Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 10

тема «Файлы»

по дисциплине «Информатика»

Выполнил: студент группы ИСТ-21-1б Евсеев В.С.

Проверил: доцент каф. ВММБ Каменских А.А.

Пермь, 2022

Содержание

[Задание 1 4](#_Toc103966638)

[1.1. Общая постановка задачи 4](#_Toc103966639)

[1.2. Решение задачи, код программы 4](#_Toc103966640)

[1.3. Тестирование программы с проверкой 6](#_Toc103966641)

[Задание 2 7](#_Toc103966642)

[2.1. Общая постановка задачи 7](#_Toc103966643)

[2.2. Решение задачи, код программы 7](#_Toc103966644)

[2.3. Тестирование программы с проверкой 7](#_Toc103966645)

[Задание 3 9](#_Toc103966646)

[3.1. Общая постановка задачи 9](#_Toc103966647)

[3.2. Решение задачи, код программы 9](#_Toc103966648)

[3.3. Тестирование программы с проверкой 10](#_Toc103966649)

[Задание 4 11](#_Toc103966650)

[4.1. Общая остановка задачи 11](#_Toc103966651)

[4.2. Решение задачи, код программы 11](#_Toc103966652)

[4.3. Тестирование программы с проверкой 13](#_Toc103966653)

[Задание 5 15](#_Toc103966654)

[5.1. Общая постановка задачи 15](#_Toc103966655)

[5.2. Решение задачи, код программы 15](#_Toc103966656)

[5.3. Тестирование программы с проверкой 16](#_Toc103966657)

[Задание 6 17](#_Toc103966658)

[6.1. Общая постановка задачи 17](#_Toc103966659)

[6.2. Решение задачи, код программы 17](#_Toc103966660)

[6.3. Тестирование программы с проверкой 18](#_Toc103966661)

[Задание 7 19](#_Toc103966662)

[7.1. Общая постановка задачи 19](#_Toc103966663)

[7.2. Решение задачи, код программы 19](#_Toc103966664)

[7.3. Тестирование программы с проверкой 20](#_Toc103966665)

[Задание 8 22](#_Toc103966666)

[8.1. Общая постановка задачи 22](#_Toc103966667)

[8.2. Решение задачи, код программы 22](#_Toc103966668)

[8.3. Тестирование программы с проверкой 26](#_Toc103966669)

[Задание 9 28](#_Toc103966670)

[9.1. Общая остановка задачи 28](#_Toc103966671)

[9.2. Решение задачи, код программы 28](#_Toc103966672)

[9.3. Тестирование программы с проверкой 32](#_Toc103966673)

[Задание 10 34](#_Toc103966674)

[10.1. Общая постановка задачи 34](#_Toc103966675)

[10.2. Решение задачи, код программы 34](#_Toc103966676)

[10.3. Тестирование программы с проверкой 36](#_Toc103966677)

[Задание 11 39](#_Toc103966678)

[11.1. Общая постановка задачи 39](#_Toc103966679)

[11.2. Решение задачи, код программы 39](#_Toc103966680)

[11.3. Тестирование программы с проверкой 40](#_Toc103966681)

[12 Задание 41](#_Toc103966682)

[12.1. Общая постановка задачи 41](#_Toc103966683)

[12.2. Решение задачи, код программы 41](#_Toc103966684)

[12.3. Тестирование программы с проверкой 42](#_Toc103966685)

[Задание 13 44](#_Toc103966686)

[13.1. Общая постановка задачи 44](#_Toc103966687)

[13.2. Решение задачи, код программы 44](#_Toc103966688)

[13.3. Тестирование программы с проверкой 48](#_Toc103966689)

[Задание 14 51](#_Toc103966690)

[14.1. Общая постановка задачи 51](#_Toc103966691)

[14.2. Решение задачи, код программы 51](#_Toc103966692)

[14.3. Тестирование программы с проверкой 53](#_Toc103966693)

# Задание 1

## 1.1. Общая постановка задачи

Переделать программу задания 3 лабораторной работы 9. Массив данных записан в файл и считывается в программе для обработки. Результаты обработки массива согласно вопросам задания 3 лабораторной работы 9 записываются в конец файла построчно с заголовками расшифровывающими, что за данные представлены ниже.

## 1.2. Решение задачи, код программы

**import** java.io.\*;  
  
**public class** Task101 {  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 1.txt"**);  
 FileOutputStream fileOutputStream = **new** FileOutputStream(file, **true**);  
 FileInputStream fileInputStream = **new** FileInputStream(file);  
 **int** n = (**int**) file.length();  
 **byte**[] masByte = **new byte**[n];  
 fileInputStream.read(masByte);  
 String string = **new** String(masByte);  
 String[] stringSplit = string.split(**"\s"**);  
 **char**[] masChar = **new char**[stringSplit.**length**];  
 **for** (**int** i = 0; i < stringSplit.**length**; i++) {  
 masChar[i] = stringSplit[i].charAt(0);  
 }  
  
 *// проверка на вхождение символов* **int** countLat = 0, countCommas = 0, parameterI = 0;  
 **boolean** checkU = **false**;  
 **boolean** checkBo = **false**;  
 **boolean** checkOb = **false**;  
 **boolean** checkDigit = **false**;  
 **boolean** checkSiSi1 = **false**;  
 **boolean** checkSjSj1 = **false**;  
 **for** (**int** i = 0; i < masChar.**length**; i++) { *// по длине всего массива* **for** (**int** k = 97; k <= 122; k++) { *// вхождение прописных латинских букв* **if** (masChar[i] == k) {  
 countLat++;  
 }  
 }  
 **if** (masChar[i] == **'ю'** | masChar[i] == **'Ю'**) { *// вхождение буквы "ю"* checkU = **true**;  
 }  
 **if** (masChar[i] == **','**) { *// вхождение запятой* countCommas++;  
 }  
 }  
 **for** (**int** i = 0; i < masChar.**length** - 1; i++) { *// идёт до предпоследнего символа* **if** (masChar[i] == **'в'** & masChar[i + 1] == **'о'**) { *// вхождение 'во'* checkBo = **true**;  
 }  
 **if** (masChar[i] == **'о'** & masChar[i + 1] == **'в'**) {*// вхождение 'ов'* checkOb = **true**;  
 }  
 **if** (Character.*isDigit*(masChar[i]) & Character.*isDigit*(masChar[i + 1])) { *// пара соседних одинаковых чисел* **if** (masChar[i] == masChar[i + 1]) {  
 checkDigit = **true**;  
 }  
 }  
 }  
 **for** (**int** k = 2; k < masChar.**length** - 1; k++) { *// идёт до предпоследнего символа* **if** ((Character.*isLowerCase*(masChar[k]) & Character.*isUpperCase*(masChar[k + 1])) |  
 (Character.*isUpperCase*(masChar[k]) & Character.*isLowerCase*(masChar[k + 1]))) {  
 *// si, si+1 это одинаковые буквы отличающиеся регистром* checkSiSi1 = **true**;  
 parameterI = k;  
 **break**;  
 }  
 }  
 **for** (**int** c = parameterI + 1; c < masChar.**length** - 1; c++) { *// идёт до предпоследнего символа* **if** (Character.*isDigit*(masChar[c])) {  
 **if** ((Character.*digit*(masChar[c], 10) == 0) & (Character.*digit*(masChar[c + 1], 10) == 0)) {  
 *// sj, sj+1 это 0* checkSjSj1 = **true**;  
 **break**;  
 }  
 }  
 }  
  
 *// вывод* String stringForOutput = **"\nКоличество латинских прописных букв: "** + String.*valueOf*(countLat);  
 **byte**[] byteForOutput1 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput1);  
 **if** (checkU) {  
 stringForOutput = **"\nВ массив входит буква 'ю'"**;  
 **byte**[] byteForOutput2 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput2);  
 }  
 **if** (countCommas >= 2) {  
 stringForOutput = **"\nВ массиве имеются две или более запятые"**;  
 **byte**[] byteForOutput3 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput3);  
 }  
 **if** (checkBo) {  
 stringForOutput = **"\nВ массив входит последовательность 'во'"**;  
 **byte**[] byteForOutput4 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput4);  
 }  
 **if** (checkOb) {  
 stringForOutput = **"\nВ массив входит последовательность 'ов'"**;  
 **byte**[] byteForOutput5 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput5);  
 }  
 **if** (checkDigit) {  
 stringForOutput = **"\nВ массив входит пара соседствующих одинаковых цифр"**;  
 **byte**[] byteForOutput6 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput6);  
 }  
 **if** (checkSiSi1 & checkSjSj1) {  
 stringForOutput = **"\nВ массиве есть такие натуральные i и j, что 1 < i < j < n и что si, si+1 "** +  
 **"это одинаковые буквы отличающиеся регистром, a sj, sj+1 это 0"**;  
 **byte**[] byteForOutput7 = stringForOutput.getBytes();  
 fileOutputStream.write(byteForOutput7);  
 }  
 fileOutputStream.close();  
 }  
}

## 1.3. Тестирование программы с проверкой

Для проверки задания в MS Excel создана таблица данных. В таблицу B2:O3 записаны индексы массива и их значения. Далее по строкам идёт проверка A5, A13, A16 – аналитическая, A7, A9, A11 – с помощью формулы,

A7=ЕСЛИ(ИЛИ(B3="Ю"; B3="ю");1;0)

A9=ЕСЛИ(B3=",";1;0)

A11=ЕСЛИ(ИЛИ(И(B3="в";C3="о");И(B3="о";C3="в"));1;0)

В таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 1

Тестирование работы программы и проверка решения задачи 1

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | До    После |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 2

## 2.1. Общая постановка задачи

Записать в файл таблицу из задания 7 лабораторной работы 6. Если в таблице присутствуют дробные ответы, то вывести данные в формате 4 знака после запятой.

## 2.2. Решение задачи, код программы

**import** java.io.\*;  
  
**public class** Task102 {  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 2.txt"**);  
 FileOutputStream fileOS = **new** FileOutputStream(file, **true**);  
 **int** i, j;  
 String stringOS = **""**;  
 **byte**[] byteOS;  
 **for** (i = 1; i <= 9; i++) {  
 **for** (j = 1; j <= 9; j++) {  
 stringOS = String.*valueOf*(i) + **" x "** + String.*valueOf*(j) + **" = "** + String.*valueOf*(i \* j) + **"\t"**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
 }  
 stringOS = **"\n"**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
 }  
 fileOS.close();  
 }  
}

## 2.3. Тестирование программы с проверкой

Блок-схема 1. Решение задачи 2

Цикл for

-

+

Цикл for

-

+

i=1, j=1

i<=9

j<=9

Вывод результата



j+1

Пустое слово

i+1

В таблице 2 представлено тестирование работы программы на языке Java.

Таблица 2

Тестирование работы программы на языке Java для задачи 2

|  |  |
| --- | --- |
| Этап | Решение Java |
| До |  |
| После |  |

Сравнение решения задачи с использованием одного прикладного пакета показала, что решения задачи в Java и блок-схемы совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 3

## 3.1. Общая постановка задачи

Напечатать все слова из текстового файла, состоящие только из маленьких латинских букв.

## 3.2. Решение задачи, код программы

**import** java.io.\*;  
  
**import static** java.lang.System.***out***;  
  
**public class** Task103 {  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 3.txt"**);  
 FileInputStream fileIS = **new** FileInputStream(file);  
 **int** n = (**int**) file.length();  
 **byte**[] byteIS = **new byte**[n];  
 fileIS.read(byteIS);  
 String string = **new** String(byteIS);  
 String[] stringSplit = string.split(**"\s"**);  
 **for** (**int** i = 0; i < stringSplit.**length**; i++) {  
 **if** (*check*(stringSplit[i].toCharArray())) {  
 ***out***.println(stringSplit[i]);  
 }  
 }  
 }  
  
 **public static boolean** check(**char**[] array) {  
 **int** count = 0;  
 **for** (**int** k = 97; k <= 122; k++) {  
 **for** (**int** i = 0; i < array.**length**; i++) {  
 **if** (array[i] == k) {  
 count++;  
 }  
 }  
 }  
 **return** count == array.**length**;  
 }  
}

## 3.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица данных с исходной строкой и преобразованной.

В таблице 3 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 3

Тестирование работы программы и проверка решения задачи 3

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | Сам файл    Слова по условию |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 4

## 4.1. Общая остановка задачи

Выделить в текстовом файле все слова, разделенные символами-разделителями «\_.,;:\n\t!?», и все слова, в которых гласных русских букв меньше чем согласных букв, построчно записать в другой текстовый файл.

## 4.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
  
**public class** Task104 {  
  
 **static** ArrayList<String> *listWordsCondition1* = **new** ArrayList<>();  
 **static** ArrayList<String> *listWordsCondition2* = **new** ArrayList<>();  
 **static** ArrayList<String> *listAllWords* = **new** ArrayList<>();  
  
 **public static void** main(String[] args) **throws** IOException{  
 File file1 = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 4\_1.txt"**);  
 File directory = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java"**);  
 File file2 = **new** File(directory, **"Для 4\_2.txt"**);  
 FileInputStream file1IS = **new** FileInputStream(file1);  
 FileOutputStream file2OS = **new** FileOutputStream(file2);  
  
 **int** n = (**int**) file1.length();  
 **byte**[] byteIS = **new byte**[n];  
 file1IS.read(byteIS);  
 String stringIS = **new** String(byteIS);  
  
 String[] stringIS\_SplitSpace = stringIS.split(**"\s"**);  
 String[] masPattern = {**"\_"**, **"\\."**, **"\\;"**, **":"**, **"\\\n"**, **"\\\t"**, **"\\!"**, **"\\?"**};  
 String[] stringsIS\_SplitSpace\_SplitPattern;  
 Pattern pattern;  
 Matcher matcher;  
 **for** (**int** i = 0 ; i < stringIS\_SplitSpace.**length**; i++) {  
 **int** countPattern = 0;  
 **for** (String s : masPattern) {  
 pattern = Pattern.*compile*(s);  
 matcher = pattern.matcher(stringIS\_SplitSpace[i]);  
 **if** (matcher.find()) {  
 countPattern++;  
 stringsIS\_SplitSpace\_SplitPattern = stringIS\_SplitSpace[i].split(s);  
 **if** (i == 0) {  
 *listWordsCondition1*.addAll(List.*of*(stringsIS\_SplitSpace\_SplitPattern));  
 } **else** {  
 **for** (String value : stringsIS\_SplitSpace\_SplitPattern) {  
 **if** (*listWordsCondition1*.contains(value)) {  
 **int** index = *listWordsCondition1*.indexOf(value);  
 *listWordsCondition1*.set(index, value);  
 } **else if** (!*listWordsCondition1*.contains(value)) {  
 *listWordsCondition1*.add(value);  
 }  
 }  
 }  
 }  
 }  
 **if** (countPattern == 0) {  
 *listAllWords*.add(stringIS\_SplitSpace[i]);  
 }  
 }  
  
 *listAllWords*.addAll(*listWordsCondition1*);  
 **int**[] symbolVowel = {1040, 1045, 1025, 1048, 1054, 1059, 1067, 1069, 1070, 1071,1072, 1077, 1105, 1080, 1086, 1091,  
 1099, 1101, 1102, 1103};  
 **char**[] masChar;  
 **for** (**int** i = 0; i < *listAllWords*.size(); i++) {  
 **int** countSymbolVowel = 0;  
 **int** countSymbolConsonant = 0;  
 masChar = *listAllWords*.get(i).toCharArray();  
 **for** (**char** c : masChar) {  
 **int** countEntering = 0;  
 **if** (Character.*isLetter*(c)) { *// проверка: гласных русских букв меньше чем согласных букв* **for** (**int** j : symbolVowel) {  
 **if** (c == j) {  
 countSymbolVowel++; *// считает кол-во гласных букв* countEntering++;  
 }  
 }  
 **if** (countEntering == 0) {  
 countSymbolConsonant++; *// если ни разу не вошла гласная, то это согласная* }  
 }  
 }  
 **if** (countSymbolConsonant > countSymbolVowel) {  
 **if** (i == 0) {  
 *listWordsCondition2*.add(*listAllWords*.get(i));  
 }  
 **else** {  
 **if** (*listWordsCondition2*.contains(*listAllWords*.get(i))) {  
 **int** index = *listWordsCondition2*.indexOf(*listAllWords*.get(i));  
 *listWordsCondition2*.set(index, *listAllWords*.get(i));  
 }  
 **else if** (!*listWordsCondition2*.contains(*listAllWords*.get(i))) {  
 *listWordsCondition2*.add(*listAllWords*.get(i));  
 }  
 }  
 }  
 }  
  
 *output*(file2OS);  
 }  
  
 **public static void** output(FileOutputStream file2OS) **throws** IOException {  
 String stringOS = **"Слова, разделенные символами-разделителями «\_.,;:\\n\\t!?»:"**;  
 **byte**[] byteOS = stringOS.getBytes();  
 file2OS.write(byteOS);  
  
 **for** (String str : *listWordsCondition1*) {  
 stringOS = **"\n"** + str;  
 byteOS = stringOS.getBytes();  
 file2OS.write(byteOS);  
 }  
  
 stringOS = **"\n"** + **"Слова, в которых гласных русских букв меньше чем согласных букв"**;  
 byteOS = stringOS.getBytes();  
 file2OS.write(byteOS);  
  
 **for** (String str : *listWordsCondition2*) {  
 stringOS = **"\n"** + str;  
 byteOS = stringOS.getBytes();  
 file2OS.write(byteOS);  
 }  
  
 file2OS.close();  
 }  
}

## 4.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица данных, содержащая исходные слова и выборку по условию задачи.

В таблице 4 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 4

Тестирование работы программы и проверка решения задачи 4

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | Исходный файл    Файл со словами по условию |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 5

## 5.1. Общая постановка задачи

Дано два файла с предложениями. Дописать в конец второго файла предложения из первого файла, в которых есть слова, которые пишутся через дефис.

## 5.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
  
**public class** Task105 {  
  
 **static** ArrayList<String> *listOfWords* = **new** ArrayList<>();  
  
 **public static void** main(String[] args) **throws** IOException{  
 File file1 = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 5\_1.txt"**);  
 File file2 = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 5\_2.txt"**);  
 FileInputStream fileIS = **new** FileInputStream(file1);  
 FileOutputStream fileOS = **new** FileOutputStream(file2, **true**);  
  
 **int** n = (**int**) file1.length();  
 **byte**[] byteIS = **new byte**[n];  
 fileIS.read(byteIS);  
 String stringIS = **new** String(byteIS);  
 String[] stringIS\_SplitSentence = stringIS.split(**"\\.\s"**);  
 String[] stringIS\_SplitWords;  
  
 **for** (**int** i = 0; i < stringIS\_SplitSentence.**length**; i++) {  
 stringIS\_SplitWords = stringIS\_SplitSentence[i].split(**"\s"**);  
 **for** ( **int** j = 0; j < stringIS\_SplitWords.**length**; j++) {  
 **if** (stringIS\_SplitWords[j].contains(**"-"**)) {  
 *listOfWords*.add(stringIS\_SplitWords[j]);  
 }  
 }  
 }  
  
 StringBuilder stringOS = **new** StringBuilder();  
 **for** (**int** i = 0; i < *listOfWords*.size(); i++) {  
 stringOS.append(**"\n"**);  
 stringOS.append(*listOfWords*.get(i));  
 }  
 **byte**[] byteOS = stringOS.toString().getBytes();  
 fileOS.write(byteOS);  
 fileOS.close();  
 }  
}

## 5.3. Тестирование программы с проверкой

Для проверки в MS Excel создана таблица данных, содержащая предложения из 1 файла и преобразованные предложения 2 файла.

В таблице 5 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 5

Тестирование работы программы и проверка решения задачи 5

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | Файл 1    Файл 2 до    Файл 2 после |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 6

## 6.1. Общая постановка задачи

Входной файл содержит сведения о волейболистах пермского края: ФИО, клуб за который выступает, с какого года, результативность в очках, цена контракта. Подсчитать количество волейболистов по клубам и их общую результативность в очках, дописать ответы построчно в конец файла.

## 6.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
  
  
**public class** Task106 {  
  
 **static** ArrayList<String> *listOfNameOfClubs* = **new** ArrayList<>();  
 **static** ArrayList<Integer> *listOfResult* = **new** ArrayList<>();  
 **static** ArrayList<Integer> *listOfMembers* = **new** ArrayList<>();  
  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 6.txt"**);  
 FileInputStream fileIS = **new** FileInputStream(file);  
 FileOutputStream fileOS = **new** FileOutputStream(file,**true**);  
  
 **int** n = (**int**) file.length();  
 **byte**[] byteIS = **new byte**[n];  
 fileIS.read(byteIS);  
 String stringIS = **new** String(byteIS);  
 String[] stringIS\_SplitSentence = stringIS.split(**"\\.\s"**);  
 String[] stringIS\_SplitWords;  
  
 **for** (**int** i = 0; i < stringIS\_SplitSentence.**length**; i++) {  
 stringIS\_SplitWords = stringIS\_SplitSentence[i].split(**",\s"**);  
 String club = stringIS\_SplitWords[1];  
 **int** score = Integer.*parseInt*(stringIS\_SplitWords[3]);  
 **if** (i == 0) {  
 *listOfNameOfClubs*.add(club);  
 *listOfResult*.add(score);  
 *listOfMembers*.add(1);  
 }  
 **else** {  
 **if** (!*listOfNameOfClubs*.contains(club)) {  
 *listOfNameOfClubs*.add(club);  
 *listOfResult*.add(score);  
 *listOfMembers*.add(1);  
 }  
 **else if** (*listOfNameOfClubs*.contains(club)) {  
 **int** index;  
 index = *listOfNameOfClubs*.indexOf(club);  
 *listOfResult*.set(index, *listOfResult*.get(index) + score);  
 *listOfMembers*.set(index, *listOfMembers*.get(index) + 1);  
 }  
 }  
 }  
  
 String stringOS\_InfAboutClub;  
 **byte**[] byteOS;  
 **for** (**int** i = 0; i < *listOfNameOfClubs*.size(); i++) {  
 String club = *listOfNameOfClubs*.get(i);  
 String members = String.*valueOf*(*listOfMembers*.get(i));  
 String score = String.*valueOf*(*listOfResult*.get(i));  
 stringOS\_InfAboutClub = **"\n"** + club + **". Количество участников: "** + members + **", общий результат: "** + score;  
 byteOS = stringOS\_InfAboutClub.getBytes();  
 fileOS.write(byteOS);  
 }  
 fileOS.close();  
 }  
}

## 6.3. Тестирование программы с проверкой

Для тестирования в MS Excel созданы таблицы данных, содержащие исходную информацию и преобразованную.

В таблице 6 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 6

Тестирование работы программы и проверка решения задачи 6

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | До    После |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 7

## 7.1. Общая постановка задачи

Записать в файл значения функции f(x) из задания 3 лабораторной работы 5 для всего диапазона переменной в виде: значение x; значение f(x). В начале файла должна быть указана постановка задачи, т.е. диапазон аргумента и вид функции (запись на языке java). Перед выводом цифровых значений должен быть сделан заголовок о том, в каком порядке идет вывод.

## 7.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.text.DecimalFormat;  
  
**import static** java.lang.Math.\*;  
  
**public class** Task107 {  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 7.txt"**);  
 FileOutputStream fileOS = **new** FileOutputStream(file);  
 String stringOS = **"Постановка задачи:"**;  
 **byte**[] byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 stringOS = **"""  
  
 for (double x = -3; x <= 3; x += 0.5) {  
 if (x >= -1) {  
 f = cos(x) + sin(x);  
 } else {  
 f = -(pow((x + 1), 2));  
 }  
 }"""**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 stringOS = **"\n"** + **"значение x"** + **"\t"** + **"значение f(x)"**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 DecimalFormat decimalFormat = **new** DecimalFormat(**"0.000"**); *// число f до двух знаков после запятой* **double** f;  
 **for** (**double** x = -3; x <= 3; x += 0.5) {  
 **if** (x >= -1) {  
 f = *cos*(x) + *sin*(x);  
 stringOS = **"\n"** + **"\s\s\s"** + x + **"\s\s\s\s\s\s\s"** + decimalFormat.format(f);  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
 } **else** {  
 f = -(*pow*((x + 1), 2));  
 stringOS = **"\n"** + **"\s\s\s"** + x + **"\s\s\s\s\s\s\s"** + decimalFormat.format(f);  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
 }  
 }  
 fileOS.close();  
 }  
}

## 7.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица данных, содержащая значение аргумента x и значение функции f(x).

В таблице 7 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 7

Тестирование работы программы и проверка решения задачи 7

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | До    После |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 8

## 8.1. Общая постановка задачи

Составить файл постановки и решения задачи для попадания точки в область лабораторная работа 5 задание 5. Структура файла: постановка задачи; массив точек для проверки; стилизованный ответ. Постановка задачи (для каждой фигуры, с заголовком о номере фигуры): уравнение кривой с порядковым номером, отрезок на котором она определена. Массив точек для проверки: номер точки, координата x; координата y. Стилизованный ответ: номер точки, ответ о попадании точки в область, на границу или о том, что точка не попала.

## 8.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.Scanner;  
  
**import static** java.lang.System.***out***;  
  
**public class** Task108 {  
 **public static void** main(String[] args) **throws** IOException {  
 File directory = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java"**);  
 File file = **new** File(directory, **"Для 8.txt"**);  
 FileWriter fileWriter = **new** FileWriter(file);  
 BufferedWriter fileBW = **new** BufferedWriter(fileWriter);  
  
 fileBW.write(**"Уравнения кривых первой фигуры:\n"**);  
 String[][] equationsOfTheFirstFigure = {{**"y = x + 4"**, **"x ∈ [-5, -3]"**},  
 {**"y = Math.sqrt(4 - Math.pow((x + 1), 2)) + 1"**, **"x ∈ [-3, -1]"**},  
 {**"y = 2.0 / 3 \* x + 11.0 / 3"**, **"x ∈ [-1, 2]"**},  
 {**"y = 5.0 / 3 \* x + 5.0 / 3"**, **"x ∈ [-1, 2]"**},  
 {**"y = x + 1"**, **"x ∈ [-1, 4]"**},  
 {**"y = -x / 2 + 7"**, **"x ∈ [4, 6]"**},  
 {**"y = x - 2"**, **"x ∈ [4, 6]"**},  
 {**"y = x / 3 + 2.0 / 3"**, **"x ∈ [1, 4]"**},  
 {**"y = -Math.sqrt(4 - Math.pow((x + 1), 2)) + 1"**, **"x ∈ [-1, 1]"**},  
 {**"y = x"**, **"x ∈ [-2, -1]"**},  
 {**"y = -2 \* x - 6"**, **"x ∈ [-3, -2]"**},  
 {**"y = 2 \* x + 6"**, **"x ∈ [-4, -3]"**},  
 {**"-x - 6"**, **"x ∈ [-5, -4]"**}};  
 **for** (**int** i = 0; i < 13; i++) {  
 **for** (**int** j = 0; j < 2 - 1; j++) {  
 String equation = (i + 1) + **") "** + equationsOfTheFirstFigure[i][j] + **", "** + equationsOfTheFirstFigure[i][j+1];  
 fileBW.write(equation);  
 }  
 fileBW.newLine();  
 }  
  
 fileBW.newLine();  
 fileBW.write(**"Уравнения кривых второй фигуры:\n"**);  
 String[][] equationsOfTheSecondFigure = {{**"y = x / 2 - 5"**, **"x ∈ [0, 4]"**},  
 {**"y = -Math.sqrt(4 - Math.pow((x - 4), 2)) - 1"**, **"x ∈ [2, 4]"**},  
 {**"y = 1 - x"**, **"x ∈ [2, 3]"**},  
 {**"y = x - 5"**, **"x ∈ [3, 4]"**},  
 {**"y = 0"**, **"x = 4"**},  
 {**"y = 0"**, **"x ∈ [4, 5]"**},  
 {**"y = 5 - x"**, **"x ∈ [4, 5]"**},  
 {**"y = Math.sqrt(4 - Math.pow((x - 4), 2)) - 1"**, **"x ∈ [4, 6]"**},  
 {**"y = 5 - x"**, **"x ∈ [6, 7]"**},  
 {**"y = -2"**, **"x ∈ [4, 7]"**},  
 {**"y = -3 \* x + 10"**, **"x ∈ [4, 5]"**},  
 {**"y = -x"**, **"x ∈ [4, 5]"**},  
 {**"y = x / 4 - 5)"**, **"x ∈ [0, 4]"**}};  
 **for** (**int** i = 0; i < 13; i++) {  
 **for** (**int** j = 0; j < 2 - 1; j++) {  
 String equation = (i + 1) + **") "** + equationsOfTheSecondFigure[i][j] + **", "** + equationsOfTheSecondFigure[i][j+1];  
 fileBW.write(equation);  
 }  
 fileBW.newLine();  
 }  
  
 *inputCoordinates*(fileBW);  
  
 fileBW.close();  
 }  
  
 **public static void** inputCoordinates (BufferedWriter fileBW) **throws** IOException{  
 Scanner scanner = **new** Scanner(System.***in***);  
 ***out***.println(**"Введите количество точек на проверку:"**);  
 **int** n = scanner.nextInt();  
 fileBW.write(**"\nМассив точек на проверку:"**);  
 String coordinates = **""**;  
 **double**[][] points = **new double**[2][n];  
 **for** (**int** j = 0; j < n; j++) {  
 fileBW.newLine();  
  
 ***out***.printf(**"Введите координату x%d:"**, j+1);  
 ***out***.println();  
 points[0][j] = scanner.nextDouble();  
 coordinates = (j + 1) + **") x = "** + points[0][j] + **", "**;  
 fileBW.write(coordinates);  
  
 ***out***.printf(**"Введите координату y%d:"**, j+1);  
 ***out***.println();  
 points[1][j] = scanner.nextDouble();  
 coordinates = **"y = "** + points[1][j];  
 fileBW.write(coordinates);  
 }  
 *printCheckHitting*(points, n, fileBW);  
 }  
  
 **public static void** printCheckHitting(**double**[][] points, **int** n, BufferedWriter fileBW) **throws** IOException{  
 fileBW.newLine();  
 fileBW.write(**"\nОтвет:\n"**);  
 String answer;  
 **for** (**int** j = 0; j < n; j++) {  
 **if** (*checkHitting*(points[0][j], points[1][j]) == 1) {  
 answer = (**"Точка "** + (j + 1) + **" ("** + points[0][j] + **","** + points[1][j] + **")"** +  
 **" попала в область первой фигуры"**);  
 fileBW.write(answer + **"\n"**);  
 }  
 **if** (*checkHitting*(points[0][j], points[1][j]) == 10) {  
 answer = (**"Точка "** + (j + 1) + **" ("** + points[0][j] + **","** + points[1][j] + **")"** +  
 **" попала на границу первой фигуры"**);  
 fileBW.write(answer + **"\n"**);  
 }  
 **if** (*checkHitting*(points[0][j], points[1][j]) == 2) {  
 answer = (**"Точка "** + (j + 1) + **" ("** + points[0][j] + **","** + points[1][j] + **")"** +  
 **" попала в область второй фигуры"**);  
 fileBW.write(answer + **"\n"**);  
 }  
 **if** (*checkHitting*(points[0][j], points[1][j]) == 20) {  
 answer = (**"Точка "** + (j + 1) + **" ("** + points[0][j] + **","** + points[1][j] + **")"** +  
 **" попала на границу второй фигуры"**);  
 fileBW.write(answer + **"\n"**);  
 }  
 **if** (*checkHitting*(points[0][j], points[1][j]) == -1) {  
 answer = (**"Точка "** + (j + 1) + **" ("** + points[0][j] + **","** + points[1][j] + **")"** +  
 **" не попала ни в область, ни на границу фигур"**);  
 fileBW.write(answer + **"\n"**);  
 }  
 }  
 }  
  
 **public static int** checkHitting(**double** x, **double** y) {  
 **if** (y >= x - 2) { *// попадание в область первой фигуры* **int** Count1 = 0;  
 **if** ((x >= -5) & (x <= -3) & (y < x + 4) & (y > -x - 6) & (y > 2 \* x + 6)) *//1 область* Count1++;  
 **if** ((x >= -3) & (x <= -1) & (y < Math.*sqrt*(4 - Math.*pow*((x + 1), 2)) + 1) & (y > x + 4)) *//2 область* Count1++;  
 **if** ((x >= -1) & (x <= 2) & (y < 2.0 / 3 \* x + 11.0 / 3) & (y > 5.0 / 3 \* x + 5.0 / 3)) *//3 область* Count1++;  
 **if** ((x >= -3) & (x <= -1) & (y < x + 4) & (y > -2 \* x - 6) & (y > x)) *//4 область* Count1++;  
 **if** ((x >= 1) & (x <= 6) & (y < x + 1) & (y < -x / 2 + 7) &  
 (y > x - 2) & (y > x / 3 + 2.0 / 3)) { *//5 область* Count1++;  
 }  
 **if** ((x >= -2) & (x <= 1) & (y < 5 \* x / 3 + 5.0 / 3) & (y > x) & (y > x + 1)) *//6 область* Count1++;  
 **if** ((x >= -1) & (x <= 1) & (y < x) & (y > -Math.*sqrt*(4 - Math.*pow*((x + 1), 2)) + 1)) *//7 область* Count1++;  
 **if** (Count1 > 0)  
 **return** 1;  
 **else** { *// попадание на границу первой фигуры* **int** count1 = 0;  
 **if** ((x >= -5) & (x <= -3) & ((y == x + 4) | (y == -x - 6) | (y == 2 \* x + 6))) *//1 область* count1++;  
 **if** ((x >= -3) & (x <= -1) & ((y == Math.*sqrt*(4 - Math.*pow*((x + 1), 2)) + 1) | (y == x + 4))) *//2 область* count1++;  
 **if** ((x >= -1) & (x <= 2) & ((y == 2.0 / 3 \* x + 11.0 / 3) | (y == 5.0 / 3 \* x + 5.0 / 3))) *//3 область* count1++;  
 **if** ((x >= -3) & (x <= -1) & ((y == x + 4) | (y == -2 \* x - 6) | (y == x))) *//4 область* count1++;  
 **if** ((x >= 1) & (x <= 6) & ((y == x + 1) | (y == -x / 2 + 7) |  
 (y == x - 2) | (y == x / 3 + 2.0 / 3))) { *//5 область* count1++;  
 }  
 **if** ((x >= -2) & (x <= 1) & ((y == 5 \* x / 3 + 5.0 / 3) | (y == x) | (y == x + 1))) *//6 область* count1++;  
 **if** ((x >= -1) & (x <= 1) & ((y == x) | (y == -Math.*sqrt*(4 - Math.*pow*((x + 1), 2)) + 1))) *//7 область* count1++;  
 **if** (count1 > 0)  
 **return** 10;  
 }  
 } **else if** (y < x - 2) { *//попадание в область второй фигуры* **int** Count2 = 0;  
 **if** ((x >= 0) & (x <= 4) & (y < x / 2 - 5) & (y > x / 4 - 5)) { *//1 область* Count2++;  
 }  
 **if** ((x >= 2) & (x <= 4) & (y > -Math.*sqrt*(4 - Math.*pow*((x - 4), 2)) - 1) & (y < 1 - x)) { *//2 область* Count2++;  
 }  
 **if** ((x >= 3) & (x <= 4) & (y > 1 - x) & (y < x - 5) & (y > 1 - x)) { *//3 область* Count2++;  
 }  
 **if** ((x >= 4) & (x <= 6) & (y < Math.*sqrt*(4 - Math.*pow*((x - 4), 2)) - 1) & (y > 5 - x)) { *//4 область* Count2++;  
 }  
 **if** ((x >= 4) & (x <= 7) & (y < 0) & (y < -x + 5) & (y > -2)) { *//5 область* Count2++;  
 }  
 **if** ((x >= 4) & (x >= 5) & (y < -3 \* x + 10) & (y > -x)) { *//6 область* Count2++;  
 }  
 **if** (Count2 > 0) {  
 **return** 2;  
 }  
 **else** {  
 **int** count2 = 0; *// попадание на границу второй фигуры* **if** ((x >= 0) & (x <= 4) & ((y == x / 2 - 5) | (y == x / 4 - 5))) { *//1 область* count2++;  
 }  
 **if** ((x >= 2) & (x <= 4) & ((y == -Math.*sqrt*(4 - Math.*pow*((x - 4), 2)) - 1) | (y < 1 - x))) { *//2 область* count2++;  
 }  
 **if** ((x >= 3) & (x <= 4) & ((y == 1 - x) | (y == x - 5) | (y == 1 - x))) { *//3 область* count2++;  
 }  
 **if** ((x >= 4) & (x <= 6) & ((y == Math.*sqrt*(4 - Math.*pow*((x - 4), 2)) - 1) | (y == 5 - x))) { *//4 область* count2++;  
 }  
 **if** ((x >= 4) & (x <= 7) & ((y == 0) | (y == -x + 5) | (y == -2))) { *//5 область* count2++;  
 }  
 **if** ((x >= 4) & (x >= 5) & ((y == -3 \* x + 10) | (y == -x))) { *//6 область* count2++;  
 }  
 **if** (count2 > 0) {  
 **return** 20;  
 }  
 }  
 }  
 **return** -1;  
 }  
}

## 8.3. Тестирование программы с проверкой

В таблице 8 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 8

Тестирование работы программы и проверка решения задачи 8

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 9

## 9.1. Общая остановка задачи

Создать файл с данными по структурированному вводу данных согласно постановке задачи 6 лабораторной работы 9. Изменить программу задания 6 лабораторной работы 9 с анализом данных по тем же критериям из файла. Ответы построчно дописать в исходный файл.

## 9.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
**import** java.util.Arrays;  
  
**import static** java.lang.System.***out***;  
  
**public class** Task109 {  
  
 **static** ArrayList<String> *listInf* = **new** ArrayList<>(); *// конструкция* **static** ArrayList<String> *listDistrict* = **new** ArrayList<>(); *// районы* **static** ArrayList<Integer> *listDistrictInf* = **new** ArrayList<>(); *// данные по районам* **public static void** main(String[] args) **throws** Exception {  
 String[] masList = {**"Район города"**, **"Номер/Название ветеринарной клиники"**, **"Количество работников"**,  
 **"Количество врачей"**, **"Количество врачей высшей категории"**,  
 **"Количество врачей прошедших повышение квалификации за последние 5 лет"**,  
 **"Общее число пациентов в базе"**, **"Число пациентов собак"**, **"Число пациентов кошек"**, **"Наличие рентгена"**,  
 **"Наличие компьютеров"**, **"Количество компьютеров"**, **"Число клеток для передержки животных"**,  
 **"Средний суммарный доход клиники"**, **"Общая стоимость оборудования клиники"**};  
 *listInf*.addAll(Arrays.*asList*(masList));  
  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 9.txt"**);  
 FileOutputStream fileOS = **new** FileOutputStream(file, **true**);  
 FileInputStream fileIS = **new** FileInputStream(file);  
 **int** fileLength = (**int**) file.length();  
 **byte**[] byteIS = **new byte**[fileLength];  
 fileIS.read(byteIS);  
 String stringIS\_InfAboutVetClinics = **new** String(byteIS);  
  
 **int** n = 20; *// количество клиник* String[] infAboutVetClinic = stringIS\_InfAboutVetClinics.split(**"\n"**);  
  
 *checkInfAboutVetClinics*(fileOS, n, infAboutVetClinic);  
  
 *analyzeInfAboutVetClinics*(infAboutVetClinic, n, fileOS);  
 }  
  
 *// анализируем данные и выделяем то, что нужно по условию задачи* **public static void** analyzeInfAboutVetClinics (String[] infAboutVetClinic, **int** n, FileOutputStream fileOS)  
 **throws** IOException {  
 String[] splitInf;  
 **int**[] answer = **new int**[2];  
 **for** (**int** i = 0; i < n; i++) {  
 splitInf = infAboutVetClinic[i].split(**";"**);  
 **if** (splitInf[1].contains(**"/"**)) { *// если строка содержит "/", то это автоматически значит, что есть номер* answer[0]++;  
 }  
  
 **int** numberDocHighCategory = Integer.*parseInt*(splitInf[4]);  
 **int** numberDocHighCategoryFive = Integer.*parseInt*(splitInf[5]);  
 **int** totalNumberDoc = Integer.*parseInt*(splitInf[2]);  
 **int** totalNumberPatient = Integer.*parseInt*(splitInf[6]);  
 **boolean** conditions = numberDocHighCategory >= 2 & numberDocHighCategoryFive > totalNumberDoc \* 3.0 / 10 &  
 totalNumberPatient > 100;  
 **if** (i == 0) {  
 *listDistrict*.add(splitInf[0]);  
 **if** (conditions) {  
 *listDistrictInf*.add(1);  
 }  
 }  
 **else** {  
 **if** (!*listDistrict*.contains(splitInf[0])) {  
 *listDistrict*.add(splitInf[0]);  
 **if** (conditions) {  
 *listDistrictInf*.add(1);  
 }  
 **else** {  
 *listDistrictInf*.add(0);  
 }  
 } **else if** (*listDistrict*.contains(splitInf[0])) {  
 **if** (conditions) {  
 **int** index = *listDistrict*.indexOf(splitInf[0]);  
 **int** newElement = *listDistrictInf*.get(index) + 1;  
 *listDistrictInf*.set(index, newElement);  
 }  
 }  
 }  
 **int** totalCostEquipment = Integer.*parseInt*(splitInf[14]);  
 **if** ((splitInf[9].equals(**"есть"**) | splitInf[9].equals(**"Есть"**)) & totalCostEquipment > 5000000) {  
 answer[1]++;  
 }  
 }  
 *output*(answer, fileOS);  
 }  
  
 *// проверка на структуру данных* **public static void** checkInfAboutVetClinics(FileOutputStream fileOS, **int** n, String[] infAboutVetClinic) {  
 String[] strMasVetClinic; *// создаем массив, в каждой ячейке - инф. в опр. виде* **for** (**int** w = 0; w < n; w++) {  
 strMasVetClinic = infAboutVetClinic[w].split(**";"**);  
 **if** (*cycleCheckInfAboutVetClinics*(strMasVetClinic, w)) {  
 ***out***.println(**"Пожалуйста повторите ввод данных"**);  
 }  
 **else** {  
 ***out***.println(**"Всё в порядке"**);  
 }  
 }  
 }  
  
 *// сама проверка* **public static boolean** cycleCheckInfAboutVetClinics(String[] strMasVetClinic, **int** w) {  
 ***out***.println(**"Клиника в строке "** + (w + 1) + **":"**);  
 **boolean** marker = **false**;  
 **for** (**char** ch : strMasVetClinic[0].toCharArray()) {  
 **if** (Character.*isDigit*(ch)) {  
 ***out***.println(**"В позиции 1, где указывается "** + *listInf*.get(0) +  
 **", вы ввели не буквы, там должна быть буквенная строка!"**);  
 marker = **true**;  
 }  
 }  
  
 String[] strMasVetClinicBoxTwo = strMasVetClinic[1].split(**"/"**);  
 **if** (strMasVetClinic[1].contains(**"/"**)) {  
 **int** countTwoFirst = 0;  
 **int** countTwoSecond = 0;  
 **for** (**char** ch : strMasVetClinicBoxTwo[0].toCharArray()) {  
 **if** (!Character.*isDigit*(ch)) {  
 marker = **true**;  
 countTwoFirst++;  
 }  
 }  
 **for** (**char** ch : strMasVetClinicBoxTwo[1].toCharArray()) {  
 **if** (!Character.*isLetter*(ch)) {  
 marker = **true**;  
 countTwoSecond++;  
 }  
 }  
 **if** (countTwoFirst > 0) {  
 ***out***.println(**"В позиции 2, где указывается номер клиники, вы ввели не число!"**);  
 }  
 **if** (countTwoSecond > 0) {  
 ***out***.println(**"В позиции 2, где указывается название клиники, вы ввели не буквы!"**);  
 }  
 }  
 **else if** (!strMasVetClinic[1].contains(**"/"**)) {  
 **int** countTwo = 0;  
 **for** (**char** ch : strMasVetClinic[1].toCharArray()) {  
 **if** (!Character.*isLetter*(ch)) {  
 countTwo++;  
 }  
 }  
 **if** (countTwo > 0) {  
 ***out***.println(**"В позиции 2, где указывается название клиники, вы ввели не буквы!"**);  
 }  
 }  
  
 **for** (**int** i = 2; i < 9; i++) {  
 **int** countThreeToNine = 0;  
 **for** (**char** ch : strMasVetClinic[i].toCharArray()) {  
 **if** (Character.*isLetter*(ch)) {  
 countThreeToNine++;  
 }  
 }  
 **if**(countThreeToNine > 0) {  
 marker = **true**;  
 ***out***.println(**"В позиции "** + (i + 1) + **" ("** + *listInf*.get(i) + **") "** + **"вы ввели не число!"**);  
 }  
 }  
  
 **for** (**int** i = 9; i < 11; i++) {  
 **int** countTenToEleven = 0;  
 **for** (**char** ch : strMasVetClinic[i].toCharArray()) {  
 **if** (!Character.*isLetter*(ch)) {  
 countTenToEleven++;  
 }  
 }  
 **if** (countTenToEleven > 0) {  
 marker = **true**;  
 ***out***.println(**"В позиции "** + (i + 1) + **" ("** + *listInf*.get(i) + **") "** + **"вы ввели не буквы, "** +  
 **"укажите наличие (есть/нет)!"**);  
 }  
 **if**(countTenToEleven == 0 & !strMasVetClinic[i].equals(**"есть"**) & !strMasVetClinic[i].equals(**"Есть"**) &  
 !strMasVetClinic[i].equals(**"нет"**) & !strMasVetClinic[i].equals(**"Нет"**)) {  
 ***out***.println(**"В позиции "** + (i + 1) + **" ("** + *listInf*.get(i) + **") "** + **"укажите наличие (есть/нет)"**);  
 }  
 }  
  
 **for** (**int** i = 11; i < 15; i++) {  
 **int** countTwelveToFifteen = 0;  
 **for** (**char** ch : strMasVetClinic[i].toCharArray()) {  
 **if** (!Character.*isDigit*(ch)) {  
 countTwelveToFifteen++;  
 }  
 }  
 **if** (countTwelveToFifteen > 0) {  
 marker = **true**;  
 ***out***.println(**"В позиции "** + (i + 1) + **" ("** + *listInf*.get(i) + **") "** + **"вы ввели не число!"**);  
 }  
 }  
  
 **return** marker;  
 }  
  
 **public static void** output(**int**[] answer, FileOutputStream fileOS) **throws** IOException{  
 String stringOS = **"\n"**;  
 **byte**[] byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 stringOS += **"Количество клиник с номером: "** + answer[0];  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 stringOS = **"\n"** + **"------------------------------------------------------------------------------------------------"**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 **for** (**int** i = 0; i < *listDistrict*.size(); i++) {  
 stringOS = **"\n"** + **"Район: "** + *listDistrict*.get(i) + **", количество клиник, удовлетворяющих условию: "** +  
 *listDistrictInf*.get(i);  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
 }  
  
 stringOS = **"\n"** + **"------------------------------------------------------------------------------------------------"**;  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 stringOS = **"\n"** + **"Количество клиник, где есть рентген с общей стоимостью оборудования клиники более 5 млн. руб: "** +  
 answer[1];  
 byteOS = stringOS.getBytes();  
 fileOS.write(byteOS);  
  
 fileOS.close();  
 }  
}

## 9.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица данных, содержащая информацию о ветеринарных клиниках.

В таблице 9 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 9

Тестирование работы программы и проверка решения задачи 9

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 | До    После |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 10

## 10.1. Общая постановка задачи

На основе программы задания 5 лабораторной работы 6 организовать запись в файл данных о результатах нахождения функции по ряду Маклорена.

Примерный вид файла ответов:

Функция f(x) = … для x = … равняется …

Результаты определения значений функции f(x) = … с помощью ряда Маклорена

Погрешность итерационной процедуры …

Значение функции по Маклорену Погрешность, % Число итераций

10 4 1

## 10.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.text.DecimalFormat;  
**import** java.util.Scanner;  
  
**import static** java.lang.Math.\*;  
**import static** java.lang.System.***out***;  
  
**public class** Task1010 {  
 **public static void** main(String[] args) **throws** IOException{  
 File directory = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java"**);  
 File file = **new** File(directory, **"Для 10.txt"**);  
 FileWriter fileWr = **new** FileWriter(file);  
 BufferedWriter fileBF = **new** BufferedWriter(fileWr);  
 *McLarenSeries*(fileBF);  
 }  
  
 **public static void** McLarenSeries(BufferedWriter fileBF) **throws** IOException{  
 Scanner scanner = **new** Scanner(System.***in***);  
 ***out***.println(**"Введите переменную x меньше 1:"**);  
 **double** x = scanner.nextDouble();  
  
 *output*(fileBF, x);  
  
 **double** Eps = *pow*(10, -2);  
 **if** (x >= 1) {  
 ***out***.println(**"Ошибка"** +  
 **"\nВведите x меньше 1"**);  
 }  
 **else** {  
 **while** (Eps >= *pow*(10, -4)) {  
 **int** count = 0;  
 **double** summ = 0;  
 **int** n = 1;  
 **while** ((*abs*(*pow*(*sin*(x), 3) - summ/4)) >= Eps) {  
 summ += *pow*(-1, (n + 1)) \* ((*pow*(3, (2 \* n + 1)) - 3) / *Fact*(2 \* n + 1)) \* *pow*(x, (2 \* n + 1));  
 n++;  
 count++;  
 }  
 *outputResult*(fileBF, x, Eps, count,((1.0 / 4) \* summ));  
 Eps \*= 0.1;  
 }  
 }  
  
 fileBF.close();  
 }  
  
 **public static int** Fact(**int** n) {  
 **int** result = 1;  
 **for** (**int** j = 1; j <= n; j++) {  
 result \*= j;  
 }  
 **return** result;  
 }  
  
 **public static void** output(BufferedWriter fileBF, **double** x) **throws** IOException{  
 DecimalFormat decimalFormat = **new** DecimalFormat(**"0.000000000000"**);  
 String stringOS = **"Функция f(x) = sin(pow(x, 3)) для x = "** + x + **" равняется "** + decimalFormat.format(*function*(x));  
 fileBF.write(stringOS);  
  
 stringOS = **"\n"** + **"Результаты определения значений функции f(x) = pow(-1, (n + 1)) \* ((pow(3, (2 \* n + 1)) - 3) / Fact(2 \* n + 1)) \* pow(x, (2 \* n + 1))"** +  
 **"\nс помощью ряда Маклорена"**;  
 fileBF.write(stringOS);  
  
 stringOS = **"\n"** + **"Где Fact(2 \* n + 1) - цикл по нахождению факториала:"** + **"\n"** +  
 **" int result = 1;\n"** +  
 **" for (int j = 1; j <= n; j++) {\n"** +  
 **" result \*= j;\n"** +  
 **" }\n"** +  
 **" return result;"**;  
 fileBF.write(stringOS);  
  
 stringOS = **"\n"** + **"Значение функции по Маклорену \tПогрешность итерационной процедуры \tПогрешность решения, % \tЧисло итераций"**;  
 fileBF.write(stringOS);  
 }  
  
 **public static void** outputResult(BufferedWriter fileBF, **double** x, **double** Eps, **int** count, **double** result)  
 **throws** IOException {  
 String formatResult = String.*format*(**"%.12f"**, result);  
 String error = String.*format*(**"%.1f"**,*abs*(result - *function*(x)) / *function*(x) \* 100);  
 fileBF.newLine();  
 String stringOS\_Result = **"\t"** + formatResult + **"\t\t\t\t\t\s\s\s"** + Eps +  
 **"\t\t\t\t\s\s\s"** + error + **"\t\t\s\s\s\s\s"** + count;  
 fileBF.write(stringOS\_Result);  
 }  
  
 **public static double** function(**double** x) {  
 **return** *sin*(*pow*(x, 3));  
 }  
}

## 10.3. Тестирование программы с проверкой

Для поверки в MS Excel создана таблица, в которой в ячейку B1 записано значение x, в ячейку B2 значение , в ячейку B3 значение , в ячейки A5:A34 значение i, в ячейки B5:B34 функция для нахождения значения ряда Маклорена, в ячейки C5:C34, в ячейки С5:С34 функция для нахождения суммы ряда Маклорена.

B5)=1/4\*(-1)^(A5+1)\*(3^(2\*A5+1)-3)/ФАКТР(2\*A5+1)\*$B$1^(2\*A5+1)

C5)=B5

C6)=ЕСЛИ(ABS($B$2-C5)>=$B$3;C5+B6;"Otvet vishe")

На рис.1 представлен вид решения в MS Excel.

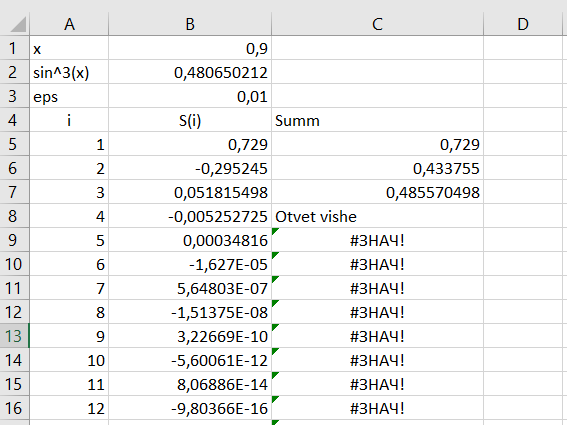


Рисунок 1. Вид решения в MS Excel задания 10

В таблице 10 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 10

Тестирование работы программы и проверка решения задачи 10

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| 2 |  |
| 3 |  |
| № п/п | Решение MS Excel |
| 1 |  |
| 2 |  |
| 3 |  |

Сравнение решения задачи с использованием трех прикладных пакетов показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 11

## 11.1. Общая постановка задачи

Организовать запись данных о поверхности из лабораторной работы по диаграммам из УИР в файл. Файл содержит шапку о содержимом: формула поверхности, интервалы определения по координатам x и y. Запись выполнить в определенном формате с понятными заголовками: массив координат x, массив координат y, матрица значений поверхности (z).

## 11.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.text.DecimalFormat;  
**import** java.util.ArrayList;  
  
**public class** Task1011 {  
  
 **static** ArrayList<Double> *listX* = **new** ArrayList<>();  
 **static** ArrayList<Double> *listY* = **new** ArrayList<>();  
  
 **public static void** main(String[] args) **throws** IOException{  
 File directory = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java"**);  
 File file = **new** File(directory, **"Для 11.txt"**);  
 FileWriter fileWr = **new** FileWriter(file);  
 FileOutputStream fileOS = **new** FileOutputStream(file);  
 BufferedWriter fileBW = **new** BufferedWriter(fileWr);  
  
 String stringOS = **"Формула поверхности: "** + **"Math.cos(Math.sin(x) + y)."** +  
 **"\nИнтервал x: [0,6] с шагом 0,1."** +  
 **"\nИнтервал y: [0,5] с шагом 0,2"**;  
 fileBW.write(stringOS);  
  
 fileBW.newLine();  
 stringOS = **"Массив значений x:"** + **"\n"**;  
 fileBW.write(stringOS);  
 **for** (**double** i = 0; i <= 6; i += 0.1) {  
 *listX*.add(i);  
 stringOS = String.*format*(**"%.1f"**, i) + **"\t"**;  
 fileBW.write(stringOS);  
 }  
  
 fileBW.newLine();  
 stringOS = **"Массив значений y:"** + **"\n"**;  
 fileBW.write(stringOS);  
 **for** (**double** j = 0; j <= 5.2; j += 0.2) {  
 *listY*.add(j);  
 stringOS = String.*format*(**"%.1f"**, j) + **"\t"**;  
 fileBW.write(stringOS);  
 }  
  
 fileBW.newLine();  
 stringOS = **"Матрица значений z:"** + **"\n"**;  
 fileBW.write(stringOS);  
 DecimalFormat decimalFormat = **new** DecimalFormat(**"0.00"**);  
 **double**[][] masZ = **new double**[*listX*.size()][*listY*.size()] ;  
 **for** (**int** i = 0; i < *listX*.size(); i++) {  
 **for** (**int** j = 0; j < *listY*.size(); j++) {  
 masZ[i][j] = *function*(*listX*.get(i), *listY*.get(j));  
 stringOS = decimalFormat.format(masZ[i][j]) + **"\t"**;  
 fileBW.write(stringOS);  
 }  
 fileBW.newLine();  
 }  
  
 fileBW.close();  
 }  
  
 **public static double** function(**double** x, **double** y) {  
 **return** Math.*cos*(Math.*sin*(x) + y);  
 }  
}

## 11.3. Тестирование программы с проверкой

В таблице 11 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 11

Тестирование работы программы и проверка решения задачи 11

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# 12 Задание

## 12.1. Общая постановка задачи

Переделать программу задания 12 лабораторной работы 9 по работе с регулярными выражениями. Дан файл с текстом, применить анализ и редактирование текст файла с использованием ранее написанного алгоритма форматирования текста. Записать отредактированный текст в файл после исходного текста через пустую строку и заголовок.

## 12.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
  
**public class** Task1012 {  
 **public static void** main(String[] args) **throws** IOException {  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 12.txt"**);  
 FileReader fileReader = **new** FileReader(file);  
 FileWriter fileWriter = **new** FileWriter(file, **true**);  
 BufferedReader fileBR = **new** BufferedReader(fileReader);  
 BufferedWriter fileBW = **new** BufferedWriter(fileWriter);  
  
 **int** numberOfStudent = 1;  
 **while** (fileBR.ready()) {  
 String infAboutStudent = fileBR.readLine();  
 *check*(infAboutStudent, fileBW, numberOfStudent);  
 numberOfStudent++;  
 }  
  
 fileBW.close();  
 }  
  
 **public static void** check(String infAboutStudents, BufferedWriter fileBW, **int** numberOfStudent) **throws** IOException{  
 String[][] masPatternAll = {{**"Мужской"**, **"мужской"**, **"мужск"**, **"Мужск"**, **"Муж"**, **"муж"**},  
 {**"Женский"**, **"женский"**, **"женск"**, **"Женск"**, **"Жен"**, **"жен"**}, {**"Город"**, **"город"**, **"Гор"**, **"гор"**},  
 {**"Область"**, **"область"**, **"Обл"**, **"обл"**}, {**"Край"**, **"край"**, **"Кр"**, **"кр"**}, {**"\\.\\.\\."**}};  
 String[] replaceAll = {**"м\\."**, **"ж\\."**, **"г\\."**, **"обл\\."**, **"кр\\."**, **"Заполнить"**};  
 Pattern pattern;  
 Matcher matcher;  
  
 fileBW.newLine();  
 fileBW.write(**"----------------------------------------------------------------------------------"**);  
  
 fileBW.newLine();  
 fileBW.write((numberOfStudent) + **" ученик:"**);  
  
 fileBW.newLine();  
 fileBW.write(infAboutStudents);  
  
 **for** (**int** i = 0; i < masPatternAll.**length**; i++) {  
 **for** ( **int** j = 0; j < masPatternAll[i].**length**; j++) {  
 pattern = Pattern.*compile*(masPatternAll[i][j]);  
 matcher = pattern.matcher(infAboutStudents);  
 **if** (matcher.find()) {  
 infAboutStudents = matcher.replaceAll(replaceAll[i]);  
 **break**;  
 }  
 }  
 }  
 fileBW.newLine();  
 fileBW.write(infAboutStudents);  
 fileBW.newLine();  
 }  
}

## 12.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица данных, содержащая исходные строки и преобразованные.

В таблице 12 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 12

Тестирование работы программы и проверка решения задачи 12

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 13

## 13.1. Общая постановка задачи

В первом файле хранится k матриц размерности m ×n, во втором – k матриц размерности n ×1. Перемножить матрицы из первого и второго файла. Записать третий файл в виде компонент структуры, где каждая компонента содержит три матрицы: первая размерности m ×n из первого файла; вторая размерности n ×1 из второго файла; третья, матрица размерности m ×1, результат произведения. Вывести на экран содержимое всех файлов.

## 13.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
  
**public class** Task1013 {  
  
 **static** ArrayList<String> *matrix\_1String* = **new** ArrayList<>();  
 **static** ArrayList<**int**[]> *matrix\_1Integer* = **new** ArrayList<>();  
 **static** ArrayList<Integer> *matrix\_1QuantityRow* = **new** ArrayList<>();  
 **static** ArrayList<String> *matrix\_2String* = **new** ArrayList<>();  
 **static** ArrayList<**int**[]> *matrix\_2Integer* = **new** ArrayList<>();  
 **static** ArrayList<Integer> *matrix\_2QuantityRow* = **new** ArrayList<>();  
 **static** ArrayList<**int**[]> *matrixResultInteger* = **new** ArrayList<>();  
 **static** ArrayList<String> *matrixResultString* = **new** ArrayList<>();  
  
 **public static void** main(String[] args) **throws** IOException {  
 File file\_1 = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 13\_1.txt"**);  
 FileReader fileReader\_1 = **new** FileReader(file\_1);  
 BufferedReader fileBF\_1 = **new** BufferedReader(fileReader\_1);  
  
 File file\_2 = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Для 13\_2.txt"**);  
 FileReader fileReader\_2 = **new** FileReader(file\_2);  
 BufferedReader fileBR\_2 = **new** BufferedReader(fileReader\_2);  
  
 File directory = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java"**);  
 File fileWithResult = **new** File(directory, **"Для 13\_3.txt"**);  
 FileWriter fileWriter\_Result = **new** FileWriter(fileWithResult);  
 BufferedWriter fileBW\_Result = **new** BufferedWriter(fileWriter\_Result);  
  
 *readMatrices*(fileBF\_1, fileBR\_2, fileBW\_Result);  
 }  
  
 **public static void** readMatrices(BufferedReader fileBR\_1, BufferedReader fileBR\_2, BufferedWriter fileBW\_Result)  
 **throws** IOException{  
 String line;  
 **int** countRowMatrix1 = 0;  
 **while** (fileBR\_1.ready()) { *// считываем матрицы из первого файла* line = fileBR\_1.readLine();  
 **if** (!line.equals(**""**)) { *// вычисление количества строк до предпоследней матрицы* countRowMatrix1++;  
 *matrix\_1String*.add(line);  
 }  
 **else** {  
 *matrix\_1QuantityRow*.add(countRowMatrix1);  
 countRowMatrix1 = 0;  
 }  
  
 }  
  
 **int** countRowMatrix2 = 0;  
 **while** (fileBR\_2.ready()) { *// считываем матрицы из второго файла* line = fileBR\_2.readLine();  
 **if** (!line.equals(**""**)) { *// вычисление количества строк до предпоследней матрицы* countRowMatrix2++;  
 *matrix\_2String*.add(line);  
 }  
 **else** {  
 *matrix\_2QuantityRow*.add(countRowMatrix2);  
 countRowMatrix2 = 0;  
 }  
  
 }  
  
 fileBR\_1.close();  
 fileBR\_2.close();  
 *transferToInteger*(fileBW\_Result);  
 }  
  
 **public static void** transferToInteger(BufferedWriter fileBW\_Result) **throws** IOException {  
 String[] row;  
 **int**[] rowInteger;  
 **int** matrix\_1quantityRowLast;  
 **int** matrix\_1QuantityRowSum = 0;  
 **for** (**int** d : *matrix\_1QuantityRow*) {  
 matrix\_1QuantityRowSum += d;  
 }  
 **for** (String value : *matrix\_1String*) { *// переводим String в int, 1 файл* row = value.split(**"\s"**);  
 rowInteger = **new int**[row.**length**];  
 **for** (**int** j = 0; j < row.**length**; j++) {  
 rowInteger[j] = Integer.*parseInt*(row[j]);  
 }  
 *matrix\_1Integer*.add(rowInteger);  
 }  
 matrix\_1quantityRowLast = *matrix\_1Integer*.size() - matrix\_1QuantityRowSum;  
 *matrix\_1QuantityRow*.add(matrix\_1quantityRowLast); *// добавляем кол-во строк последней матрицы, 1 файл* **int** matrix\_2quantityRowLast;  
 **int** matrix\_2QuantityRowSum = 0;  
 **for** (**int** d : *matrix\_2QuantityRow*) {  
 matrix\_2QuantityRowSum += d;  
 }  
 **for** (String s : *matrix\_2String*) { *// переводим String в int, 2 файл* row = s.split(**"\s"**);  
 rowInteger = **new int**[row.**length**];  
 **for** (**int** j = 0; j < row.**length**; j++) {  
 rowInteger[j] = Integer.*parseInt*(row[j]);  
 }  
 *matrix\_2Integer*.add(rowInteger);  
 }  
 matrix\_2quantityRowLast = *matrix\_2Integer*.size() - matrix\_2QuantityRowSum;  
 *matrix\_2QuantityRow*.add(matrix\_2quantityRowLast); *// добавляем кол-во строк последней матрицы, 2 файл  
  
 multiply*();  
 *output*(fileBW\_Result);  
 }  
  
 **public static void** multiply() {  
 **int** rowLengthMatrix1; *// количество строк матрицы 1 файла* **int** columnLengthMatrix1; *// количество столбцов матрицы 1 файла* **int** rowLengthMatrix2; *// количество строк матрицы 2 файла* **int** columnLengthMatrix2; *// количество столбцов матрицы 2 файла* **for** (**int** c = 0; c < *matrix\_1QuantityRow*.size(); c++) { *// в 1 и 2 лежит одиннаковое кол-во размеров* ArrayList<**int**[]> copyOfMatrix\_1Integer = **new** ArrayList<>(*matrix\_1Integer*);  
 ArrayList<**int**[]> copyOfMatrix\_2Integer = **new** ArrayList<>(*matrix\_2Integer*);  
  
 rowLengthMatrix1 = *matrix\_1QuantityRow*.get(c);  
 columnLengthMatrix1 = *matrix\_1Integer*.get(rowLengthMatrix1 - 1).**length**;  
  
 rowLengthMatrix2 = *matrix\_2QuantityRow*.get(c);  
 columnLengthMatrix2 = *matrix\_2Integer*.get(rowLengthMatrix2 - 1).**length**;  
  
 String[] gettingMatrixRowString = **new** String[rowLengthMatrix1];  
 **int**[][] gettingMatrix = **new int**[rowLengthMatrix1][columnLengthMatrix2];  
 **for** (**int** i = 0; i < rowLengthMatrix1; i++) {*// перемножение матриц из 1 и 2 файлов* **for** (**int** j = 0; j < columnLengthMatrix2; j++) {  
 **for** (**int** k = 0; k < columnLengthMatrix1; k++) {  
 gettingMatrix[i][j] += *matrix\_1Integer*.get(i)[k] \* *matrix\_2Integer*.get(k)[j];  
 }  
 gettingMatrixRowString[i] = String.*valueOf*(gettingMatrix[i][j]);  
 }  
 *matrixResultString*.add(gettingMatrixRowString[i]);  
 *matrixResultInteger*.add(gettingMatrix[i]);  
 }  
  
  
 *matrix\_1Integer*.clear();  
 **for** (**int** u = rowLengthMatrix1; u < copyOfMatrix\_1Integer.size(); u++) {  
 *matrix\_1Integer*.add(copyOfMatrix\_1Integer.get(u));  
 }  
  
 *matrix\_2Integer*.clear();  
 **for** (**int** u = rowLengthMatrix2; u < copyOfMatrix\_2Integer.size(); u++) {  
 *matrix\_2Integer*.add(copyOfMatrix\_2Integer.get(u));  
 }  
 }  
 }  
  
 **public static void** output(BufferedWriter fileBW\_Result) **throws** IOException {  
 ArrayList<String> cloneOfMatrix\_1String = **new** ArrayList<>(*matrix\_1String*);  
 ArrayList<String> cloneOfMatrix\_2String = **new** ArrayList<>(*matrix\_2String*);  
 ArrayList<String> cloneOfMatrixResultString = **new** ArrayList<>(*matrixResultString*);  
 **int** indexOfBeginningMatrix1 = 0;  
 **int** indexOfBeginningMatrix2 = 0;  
  
 **for** (**int** i = 0; i < *matrix\_1QuantityRow*.size(); i++) {  
 String result = **"\t\t\t\t\t\t\tБлок матриц №"** + (i + 1);  
 fileBW\_Result.write(result);  
  
 result = **"Матрица из 1-го файла\t\t\t"** + **"Матрица из 2-го файла\t\t\t"** +  
 **"Матрица, полученная умножением предыдущих"**;  
 fileBW\_Result.newLine();  
 fileBW\_Result.write(result);  
  
 **int** maxQuantityRow;  
 **if** (*matrix\_1QuantityRow*.get(i) > *matrix\_2QuantityRow*.get(i)) {  
 maxQuantityRow = *matrix\_1QuantityRow*.get(i);  
 }  
 **else** {  
 maxQuantityRow = *matrix\_2QuantityRow*.get(i);  
 }  
 *//System.out.println(maxQuantityRow);  
 // добавляем пустые строки, чтобы можно было вывести* **for** (**int** t = 0; t < maxQuantityRow; t++) {  
 **if** (*matrix\_1QuantityRow*.get(i) < t + 1) {  
 cloneOfMatrix\_1String.add(t, **""**);  
  
 }  
 **if** (*matrix\_2QuantityRow*.get(i) < t + 1) {  
 cloneOfMatrix\_2String.add(t, **""**);  
  
 }  
 **if** (*matrix\_1QuantityRow*.get(i) < t + 1) {  
 cloneOfMatrixResultString.add(t, **""**);  
  
 }  
 }  
 String matrix1Output;  
 String matrix2Output;  
 String matrixResultOutput;  
 **for** (**int** l = 0; l < maxQuantityRow; l++) {  
 matrix1Output = cloneOfMatrix\_1String.get(l);  
 matrix2Output = cloneOfMatrix\_2String.get(l);  
 matrixResultOutput = cloneOfMatrixResultString.get(l);  
 result = **"\t"** + matrix1Output + **"\t\t\t\t\t\t"** + matrix2Output + **"\t\t\t\t\t\t"** + matrixResultOutput;  
 fileBW\_Result.newLine();  
 fileBW\_Result.write(result);  
 }  
  
 indexOfBeginningMatrix1 += *matrix\_1QuantityRow*.get(i);  
 indexOfBeginningMatrix2 += *matrix\_2QuantityRow*.get(i);  
  
 cloneOfMatrix\_1String.clear();  
 **for** (**int** u = indexOfBeginningMatrix1; u < *matrix\_1String*.size(); u++) {  
 cloneOfMatrix\_1String.add(*matrix\_1String*.get(u));  
 }  
 cloneOfMatrix\_2String.clear();  
 **for** (**int** u = indexOfBeginningMatrix2; u < *matrix\_2String*.size(); u++) {  
 cloneOfMatrix\_2String.add(*matrix\_2String*.get(u));  
 }  
 cloneOfMatrixResultString.clear();  
 **for** (**int** u = indexOfBeginningMatrix1; u < *matrix\_1String*.size(); u++) {  
 cloneOfMatrixResultString.add(*matrixResultString*.get(u));  
 }  
  
  
 fileBW\_Result.newLine();  
 }  
  
 fileBW\_Result.close();  
 }  
}

## 13.3. Тестирование программы с проверкой

Для тестирования в MS Excel созданы таблицы данных, содержащие элементы двумерного массива.

На рис. 2 и рис. 3 представлены данные 1-го и 2-го файла соответственно.

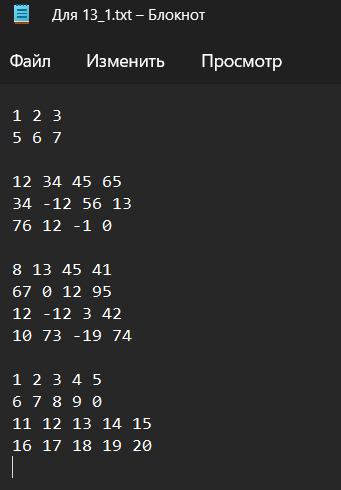


Рисунок 2. Данные 1-го файла

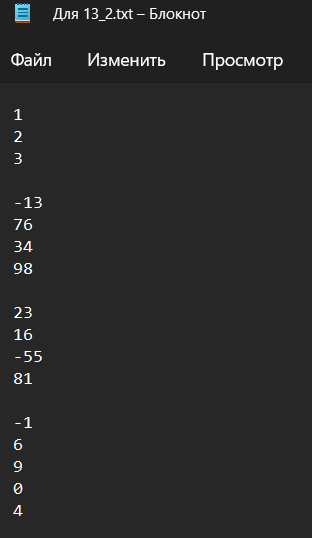


Рисунок 3. Данные 2-го файла

В таблице 13 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 13

Тестирование работы программы и проверка решения задачи 13

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одного прикладного пакета показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задание 14

## 14.1. Общая постановка задачи

Дан файл результатов исследования (листинг натурного эксперимента, результаты численного решения задачи и т.п.). Выполнить анализ данных: 1. Найти максимальное, минимальное и среднее значение параметров по столбцу; 2. Отклонение каждого значения параметра от среднего. Записать данные в конец файла построчно. Сначала ответ на первый вопрос, потом ответ на 2 вопрос.

## 14.2. Решение задачи, код программы

**import** java.io.\*;  
**import** java.util.ArrayList;  
  
**public class** Task1014 {  
  
 **static** ArrayList<String> *data* = **new** ArrayList<>();  
 **static** ArrayList<String> *firstLine* = **new** ArrayList<>();  
 **static** ArrayList<**double**[]> *dataDouble* = **new** ArrayList<>();  
 **static** ArrayList<**double**[]> *dataDoubleDeviation* = **new** ArrayList<>();  
  
 **public static void** main (String[] args) **throws** IOException{  
 File file = **new** File(**"C:\\Users\\pc\\OneDrive\\Документы\\Инфа\\Java\\Лаба 10(7) Java\\Файлы к заданию\\Variant\_5,15.txt"**);  
 FileReader fileReader = **new** FileReader(file);  
 BufferedReader fileBR = **new** BufferedReader(fileReader);  
  
 FileWriter fileWriter = **new** FileWriter(file, **true**);  
 BufferedWriter fileBW = **new** BufferedWriter(fileWriter);  
  
 *readData*(fileBR);  
 *analyzes*(fileBW);  
 }  
  
 **public static void** readData (BufferedReader fileBR) **throws** IOException{  
 String line;  
 **int** count = 0, marker = 0;  
 String[] lineSplit;  
 **while** (fileBR.ready()) {  
 line = fileBR.readLine();  
 **if** (count == 0) {  
 marker = Integer.*parseInt*(line);  
 }  
  
 **if** (count == marker) {  
 lineSplit = line.split(**"\s"**);  
 **for** (**int** i = 0; i < lineSplit.**length**; i++) {  
 *firstLine*.add(lineSplit[i]);  
 }  
 }  
 **if** (count > marker) {  
 lineSplit = line.split(**"\s"**);  
 **double**[] lineInteger = **new double**[lineSplit.**length**];  
 **for** (**int** i = 0; i < lineSplit.**length**; i++) {  
 lineInteger[i] = Double.*parseDouble*(lineSplit[i]);  
 }  
 *data*.add(line);  
 *dataDouble*.add(lineInteger);  
 }  
 count++;  
 }  
  
 fileBR.close();  
 }  
  
 **public static void** analyzes(BufferedWriter fileBW) **throws** IOException{  
 **double**[][] answer = **new double**[3][*dataDouble*.get(0).**length**];  
 **for** (**int** j = 0; j < *dataDouble*.get(0).**length**; j++) { *// max, min, middle value* **double** max = *dataDouble*.get(0)[j], min = *dataDouble*.get(0)[j];  
  
 **for** (**int** i = 0; i < *dataDouble*.size(); i++) {  
 **if** (*dataDouble*.get(i)[j] > max) {  
 max = *dataDouble*.get(i)[j];  
 }  
  
 **if** (*dataDouble*.get(i)[j] < min) {  
 min = *dataDouble*.get(i)[j];  
 }  
  
 answer[2][j] += *dataDouble*.get(i)[j]/ *dataDouble*.size();  
 }  
  
 answer[1][j] = min;  
 answer[0][j] = max;  
 }  
  
 **for** (**int** i = 0; i < *dataDouble*.size(); i++) { *// отклонение от средненего значения* **double** deviation;  
 **double**[] masDeviation = **new double**[*dataDouble*.get(0).**length**];  
  
 **for** (**int** j = 0; j < *dataDouble*.get(0).**length**; j++) {  
 deviation = Math.*abs*(answer[2][j] - *dataDouble*.get(i)[j]);  
 masDeviation[j] = deviation;  
 }  
  
 *dataDoubleDeviation*.add(masDeviation);  
 }  
 *output*(fileBW, answer);  
 }  
  
 **public static void** output(BufferedWriter fileBW, **double**[][] answer) **throws** IOException {  
 fileBW.newLine();  
 String print = **"\nОтвет на первый вопрос:"**;  
 fileBW.write(print);  
 fileBW.newLine();  
  
 **for** (**int** i = 0; i < 3; i++) {  
 print = **""**;  
 **if** (i == 0) {  
 print = **"max\t"**;  
 }  
 **else if** (i == 1) {  
 print = **"min\t"**;  
 }  
 **else** {  
 print = **"middle value\t"**;  
 }  
 fileBW.write(print);  
  
 **for** (**int** j = 0; j < *dataDouble*.get(0).**length**; j++) {  
 print = String.*format*(**"%.4e\t"**, answer[i][j]);  
 fileBW.write(print);  
 }  
  
 fileBW.newLine();  
 }  
  
 fileBW.newLine();  
 print = **"Ответ на второй вопрос:"**;  
 fileBW.write(print);  
 fileBW.newLine();  
  
 **for** (**int** i = 0; i < *dataDoubleDeviation*.size(); i++) {  
 **for** (**int** j = 0; j < *dataDoubleDeviation*.get(0).**length**; j++) {  
 print = String.*format*(**"%.4e\t"**, *dataDoubleDeviation*.get(i)[j]);  
 fileBW.write(print);  
 }  
 fileBW.newLine();  
 }  
  
 fileBW.close();  
 }  
}

## 14.3. Тестирование программы с проверкой

Для тестирования в MS Excel создана таблица с исходными данными

В таблице 14 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением в MS Excel.

Таблица 14

Тестирование работы программы и проверка решения задачи 14

|  |  |
| --- | --- |
| № п/п | Решение Java |
| 1 |  |
| № п/п | Решение MS Excel |
| 1 |  |

Сравнение решения задачи с использованием одним прикладным пакетом показало, что решения задачи в Java и в MS Excel совпадают. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.