

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ  
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

## Отчет по лабораторной работе №2

Специальность ПО

Выполнил  
Войтюк Е.О.  
студент группы ПО-8  
Проверил  
А. А. Крощенко,  
ст. преп. кафедры ИИТ,  
«\_\_k\_\_\_\_\_» 2024 г.

Брест 2024

Цель работы: приобрести базовые навыки работы с файловой системой в Java

**Задание 1. 6) Напишите программу сравнения двух файлов, которая будет печатать первую строку и позицию символа, где они различаются. В противном случае должно выводиться сообщение об эквивалентности содержимого файлов.**

Выполнение:

### Код программы

```
import java.io.*;
import java.nio.file.*;

public class FileComparator {
    public static void main(String[] args) {
        Path filePath1 = Paths.get("file1.txt");
        Path filePath2 = Paths.get("file2.txt");

        try (BufferedReader reader1 = Files.newBufferedReader(filePath1);
            BufferedReader reader2 = Files.newBufferedReader(filePath2)) {

            String line1 = reader1.readLine();
            String line2 = reader2.readLine();
            int lineNum = 1;
            boolean areFilesEquivalent = true;

            while (line1 != null || line2 != null) {
                if (!line1.equals(line2)) {
                    for (int i = 0; i < Math.min(line1.length(), line2.length()); i++) {
                        if (line1.charAt(i) != line2.charAt(i)) {
                            System.out.println("Files differ at line " + lineNum + ", position " + (i+1));
                            System.out.println("Character in file1: " + line1.charAt(i) + ", Character in file2: " + line2.charAt(i));
                            areFilesEquivalent = false;
                        }
                    }
                }
                if (line1.length() != line2.length()) {
                    System.out.println("Files differ at line " + lineNum);
                    areFilesEquivalent = false;
                }
                line1 = reader1.readLine();
                line2 = reader2.readLine();
                lineNum++;
            }
            if (areFilesEquivalent) {
                System.out.println("The contents of the files are equivalent.");
            }
        }
    }
}
```

```
} catch (IOException e) {  
    e.printStackTrace();  
}  
}  
}
```

Наполнение файлов:

File1:

hello world

hii

hello

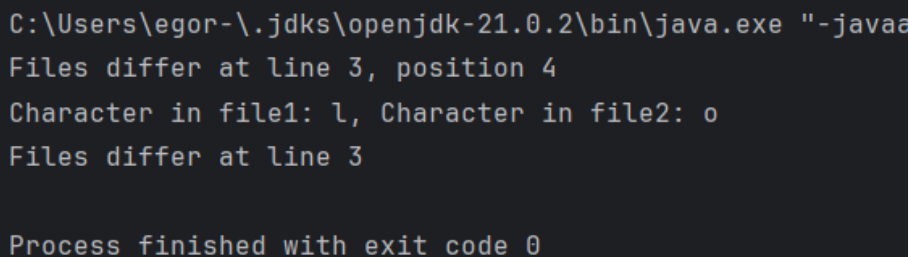
File2:

hello world

hii

helo

### Рисунки с результатами работы программы



```
C:\Users\egor-.jdk\openjdk-21.0.2\bin\java.exe "-javaa  
Files differ at line 3, position 4  
Character in file1: l, Character in file2: o  
Files differ at line 3  
  
Process finished with exit code 0
```

**Задание 2. 6) Утилита split копирует и разбивает файл на отдельные файлы заданной длины. В качестве аргументов ей надо указать имя исходного файла и префикс имен выходных файлов. Если файл не задан или задан как -, программа читает стандартный ввод. По умолчанию размер части разбиения равен 10 строк, а префикс равен x. Имена выходных файлов будут состояться из этого префикса и двух дополнительных букв aa, ab, ac и т. д. (без пробелов и точек между префиксом и буквами). Если префикс имен файлов не задан, то по умолчанию используется x, так что выходные файлы будут называться хаа, хаб и т. д. Формат использования: split [-b | -l] [-d] [входной\_файл [префикс\_выходных\_файлов]] где ключи имеют следующее значение:**

- **-b , --bytes=num** Записывать в каждый выходной файл заданное число **num** байт. При задании числа байт можно использовать суффиксы: **b** означает байты, **k** – 1kb , **m** – 1Mb.

- **-l , --lines=num** Записывать в каждый выходной файл **num** строк.

- **-d , --numericSuffixes** Использовать числовые, а не алфавитные суффиксы, начинающиеся с 00. Суффиксы файлов будут иметь вид: 00, 01, 02 и т. д.

Выполнение:

## Код программы

```
import java.io.*;
import java.nio.file.*;

public class Split {
    private static final int DEFAULT_CHUNK_SIZE = 10;
    private static final String DEFAULT_OUTPUT_PREFIX = "x";
    private static final String DEFAULT_INPUT_FILE = "-";
    private static final boolean DEFAULT_NUMERIC_SUFFIXES = false;

    public static void splitFile(String inputFile, String outputPrefix, int chunkSize, boolean numericSuffixes) throws
IOException {
        try (BufferedReader reader = Files.newBufferedReader(Paths.get(inputFile))) {
            char[] buffer = new char[chunkSize];
            int bytesRead;
            int fileCount = 0;

            BufferedWriter writer = new BufferedWriter(new FileWriter(outputPrefix + getSuffix(fileCount,
numericSuffixes)));

            while ((bytesRead = reader.read(buffer)) != -1) {
                writer.write(buffer, 0, bytesRead);

                if (bytesRead == chunkSize) {
                    writer.close();
                    fileCount++;
                    writer = new BufferedWriter(new FileWriter(outputPrefix + getSuffix(fileCount, numericSuffixes)));
                }
            }

            writer.close();
        }
    }

    private static String getSuffix(int fileCount, boolean numericSuffixes) {
        return numericSuffixes ? String.format("%02d", fileCount) : String.valueOf((char) ('a' + fileCount));
    }
}
```

```

public static void main(String[] args) {
    String inputFile = DEFAULT_INPUT_FILE;
    String outputPrefix = DEFAULT_OUTPUT_PREFIX;
    int chunkSize = DEFAULT_CHUNK_SIZE;
    boolean numericSuffixes = DEFAULT_NUMERIC_SUFFIXES;

    for (int i = 0; i < args.length; i++) {
        if (args[i].startsWith("-l")) {
            chunkSize = Integer.parseInt(args[i].substring(2));
        } else if (args[i].startsWith("-b")) {
            String sizeStr = args[i].substring(2);
            int multiplier = 1;

            if (sizeStr.endsWith("k")) {
                multiplier = 1024;
                sizeStr = sizeStr.substring(0, sizeStr.length() - 1);
            } else if (sizeStr.endsWith("m")) {
                multiplier = 1024 * 1024;
                sizeStr = sizeStr.substring(0, sizeStr.length() - 1);
            }

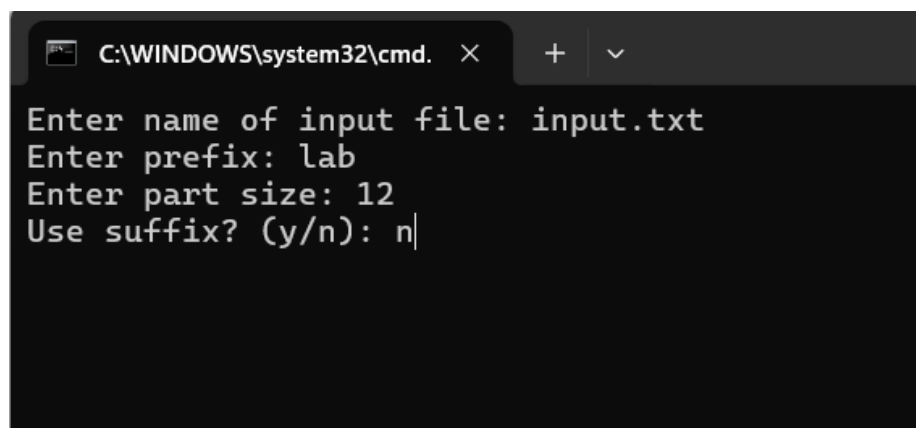
            chunkSize = Integer.parseInt(sizeStr) * multiplier;
        } else if (args[i].equals("-d")) {
            numericSuffixes = true;
        } else if (!args[i].startsWith("-")) {
            if (inputFile.equals(DEFAULT_INPUT_FILE)) {
                inputFile = args[i];
            } else {
                outputPrefix = args[i];
            }
        }
    }










    try {
        splitFile(inputFile, outputPrefix, chunkSize, numericSuffixes);
    } catch (IOException e) {
        System.err.println("Ошибка при разделении файла: " + e.getMessage());
    }
}

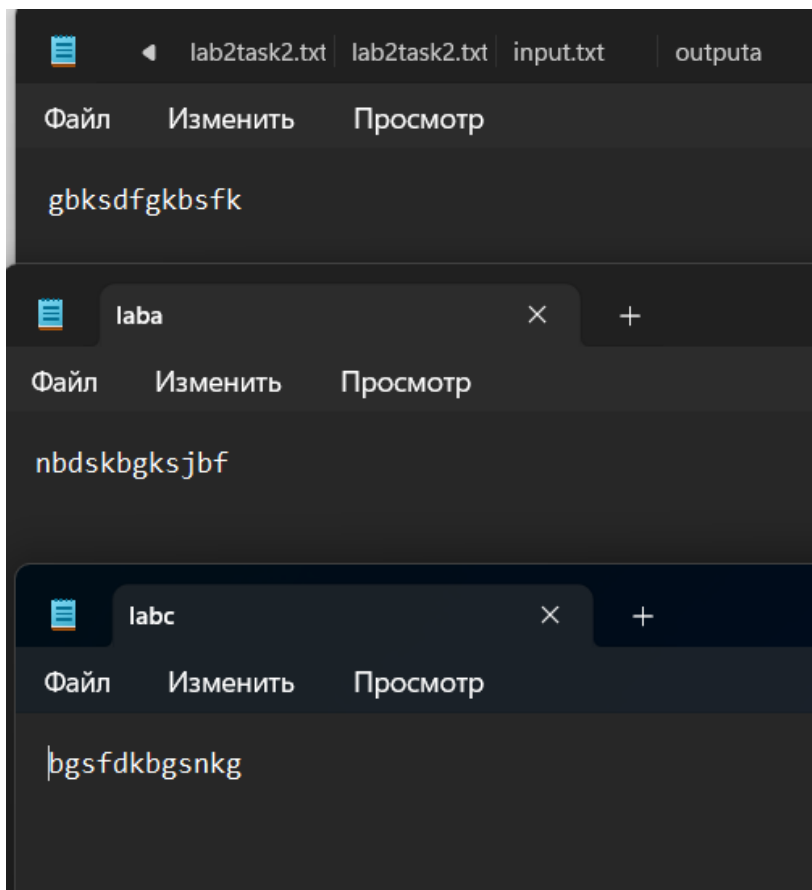
```

**Input.txt:** nbdskbgsjbfgbksdfgkbsfkbgsfdbkbgnskgnskfngskngfsdfgnsfkgnsngf

**Рисунки с результатами работы программы**



 input.txt	05.05.2024 23:33	Текстовый докум...	1 КБ
 lab2task2.bat	05.05.2024 23:52	Пакетный файл ...	1 КБ
 lab2task2.jar	05.05.2024 23:48	Executable Jar File	3 КБ
 laba	06.05.2024 0:12	Файл	1 КБ
 labb	06.05.2024 0:12	Файл	1 КБ
 labc	06.05.2024 0:12	Файл	1 КБ
 labd	06.05.2024 0:12	Файл	1 КБ
 labe	06.05.2024 0:12	Файл	1 КБ
 labf	06.05.2024 0:12	Файл	1 КБ



**Вывод:** приобрел практические навыки работы с файлами в java