Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційні систем та технологій

Лабораторна робота №14

із дисципліни «Основи програмування»

Тема: Потоки вводу-виводу

Виконали:
Студенти групи ІА-24
Призвіще:
Шкарніков Антон,

Кармазіна Анастасія,

Сотніков Олексій.

Перевірив:
Колеснік Валерій Миколайович

Київ - 2022

Хід роботи:

1. Ознайомитись з API класів та інтерфейсів для здійснення операцій вводу-виводу. Особливу увагу звернути на такі класи та інтерфейси:

-InputStream

- FileInputStream

- OutputStream

- FileOutputStream

- Reader

- FileReader

- Writer

- FileWriter

- AutoCloseable

- Closable

- IOException

2. Виконати завдання з таблиці 2 відповідно до свого варіанту у таблиці 1.

- Кожне завдання має бути реалізовано як окремий клас.

- Кожен клас має складатись щонайменше з двох методів:

- public static void main(String[] agrs) - точка входу.

- Метод, що реалізує задане завдання. Метод має перевіряти аргументи та у разі їх помилковості аварійно закінчувати свою роботу шляхом викидання стандартного виключення IllegalArgumentException або NullPointerException. В разі неможливості виконання операції, метод повинен викидати IOException або FileNotFoundException. В жодному разі цей метод не повинен напряму взаємодіяти з користувачем через консоль або інший UI (ніколи не змішуйте бізнес-логіку та користувацький інтерфейс).

- Клас може містити інші допоміжні методи.

- При виконанні завдань слід звернути увагу на ефективність з точки зору швидкодії. При виконанні завдань 1-6 слід використовувати клас BufferedReader та BufferedWriter, а при 7-11 – ні в якому разі не намагатись обробляти усі байти по одному, а використовувати методи read(byte[] b) та write(byte[] b), які працюють з масивами.

3. Відповісти на контрольні питання

Таблиця 1. Варіанти завдань

Варіант	Завдання № 1	Завдання № 2
1	1	7
2	2	8
3	3	9
4	4	10
5	5	11
6	6	7
7	1	8
8	2	9
9	3	10
10	4	11
11	5	7

4	void copyToUpperCase(String source, String destination) Скопіювати текстовий файл <*source*> у <*destination*>, замінивши в процесі усі символи у верхній регістр.
10	void split(String source, String destinationPrefix, long maxSize) Розбити файл <source> на кілька файлів (1 та більше) з назвами <*destinationPrefix*>+".000", <*destinationPrefix*>+".001", <*destinationPrefix*>+".002", ..., кожен з яких буде мати розмір, не більший ніж <*maxSize*>.

Клас task1

```
import java.io.*;

public class Task1 {
    public static void main(String[] args) {
        //Test with legal arguments
        System.out.print("Test 1: ");
        printResult("IO/task1/input.txt", "IO/task1/output.txt");

        //Test with null as argument
        System.out.print("Test 2: ");
        printResult(null, null);

        //Test with wrong path
        System.out.print("Test 3: ");
        printResult("smth.txt", "somewhere.txt");
    }

    private static void copyToUpperCase(String source, String destination)
throws IOException {
        if (source != null && destination != null) {
            try (FileReader fileReader = new FileReader(source);
                 BufferedReader bufferedFileReader = new
BufferedReader(fileReader);
                 FileWriter fileWriter = new FileWriter(destination);
                 BufferedWriter bufferedFileWriter = new
BufferedWriter(fileWriter)) {

                String line;
```

```
                while ((line = bufferedFileReader.readLine()) != null) {
                    bufferedFileWriter.write(line.toUpperCase());
                    bufferedFileWriter.newLine();
                }
            }
        } else {
            throw new IllegalArgumentException("Empty argument!");
        }
    }

    private static void printResult(String source, String destination) {
        try {
            copyToUpperCase(source, destination);
            System.out.println("Done successfully!");
        } catch (FileNotFoundException f404Ex) {
            System.out.println("Wrong path or such files doesn't exist!");
        } catch (IOException ioEx) {
            System.out.println("Something happened while working with
file!");
        } catch (IllegalArgumentException iae) {
            System.out.println(iae.getMessage());
        }
    }
}
```

Клас Task2

```
import java.io.*;

public class Task2 {
    public static void main(String[] args) {
        //Test with legal arguments
        System.out.print("Test 1: ");
        printResult("IO/task2/input.txt", "IO/task2/parts", 512);

        //Test with null as argument
        System.out.print("Test 2: ");
        printResult(null, null, 0);

        //Test with wrong path
        System.out.print("Test 3: ");
        printResult("smth.txt", "somewhere.txt", 200);
    }

    private static void split(String source, String destinationPrefix, long
maxSize) throws IOException {

        if (source == null || destinationPrefix == null || maxSize <= 0) {
            throw new IllegalArgumentException("IllegalArgument!");
        }

        final int DEFAULT_SIZE = 4194304;
        byte[] bytes = new byte[maxSize < DEFAULT_SIZE ? (int) maxSize :
DEFAULT_SIZE];
        int i = 0;

        try (FileInputStream fileInputStream = new FileInputStream(source);
             BufferedInputStream bufferedFileInputStream = new
BufferedInputStream(fileInputStream)) {

            flag:
            while (true) {
```

```
                int written = 0;
                String destination = destinationPrefix +
String.format(".%03d", i);

                try (FileOutputStream fileOutputStream = new
FileOutputStream(destination);
                     BufferedOutputStream bufferedFileOutputStream = new
BufferedOutputStream(fileOutputStream)) {

                    while (written < maxSize) {
                        int read;
                        if ((read = bufferedFileInputStream.read(bytes)) == -
1) {
                            break flag;
                        }
                        bufferedFileOutputStream.write(bytes, 0, read);
                        written += read;
                    }
                }
                i++;
            }
        }
    }

    private static void printResult(String source, String destinationPrefix,
long maxSize) {
        try {
            split(source, destinationPrefix, maxSize);
            System.out.println("Done successfully!");
        } catch (FileNotFoundException f404Ex) {
            System.out.println("Wrong path or such files doesn't exist!");
        } catch (IOException ioEx) {
            System.out.println("Something happened while working with
file!");
        } catch (IllegalArgumentException iae) {
            System.out.println(iae.getMessage());
        }
    }
}
```

Результат виконання програми:

```
Test 1: Done successfully!
Test 2: Empty argument!
Test 3: Wrong path or such files doesn't exist!
```

```
Test 1: Done successfully!
Test 2: Empty argument!
Test 3: Wrong path or such files doesn't exist!
```

Висновок: у даній лабораторній роботі ми ознайомилися з API класів та інтерфейсів для здійснення операцій вводу-виводу, а також попрактикувалися

в їх застосуванні. Ознайомилися з класами InputStream, OutputStream, Reader, Writer, а також інтерфейсами Autoclosable, Closable та IOExcepttion.