

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
Кафедра систем штучного інтелекту



Звіт до лабораторної 8
з дисципліни
“ ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ ”

Виконала:
студентка групи КН-108
інституту ІКНІ
Бокшо Каріна
Викладач:
Гасько Р.Т.

Львів-2018

Лабораторна робота №8
Утилітарні класи Java SE. Обробка масивів і рядків. Інтерактивні
консольні програми для платформи

Мета роботи:

- Розробка власних утилітарних класів.
- Набуття навичок вирішення прикладних задач з використанням масивів і рядків.
- Реалізація діалогового режиму роботи з користувачем в консольних програмах мовою Java.

1. Індивідуальне завдання (1 варіант):

Ввести текст. Визначити та вивести, яких літер (голосних чи приголосних) більше в кожному реченні тексту. Результат вивести у вигляді таблиці.

1.1 Виконала:

Студентка Бокшо К. Е.; КН-108;

1.2 Рекомендації / вимоги до лабораторної роботи:

Вимоги

1. Розробити та продемонструвати консольну програму мовою *Java* в середовищі *Eclipse* для вирішення прикладної задачі за номером, що відповідає номеру студента в журналі групи з поверненням до початку. Наприклад **1 ->1, 2->2, ..., 15->15, 16->1, 17->2, ..., 30->15, 31-1** і т.д.
2. Використовуючи програму рішення завдання відповідно до **прикладної задачі** забезпечити обробку команд користувача у вигляді **текстового меню** :
 - а. введення даних;
 - б. перегляд даних;
 - с. виконання обчислень;
 - д. відображення результату;
 - е. завершення програми і т.д.
3. Забезпечити обробку параметрів командного рядка для визначення режиму роботи програми:
 - а. параметр “-h” чи “-help”: відображається інформація про автора програми, призначення (індивідуальне завдання), детальний опис режимів роботи (пунктів меню та параметрів командного рядка);
 - б. параметр “-d” чи “-debug”: в процесі роботи програми відображаються додаткові дані, що полегшують налагодження та перевірку працездатності програми: діагностичні повідомлення, проміжні значення змінних, значення тимчасових змінних та ін.

4. При вирішенні прикладних задач використовувати латинку .
5. Продемонструвати використання об'єктів класу `StringBuilder` або `StringBuffer`
6. Застосувати функціональну (процедурну) декомпозицію - розробити власні утилітарні класи (особливий випадок допоміжного класу, див. та для обробки даних використовувати відповідні статичні методи.
7. Забороняється використовувати засоби обробки регулярних виразів: класи пакету `java.util.regex` (`Pattern` , `Matcher` та ін.), а також відповідні методи класу `String` (`matches` , `replace` , `replaceFirst` , `replaceAll` , `split`)

2. Розробка програми

2.1 Ієрархія та структура класів

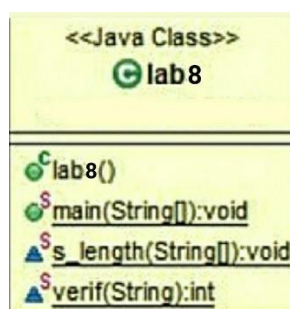
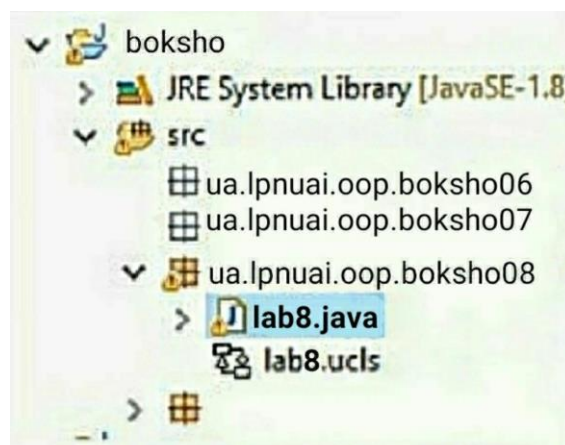


Рисунок 1 « Ієрархія та структура класів»

Структура розробленого проекту:



2.2 Опис програми

Програма реалізована у вигляді консольного вікна з послідовним виконанням завдання.

Основне призначення: використовуючи введений текст, визначає та виводить у вигляді таблиці, яких літер (голосних чи приголосних) більше в кожному реченні тексту.

Програма працює лише з текстом написаним на латинкою. Для обробки даних використовуються класи-утиліти. Регулярних вирази не використовуються при виконанні завдання.

2.3 Важливі фрагменти програми

```
/**
 * class TextHelper Утилітарний клас, що опрацьовує текст (розбиває його на
 * речення).
 *
 *
 */
class TextHelper {
    private static final char DOT = '.'; // Крапка
    private static final char EXCLAMATION = '!'; // Знак оклику
    private static final char QUESTION = '?'; // Знак питання

    /* Розбиває отриманий текст на речення */
    public static ArrayList<String> getSentences(String text) {

        /* Список, що зберігає результат */
        ArrayList<String> result = new ArrayList<String>();
        String temp = ""; // Буфер
        for (int i = 0; i < text.length(); ++i) {
            char sign = text.charAt(i);
            if (sign == DOT || sign == EXCLAMATION || sign == QUESTION) {
                result.add(temp);
                temp = "";
            } else {
                temp += text.charAt(i);
            }
        }
        return result;
    }
}

/**
 * class StringHelper Утилітарний клас, що виконує пошук та підрахунок голосних
 * та приголосних у реченні.
 *
 *
 */
class StringHelper {
    /* Перелік голосних */
    private static final String VOWELS = "aeiouyAEIOUY";
    /* Перелік приголосних */
    private static final String CONSONANTS = "bcdfghjklmnpqrstvwxyz" +
    "BCDFGHJKLMNPQRSTVWXYZ";

    /* Перевіряє чи є символ голосною буквою */
    public static boolean isVowel(char ch) {

        return VOWELS.indexOf(ch) >= 0;
    }

    /* Перевіряє чи є символ приголосною буквою */
    public static boolean isConsonants(char ch) {

        return CONSONANTS.indexOf(ch) >= 0;
    }

    /* Підраховує голосні */
    public static int countVowel(String sentence) {
```

```

        int counter = 0;
        for (int i = 0; i < sentence.length(); ++i) {
            if (isVowel(sentence.charAt(i))) {
                ++counter;
            }
        }
        return counter;
    }

    /* Підраховує приголосні */
    public static int countConsonants(String sentence) {

        int counter = 0;
        for (int i = 0; i < sentence.length(); ++i) {
            if (isConsonants(sentence.charAt(i))) {
                ++counter;
            }
        }
        return counter;
    }
}

/**
 * class CountHelper Утилітарний клас, що заповнює список даними для подальшого
 * опрацювання.
 *
 */
class CountHelper {
    /* Заносить кількість голосних та приголосних до списку */
    public static ArrayList<Integer> Count(String text) {
        ArrayList<Integer> result = new ArrayList<Integer>();
        ArrayList<String> sentences = TextHelper.getSentences(text);
        for (int i = 0; i < sentences.size(); i++) {
            /* Кількість голосних */
            int vowels = StringHelper.countVowel(sentences.get(i));
            /* Кількість приголосних */
            int consonants =
StringHelper.countConsonants(sentences.get(i));
            result.add(vowels);
            result.add(consonants);
        }
        return result;
    }
}

/**
 * class ChartHelper Утилітарний клас, що виконує виведення результатів.
 *
 */
class ChartHelper {
    /* Виводить дані у вигляді таблиці */
    public static void printChart(String text) {
        ArrayList<Integer> data = CountHelper.Count(text);
        int counter = 0;
        System.out.println("-----"
                                + "-----"
                                + "\n");
        System.out.format("        Речення №      Голосних      Приголосних\n\n");
        for (int i = 0; i < data.size(); i += 2) {
            counter++;
            System.out.format("        %d          %d          %d\n", counter,
data.get(i), data.get(i + 1));
        }
        System.out.println("\n-----")

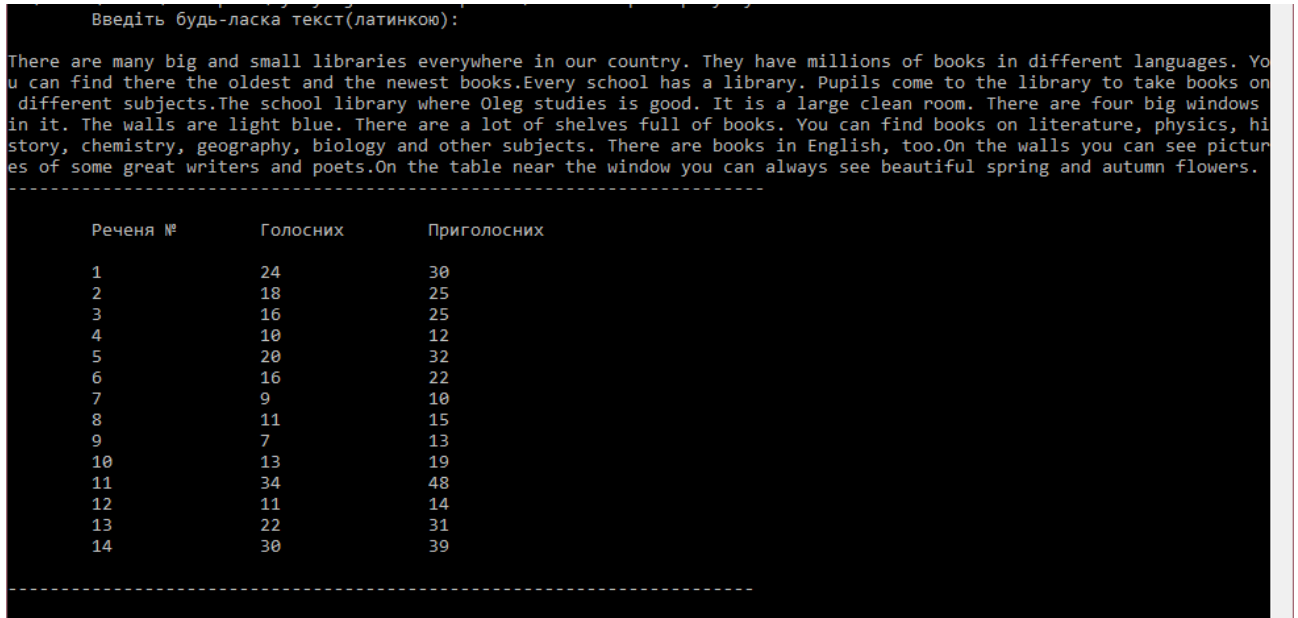
```

```
-----\n");  
}
```

+ "-----"

3. РЕЗУЛЬТАТ РОБОТИ

Для налагодження роботи програми було успішно проведено її тестування.



Речення №	Голосних	Приголосних
1	24	30
2	18	25
3	16	25
4	10	12
5	20	32
6	16	22
7	9	10
8	11	15
9	7	13
10	13	19
11	34	48
12	11	14
13	22	31
14	30	39

Рисунок 2 "Результат роботи програми"

ВИСНОВКИ

Створено і налагоджено програму, що повністю виконую поставлене індивідуальне завдання та відповідає вимогам.

Було отримано і вдосконалено навички у розробці власних утилітарних класів та у вирішенні прикладних задач з використанням масивів і рядків