();

                int height = Integer.parseInt(fields[5].trim());

                int weight = Integer.parseInt(fields[6].trim());

                String[] birthDateFields = fields[7].trim().split(", ");

                int year = Integer.parseInt(birthDateFields[0]);

                int month = Integer.parseInt(birthDateFields[1]);

                int day = Integer.parseInt(birthDateFields[2]);

                String[] addressFields = fields[8].trim().split(", ");

                String postalCode = addressFields[0];

                String country = addressFields[1];

                String region = addressFields[2];

                String district = addressFields[3];

                String city = addressFields[4];

                String street = addressFields[5];

                String house = addressFields[6];

                String apartment = addressFields[7];

                Mannequin mannequin = new Mannequin(lastName, firstName, middleName, gender, nationality, height, weight, year, month, day, postalCode, country, region, district, city, street, house, apartment);

                mannequins.add(mannequin);

            }

        }

        int count = 0;

        Mannequin youngestMannequin = null;

        for (Mannequin mannequin : mannequins) {

            if (mannequin.getPhoneNumber().startsWith("+7") && mannequin.getAge() >= 20 && mannequin.getAge() <= 23) {

                count++;

                if (youngestMannequin == null || mannequin.getAge() < youngestMannequin.getAge()) {

                    youngestMannequin = mannequin;

                }

            }

        }

        System.out.println("Количество людей с российскими номерами телефона (телефон начинается на +7) и возрастом от 20 до 23 лет: " + count);

        if (youngestMannequin != null) {

            System.out.println("Самая молодая манекенщица: " + youngestMannequin.getFullName() + ", возраст: " + youngestMannequin.getAge());

        }

    }

}

class Mannequin {

    private String lastName;

    private String firstName;

    private String middleName;

    private String gender;

    private String nationality;

    private int height;

    private int weight;

    private int year;

    private int month;

    private int day;

    private String postalCode;

    private String country;

    private String region;

    private String district;

    private String city;

    private String street;

    private String house;

    private String apartment;

    public Mannequin(String lastName, String firstName, String middleName, String gender, String nationality, int height, int weight, int year, int month, int day, String postalCode, String country, String region, String district, String city, String street, String house, String apartment) {

        this.lastName = lastName;

        this.firstName = firstName;

        this.middleName = middleName;

        this.gender = gender;

        this.nationality = nationality;

        this.height = height;

        this.weight = weight;

        this.year = year;

        this.month = month;

        this.day = day;

        this.postalCode = postalCode;

        this.country = country;

        this.region = region;

        this.district = district;

        this.city = city;

        this.street = street;

        this.house = house;

        this.apartment = apartment;

    }

    public String getFullName() {

        return lastName + " " + firstName + " " + middleName;

    }

    public String getPhoneNumber() {

        return postalCode;

    }

    public int getAge() {

        // Assuming current date is 2023-06-04

        int currentYear = 2023;

        int currentMonth = 5;

        int currentDay = 4;

        int age = currentYear - year;

        if (currentMonth < month || (currentMonth == month && currentDay < day)) {

            age--;

        }

        return age;

    }

}

## 10.3. Тестирование работы программы



# Задание 11

Создать программу форматирования массива строк используя три разных регулярных выражения, т.е. регулярные выражения разного типа, созданные при использовании разных метасимволов и их сочетаний.

## 11.2. Решение задачи, код программы

import java.util.Arrays;

public class task11 {

    public static void main(String[] args) {

        String[] array = {"hello123", "world456", "java789"};

        // Регулярное выражение 1: Заменить все цифры на пустую строку

        String regex1 = "\\d";

        String[] formattedArray1 = formatArray(array, regex1);

        System.out.println("Formatted Array 1: " + Arrays.toString(formattedArray1));

        // Регулярное выражение 2: Заменить все буквы на символ 'X'

        String regex2 = "[a-zA-Z]";

        String[] formattedArray2 = formatArray(array, regex2);

        System.out.println("Formatted Array 2: " + Arrays.toString(formattedArray2));

        // Регулярное выражение 3: Удалить все символы, кроме букв и цифр

        String regex3 = "[^a-zA-Z0-9]";

        String[] formattedArray3 = formatArray(array, regex3);

        System.out.println("Formatted Array 3: " + Arrays.toString(formattedArray3));

    }

    public static String[] formatArray(String[] array, String regex) {

        String[] formattedArray = new String[array.length];

        for (int i = 0; i < array.length; i++) {

            formattedArray[i] = array[i].replaceAll(regex, "");

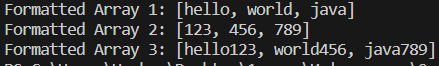
        }

        return formattedArray;

    }

}

## 11.3. Тестирование работы программы



# Задание 12

## 12.1. Постановка задачи

Дан массив строк содержащих информацию о семьях 10-б класса СОШ №12 (возраст, пол, численность семьи, средний балл, место учебы, место жительства, количество кружков). Данные не структурированы. В цикле с использование регулярных выражений (массив данных) выполнить автозамену информации, ввести сокращение ряда понятий: «квартира» и его различные записи до «кв.», «количество» и его различные записи до «кол-во», «улица» и его различные записи до «ул.», «город» и его различные записи до «г.». Исходную и измененную информацию вывести построчно: сначала исходная строка, затем измененная..

## 12.2. Решение задачи, код программы

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class task12 {

    public static void main(String[] args) {

        String[] familyData = {

            "Возраст: 30, Пол: М, Численность семьи: 4, Средний балл: 4.5, Место учебы: Школа №12, Место жительства: улица Пушкина, Количество кружков: 2",

            "Возраст: 25, Пол: Ж, Численность семьи: 3, Средний балл: 4.8, Место учебы: Школа №12, Место жительства: город Москва, Количество кружков: 1",

            // ... add more data strings as needed

        };

        Pattern pattern = Pattern.compile("(квартира|кв\\.|количество|кол-во|улица|ул\\.|город|г\\.)");

        String replacement = "$1.";

        for (String data : familyData) {

            Matcher matcher = pattern.matcher(data);

            String replacedData = matcher.replaceAll(replacement);

            System.out.println("Исходная строка: " + data);

            System.out.println("Измененная строка: " + replacedData);

            System.out.println();

        }

    }

}

## 12.3. Тестирование работы программы

