Лабораторна робота №4. Інтерактивні консольні програми для платформи Java SE

Мета: Реалізація діалогового режиму роботи з користувачем в консольних програмах мовою Java.

1 ВИМОГИ

1. Використовуючи програму рішення завдання лабораторної роботи №3, відповідно до прикладної задачі забезпечити обробку команд користувача у вигляді текстового меню:
   * введення даних;
   * перегляд даних;
   * виконання обчислень;
   * відображення результату;
   * завершення програми і т.д.
2. Забезпечити обробку параметрів командного рядка для визначення режиму роботи програми:
   * параметр "-h" чи "-help": відображається інформація про автора програми, призначення (індивідуальне завдання), детальний опис режимів роботи (пунктів меню та параметрів командного рядка);
   * 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.
   1. Розробник

* Абдулаєв Ібрагім Заурбекович
* Група КІТ-119В
* Варіант 1

1. ОПИС ПРОГРАМИ

2.1 Опис класів

* **ua.khpi.oop.abdulaev04.Helper**



Рис. 2.1 – Опис класу Helpers

* **ua.khpi.oop.abdulaev04.Main**



Рис. 2.2 – Опис класу **Main**

2.2 Текст програми

**Main.java**

package ua.khpi.oop.abdulaev04;

import java.io.IOException;

import java.util.Scanner;

public class Main {

public static boolean exitFlag = false;

/\*\*

\* Очікування клавіші Enter

\*/

public static void waitForEnter() {

try {

System.in.read();

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Очищення вікна консолі

\*/

public static void clearConsole() {

System.out.print("\033[H\033[2J");

System.out.flush();

}

/\*\*

\* Обробка аргументів командного рядка

\* @param args масив аргументів командного рядка

\*/

public static void commandArguments(String[] args) {

for (String arg : args) {

switch (arg) {

case "-debug":

case "-d":

Helper.DEBUG = true;

break;

case "-help":

case "-h":

exitFlag = true;

System.out.println(

"Аргументи командного рядка: \n"

+ "\t-d[-debug] діагностичні повідомлення, проміжні значення змінних\n"

+ "\t-h[-help] інформація про автора програми, призначення, опис режимів роботи\n"

+ "Програмне меню:\n"

+ "\t0 - Вихід\n"

+ "\t1 - Введення даних\n"

+ "\t2 - Перегляд даних\n"

+ "\t3 - Виконання обчислень\n"

+ "\t4 - Відображення результату\n"

+ "Автор:\n"

+ "\tАбдулаєв Ібрагім студент гр. КІТ-119в\n"

);

break;

}

}

}

/\*\*

\* Циклічне відображення програмного меню

\*/

public static void appMenu() {

Scanner input = new Scanner(System.in);

int option = 0;

int inputLength = 0;

String inputData = null;

String inputReplacer = null;

String outputData = null;

while (true) {

clearConsole();

System.out.println(

"\t0 - Вихід\n"

+ "\t1 - Введення даних\n"

+ "\t2 - Перегляд даних\n"

+ "\t3 - Виконання обчислень\n"

+ "\t4 - Відображення результату"

);

option = input.nextInt();

switch (option) {

case 0:

return;

case 1:

input.nextLine();

System.out.println("Введіть текст для обробки:");

inputData = input.nextLine();

input.nextLine();

break;

case 2:

if (inputData != null && !inputData.isEmpty())

System.out.printf("Дані: %s\n", inputData);

else

System.out.println("Введено невірні значення. Перевірте вхідні дані");

waitForEnter();

break;

case 3:

if (inputData != null && !inputData.isEmpty())

outputData = Helper.stringsInfo(inputData.split(" "));

else

System.out.println("Введено невірні значення. Перевірте вхідні дані");

waitForEnter();

break;

case 4:

if (outputData != null)

System.out.println(outputData);

else

System.out.println("Обчислення не виконувались");

waitForEnter();

break;

}

}

}

public static void main(String[] args) {

commandArguments(args);

if (exitFlag) return;

appMenu();

}

}

**Helpers.java**

package ua.khpi.oop.abdulaev04;

/\*\*

\* Уилітарний клас Helper

\*/

public class Helper {

public static boolean DEBUG = false;

/\*\*

\* Розбити рядки на три групи:

\* - починається з голосної;

\* - починається з приголосної;

\* - починається не з букви.

\* Знайти найкоротший рядок в кожній групі. Вивести цей рядок та йогої довжину.

\* @param arr масив рядків

\*/

public static String stringsInfo(String[] arr) {

final char[][] letterRange = {{65, 90}, {97, 122}};

final char[] vowels = {65, 69, 73, 79, 85, 89, 97, 101, 105, 111, 117, 121};

StringBuilder vowelGroup, consonantGroup, notLetterGroup, output;

String[] minStrings = new String[3];

/// [0] -> notLetter, [1] -> vowel, [2] -> consonant

vowelGroup = new StringBuilder();

consonantGroup = new StringBuilder();

notLetterGroup = new StringBuilder();

output = new StringBuilder();

for (String str : arr) {

if (str.isEmpty()) continue;

char ch = str.charAt(0);

boolean isLetter = false;

boolean isVowel = false;

if (DEBUG) System.out.printf("[%s, %c]\n", str, ch);

for (char[] lr : letterRange) {

if (ch >= lr[0] && ch <= lr[1]) {

isLetter = true;

break;

}

}

if (!isLetter) {

if (minStrings[0] == null || str.length() < minStrings[0].length())

minStrings[0] = str;

if (DEBUG) System.out.printf("isLetter -> %b\n", false);

notLetterGroup.append(str).append(" ");

continue;

}

if (DEBUG) System.out.printf("isLetter -> %b\n", true);

for (char v : vowels) {

if (ch == v) {

isVowel = true;

break;

}

}

if (isVowel) {

if (minStrings[1] == null || str.length() < minStrings[1].length())

minStrings[1] = str;

vowelGroup.append(str).append(" ");

}

else {

if (minStrings[2] == null || str.length() < minStrings[2].length())

minStrings[2] = str;

consonantGroup.append(str).append(" ");

}

System.out.printf("isVowel -> %b\n", isVowel);

System.out.printf("isConsonant -> %b\n", !isVowel);

if (DEBUG) {

System.out.println("Groups ->");

System.out.printf("\tVowel: '%s'\n", vowelGroup.toString());

System.out.printf("\tConsonant: '%s'\n", consonantGroup.toString());

System.out.printf("\tNotLetter: '%s'\n", notLetterGroup.toString());

}

}

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

if (minStrings[i] == null)

minStrings[i] = "";

output

.append(String.format("1) Починається з голосної: -> \n\t %s\n", vowelGroup.toString()))

.append(String.format("Найкоротший рядок та його довжина -> \n\t['%s', %d]\n", minStrings[1], minStrings[1].length()))

.append(String.format("2) Починається з приголосної: -> \n\t %s\n", consonantGroup.toString()))

.append(String.format("Найкоротший рядок та його довжина -> \n\t['%s', %d]\n", minStrings[2], minStrings[2].length()))

.append(String.format("3) Починається не з букви: -> \n\t %sa\n", notLetterGroup.toString()))

.append(String.format("Найкоротший рядок та його довжина -> \n\t['%s', %d]\n", minStrings[0], minStrings[0].length()));

return output.toString();

}

}

3 ВИСНОВКИ

На лабораторній роботі навчилися реалізувати діалоговий режим роботи з користувачем в консольних програмах мовою Java.

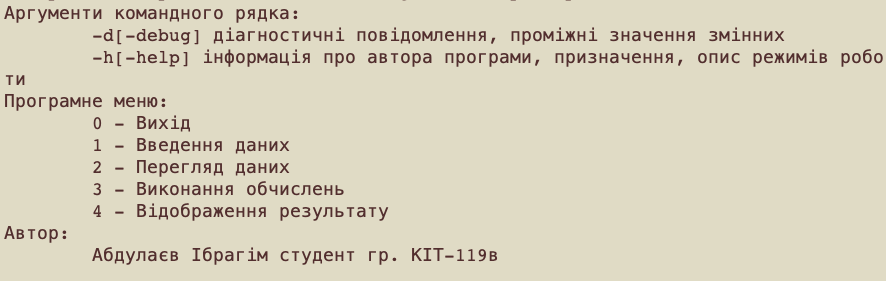


Рис. 3.1 – Результат роботи програми з параметром -h[-help]

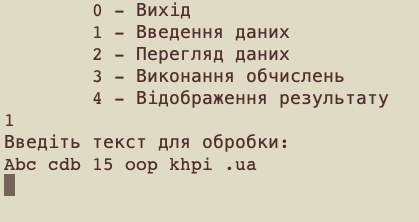


Рис. 3.2 – Демонстрація програмного меню

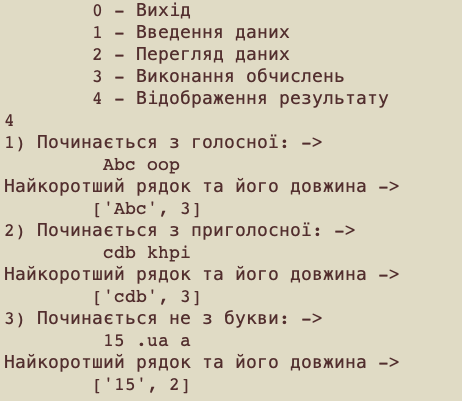


Рис. 3.3 – Результат роботи програми

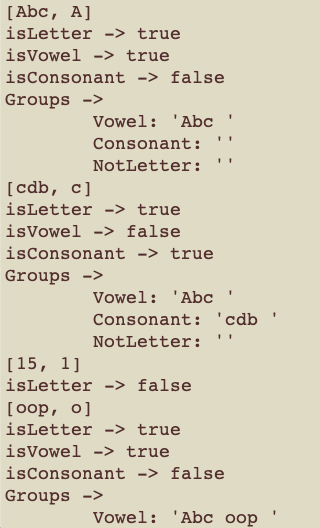


Рис. 3.4 – Результат роботи програми з параметром -d[-debug] (Виведення додаткової інформації про перебір символів тексту, про поточне слово, першу літеру, голосна чи приголосна, поточний стан змінної кожної групи)