

Лабораторна робота №4

ІНТЕРАКТИВНІ КОНСОЛЬНІ ПРОГРАМИ ДЛЯ ПЛАТФОРМИ JAVA SE

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

ВИМОГИ

Розробник:

- Чугунов Вадим Юрійович;
- КІТ-119а;
- Варіант №24.

Загальне завдання:

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.

ОПИС ПРОГРАМИ

Опис змінних:

```
StringBuilder key;                                //змінна для обробки команд

Scanner in;                                       // змінна для введення даних

String[] words;                                  // масив підрядків який буде заповнений
результатом Split методу

HashMap<String, Integer> wordToCount //наша хеш таблиця для виводу слова та
його поторів в тексті

boolean check[];                                //змінна, яка відповідає за параметри які ми
передаємо програмі
```

Ієрархія та структура класів:

class Main – точка входу в програму.

class Helper – клас, який реалізує методи для виконання індивідуального завдання, діалогове меню з користувачем та обробку параметрів які задаються під час запуску програми.

ТЕКСТ ПРОГРАМИ

Текст файлу **Main**:

```
package ua.oop.khpi.chugunov04;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

public class Main {

    /**
     * An entry point of program.
     * @param args - parameters of command line
     *
     */

    public static void main(String[] args) {

        /**
         * Using a class StringBuilder to command processing.
         */

        final StringBuilder key = new StringBuilder();

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

            switch (args[i]) {
```

```

// Overall info commands

case "-help":

    Helper.InfoHelp();

    break;


case "-debug":

    Helper.DebugHelp(args);

    break;


// Text commands

case "-text":

    for (int it = 1; it < args.length - 1; it++)

        key.append(args[it] + " ");

    break;


case "-showtext":

    Helper.PrintText(key.toString());

    break;


case "-manip":

    String[] words = Helper.SplitString(key.toString());

    Helper.PrintResult(words);

    break;


default:

    break;

}

```

```

        }

    }

}

/**
 * A utilitarian class for manipulating lines.
 * Contains methods for text output
 * and methods for output the number
 * of repetitions of words in the text
 *
 * @version 1.0 25 Nov 2020
 * @author chugunov-vadim
 */
class Helper {

    static void PrintText(final String line) {

        /**
         * Using a class  StringBuffer
         * for changing input string.
         */

        StringBuilder str = new StringBuilder();

        for (char symbol : line.toCharArray()) {

            str.append(symbol);

        }

        System.out.println(str.toString());
    }
}

```

```

    }

    public static String[] SplitString(String text) {

        List<String> words = new ArrayList<>();

        StringBuilder builder = new StringBuilder();

        for(char symbol : text.toCharArray()) {

            if((int)symbol == 32 | (int)symbol == 33 | (int)symbol ==
58| (int)symbol == 44| (int)symbol == 46) {

                words.add(builder.toString());

                builder = new StringBuilder();

                continue;

            }

            builder.append(symbol);

        }

        if(builder.length() != 0) {

            words.add(builder.toString());

        }

        for (int i = 0; i < words.size(); i++) {

            if(words.get(i).length() == 0) {

                words.remove(i);

            }

        }

        String[] output = new String[words.size()];

        for (int i = 0; i < words.size(); i++) {

            output[i] = words.get(i);

        }

        return output;

    }

```

```

public static void PrintResult (String[]words){

    System.out.println( "=====");

    System.out.println( "A Word" + "\t\t\t" + "Count");

    System.out.println( "=====");

    HashMap<String, Integer> wordToCount = new HashMap<>();

    for (String word : words) {

        if (!wordToCount.containsKey(word)) {

            wordToCount.put(word, 0);

        }

        wordToCount.put(word, wordToCount.get(word) + 1);

    }

    for (String word : wordToCount.keySet()) {

        System.out.println(word + "\t\t\t" + wordToCount.get(word));

    }

    System.out.println( "=====");

}

/**
 * Method for displaying information
 * about the author of the program,
 * appointment (individual task),
 * detailed description of operating modes
 * (menu items and command line options);

```

```

*/

static void InfoHelp() {

    System.out.println("\n---HELP OPTION LAUNCHED---\n");

    System.out.println("Author :\tChugunov Vadim");

    System.out.println("Program task :\tEnter text."
        + " Find and display how many times each word is repeated in the
text."

        + "\n\t\tDisplay the result as a table.");

    System.out.println("\n---MENU OPTIONS---\n");

    System.out.println("1. Enter text -> -text <your text> ");
    System.out.println("2. Show entered text -> -showtext ");
    System.out.println("3. Manipulate entered text -> -manip ");
    System.out.println("4. Exit program -> -exit");

}

/**
 * Method for outputting information
 * about the values of intermediate variables,
 * about diagnostic messages,
 * about performing operations on input data.
 * @param cmdArgs -
 * specified command line parameters with input data
 */

static void DebugHelp(String[] cmdArgs) {

```



```
System.out.println("---DEBUG INFO---");

boolean check[] = {false, false, false};

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

    if(cmdArgs[i].contains("-text"))

        check[0] = true;

    if(cmdArgs[i].contains("-showtext"))

        check[1] = true;

    if(cmdArgs[i].contains("-manip"))

        check[2] = true;
}

if(check[0])

    System.out.println("Text has been already entered!");
else

    System.out.println("Text hasn't been entered yet!");

if(check[1])

    System.out.println("Your text has been already shown!");
else

    System.out.println("Your text hasn't been shown yet!");
```

```
        if(check[2])

            System.out.println("Your text has been already processed!");

        else

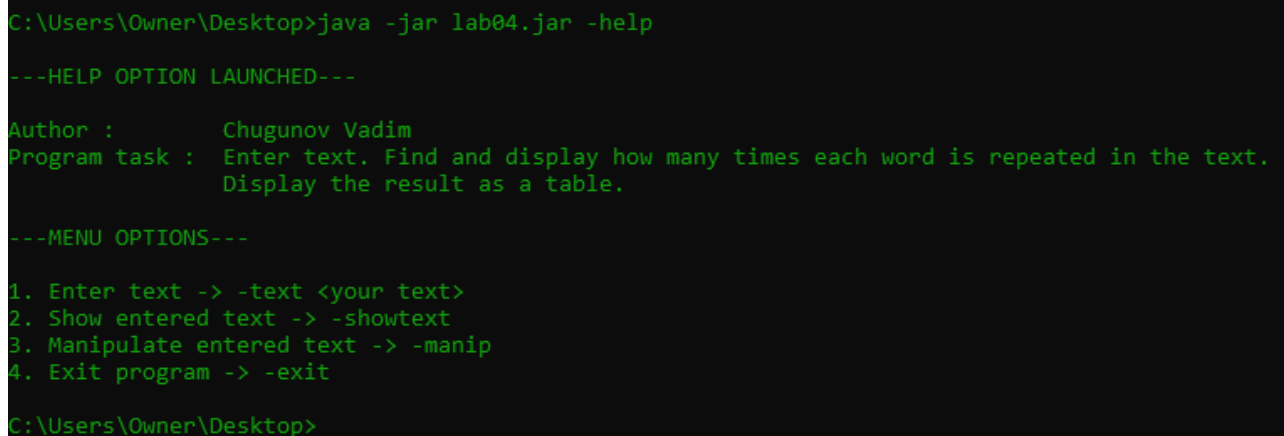
            System.out.println("Your text hasn't been processed yet!");

    }

}
```

РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ

На рисунку 4.1 зображено результат роботи програми — виклик програми через консоль, введення параметрів **-h**, **-help** (рис. 4.1а), **-debug** (рис. 4.1в) консольного рядка та команд для виконання певних дій над вхідними даними (рис. 4.1б).



```
C:\Users\Owner\Desktop>java -jar lab04.jar -help

---HELP OPTION LAUNCHED---
```

Author : Chugunov Vadim

Program task : Enter text. Find and display how many times each word is repeated in the text.
Display the result as a table.

---MENU OPTIONS---

1. Enter text -> -text <your text>
2. Show entered text -> -showtext
3. Manipulate entered text -> -manip
4. Exit program -> -exit

C:\Users\Owner\Desktop>

a)

```

---MENU OPTIONS---

1. Enter text -> -text <your text>
2. Show entered text -> -showtext
3. Manipulate entered text -> -manip
4. Exit program -> -exit

C:\Users\Owner\Desktop>java -jar lab04.jar -text Hello Hello Vadim Vadim Hi -manip
=====
A Word                      Count
=====
Hi                          1
Hello                       2
Vadim                       2
=====

```

б)

```

C:\Users\Owner\Desktop>java -jar lab04.jar -debug
---DEBUG INFO---
Text hasn't been entered yet!
Your text hasn't been shown yet!
Your text hasn't been processed yet!

C:\Users\Owner\Desktop>java -jar lab04.jar -text How are you -debug
---DEBUG INFO---
Text has been already entered!
Your text hasn't been shown yet!
Your text hasn't been processed yet!

```

в)

Рисунок 4.1 – Результат виконання завдання

ВАРІАНТИ ВИКОРИСТАННЯ

Програму можна використовувати для пошуку кількості повторів у тексті потрібного нам слова. Програму можна запускати у двох режимах: Debug (додається допоміжна інформація для користувача) та Release. Програма реалізує в собі роботу у режимі Help, який дає певну інформацію про автора програми, та забезпечує користувача потрібною інформацією для комфортної роботи з програмою.

ВИСНОВОК

Під час виконання даної лабораторної роботи було набуто навичок роботи з реалізацією діалогового режиму роботи з користувачем в консольних програмах мовою Java. Забезпечено обробку команд користувача у вигляді текстового меню. Забезпечено обробку параметрів командного рядка для визначення режиму роботи програми..