Лабораторна робота №4

**ІНТЕРАКТИВНІ КОНСОЛЬНІ ПРОГРАМИ ДЛЯ ПЛАТФОРМИ JAVA SE**

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

**ВИМОГИ**

**Розробник:**

- Веремчук Дарина Анатоліївна;

- КІТ-119д;

- Варіант №5.

**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:**

ArgsHandler handler; // обробник консольних команд (-h, -d)

Scanner in; // змінна для введення даних

String word; // текст, у якому проводиться пошук

final String[] lines; // масив підрядків

List<String> words; // список рядків

**Ієрархія та структура класів:**

**class** Main – точка входу в програму.

**class** UI – діалоговий клас, реалізує текстове меню роботи.

**class** NewHelper – клас, який виконує індивідуальне завдання.

**class** ArgsHandler – клас, який обробляє консольні команди.

**ТЕКСТ ПРОГРАМИ**

Текст класу **Main**:

package ua.oop.khpi.veremchuk04;

public class Main {

/\*\*

\* An entry point - main method.

\* @param args - arguments of main method

\*/

public static void main(String[]args) {

ArgsHandler handler = new ArgsHandler(args);

if (!handler.empty()) {

handler.execute();

}

final int exit = 0;

final int setValues = 1;

final int getValues = 2;

final int execute2 = 3;

final int printResult = 4;

String word = null;

/\*\*

\* Our dialog menu with checking of input argumet's of program

\*/

do {

UI.mainMenu();

UI.enterChoice();

switch (UI.getChoice()) {

case exit:

if (ArgsHandler.isDebug()) {

System.out.println("\nYou chosen 0. Exiting...");

System.out.format("%n############################################################### DEBUG #############################################################");

}

break;

case setValues:

if (ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

System.out.println("\nYou chosen 1. Setting values...");

}

word = UI.enterValues();

break;

case getValues:

if (ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

System.out.println("\nYou chosen 2. Getting values...");

}

if (word != null ) {

UI.printText(word);

} else {

System.out.format("%nFirst you need to add values.");

}

break;

case execute2:

if (ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

System.out.println("\nYou chosen 3. Executing task...");

}

if (word != null) {

final String []lines = NewHelper.DivString(word);

System.out.println("\nTask done...");

if (ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

}

} else {

System.out.format("%nFirst you need to add values.");

}

break;

case printResult:

if (ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

System.out.format("%nYou chosen 4. "

+ "Printing out result...%n");

}

if (word != null) {

final String[] lines2 = NewHelper.DivString(word);

for (final String line2 : lines2) {

NewHelper.printSymbols(line2);

NewHelper.printSymbolNumbers(line2);

}

if(ArgsHandler.isDebug()) {

System.out.format("%n############################################################### DEBUG #############################################################");

}

else {

System.out.format("%nFirst you need to add values.");

}

break;

}

default:

System.out.println("\nEnter correct number.");

}

} while (UI.getChoice() != 0);

}

}

Текст класу **NewHelper**:

package ua.oop.khpi.veremchuk04;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class NewHelper {

/\*\*

\* The method that printing symbols with using String Builder object

\*

\* @param line - our text after splitting

\*/

public static void printSymbols(final String line) {

StringBuilder builder = new StringBuilder();

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

//Checking for ' '/'!'/'.'/':'

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

{

continue;

}

builder.append("\t");

builder.append(symbol);

}

if (ArgsHandler.isDebug()) {

System.out.println("\nInput Text: " + builder.toString());

}

else {

System.out.print(builder.toString());}

System.out.println();

}

/\*\*

\* The method that printing integer value (ASCII CODE) of symbol

\*

\* @param line - our line of symbols where we looking for (ASCII CODE)

\*/

public static void printSymbolNumbers(final String line) {

StringBuilder builder = new StringBuilder();

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

//Checking for ' '/'!'/'.'/':'

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

{

continue;

}

builder.append("\t");

builder.append((int)symbol);

}

if (ArgsHandler.isDebug()) {

System.out.println("\nTask Result: " + builder.toString());

}

else {

System.out.print(builder.toString());

}

System.out.println();

}

/\*\*

\* The method that splits text

\*

\* @param text - text we are splitting

\* @return - returns array of words

\*/

public static String[] DivString(String text) {

if(ArgsHandler.isDebug())

{

System.out.format("%nString text = "

+ text +" (a string that stores our text that we'll split.)");

System.out.println();

};

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) {

continue;

}

builder.append(symbol);

}

words.add(builder.toString());

builder = new StringBuilder();

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);

}

if(ArgsHandler.isDebug())

{

System.out.format("%nString output = ");

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

System.out.format(output[i]);

}

System.out.format (" (our array of output strings after Split Method!)");

System.out.format("%n############################################################### DEBUG #############################################################");

System.out.println();

}

return output;

}

}

Текст класу **UI**:

package ua.oop.khpi.veremchuk04;

import java.util.Scanner;

public class UI {

private UI() {

}

/\*\* Gets values from user. \*/

private static Scanner scan = new Scanner(System.in);

/\*\* User's choice in main menu. \*/

private static int choice;

static int getChoice() {

return choice;

}

/\*\*

\* The method that printing a dialog menu

\*/

static void mainMenu() {

System.out.format("%n1. Enter values.%n");

System.out.format("2. Print values.%n");

System.out.format("3. Task completion.%n");

System.out.format("4. Print result.%n");

System.out.format("0. Exit.%n");

System.out.format("Enter your choose: ");

}

/\*\*

\* The method that scans our choice

\*/

static void enterChoice() {

choice = scan.nextInt();

scan.nextLine();

}

/\*\*

\* The method that adding a values in the text

\*

\* @return text of our values

\*/

public static String enterValues()

{

System.out.println("Enter text:\n");

Scanner in = new Scanner(System.in);

String text = in.nextLine();

return text;

}

/\*\*

\* The method that printing our input data

\*

\* @param text that we have entered

\*/

public static void printText(String text) {

System.out.println("Entered text:" + text);

}

}

Текст класу **ArgsHandler**:

package ua.oop.khpi.veremchuk04;

public class ArgsHandler {

/\*\* An arguments of command line. \*/

private String[] arguments;

/\*\* Is debug mode on? \*/

private static boolean debug = false;

static boolean isDebug() {

return debug;

}

ArgsHandler(final String[] args) {

this.arguments = args;

}

boolean empty() {

return arguments.length == 0;

}

void execute() {

for (String i : arguments) {

switch (i) {

case "-h":

System.out.println("\nAuthor: Veremchuk Darina, KIT119-D");

System.out.println("Task: Enter text. "

+ "Output the initial text and ASCI-code of this text.");

System.out.println("-h (-help) : Print info about "

+ "the work and console commands.");

System.out.println("-d (-debug) : "

+ "Displays additional data.");

break;

case "-help":

System.out.println("\nAuthor: "

+ "Veremchuk Darina, KIT119-D");

System.out.println("Task: Enter text. "

+ "Output the initial text and ASCI-code of this text.");

System.out.println("-h (-help) : Print info about "

+ "the work and console commands.");

System.out.println("-d (-debug) : "

+ "Displays additional data.");

break;

case "-d":

debug = true;

break;

case "-debug":

debug = true;

break;

default:

System.out.format("%n Incorrect command %s.%n", i);

System.out.println("-h, -help, -d, -debug "

+ "is only allowed.");

}

}

}

}

**РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ**

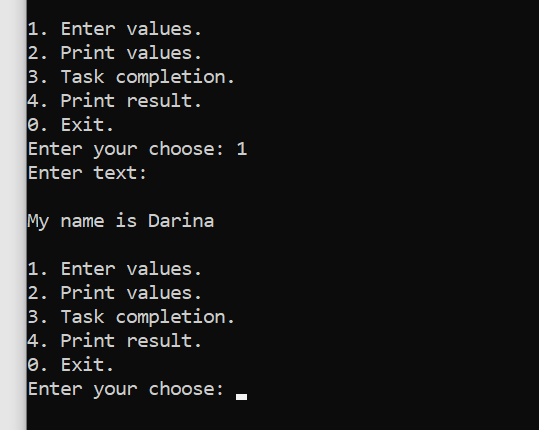
****

Рисунок 4.1 – Введення даних

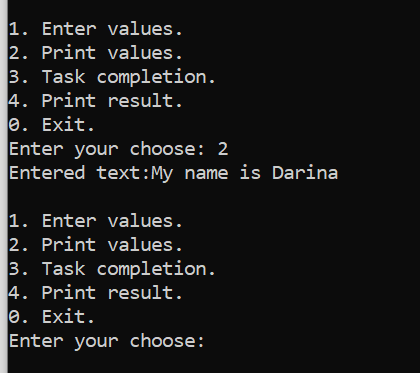
****

Рисунок 4.2 – Вивід початкових даних

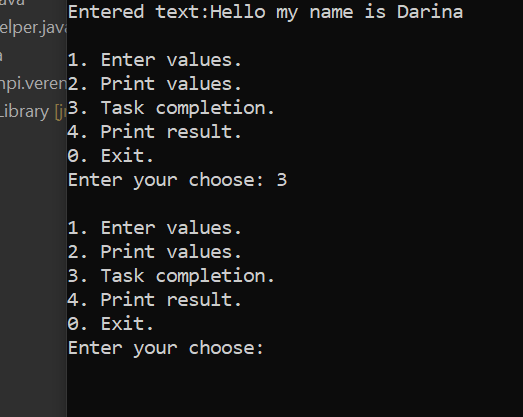
****

Рисунок 4.3 – Виконання завдання

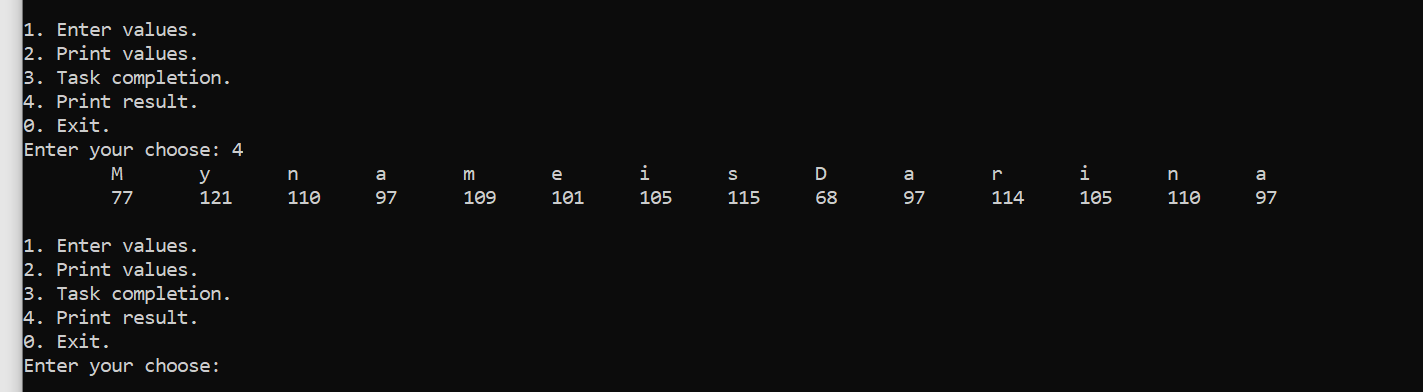
****

Рисунок 4.4 – Вивід результату

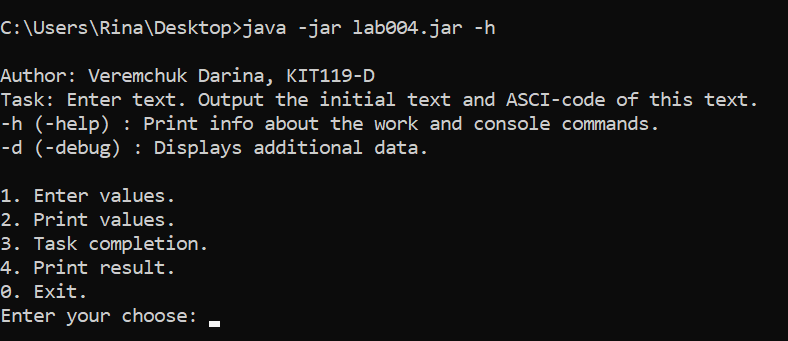
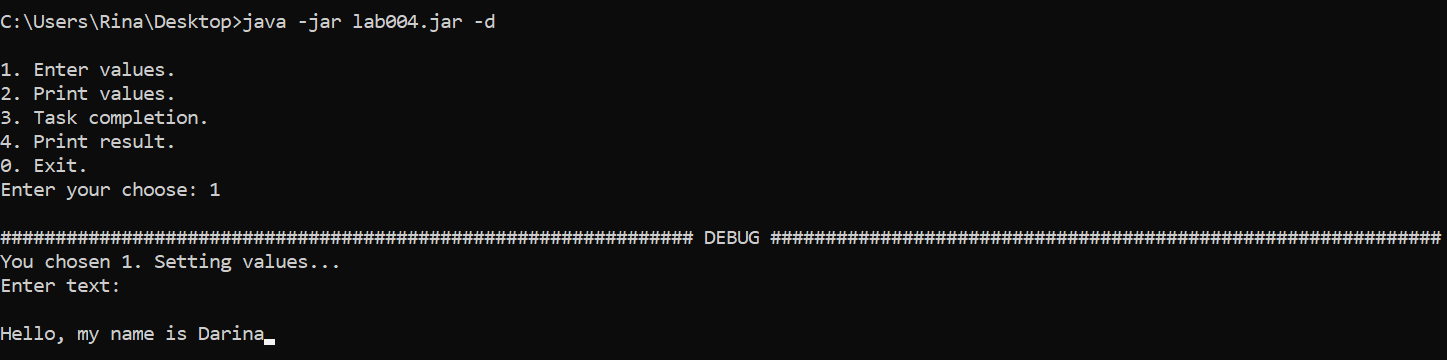
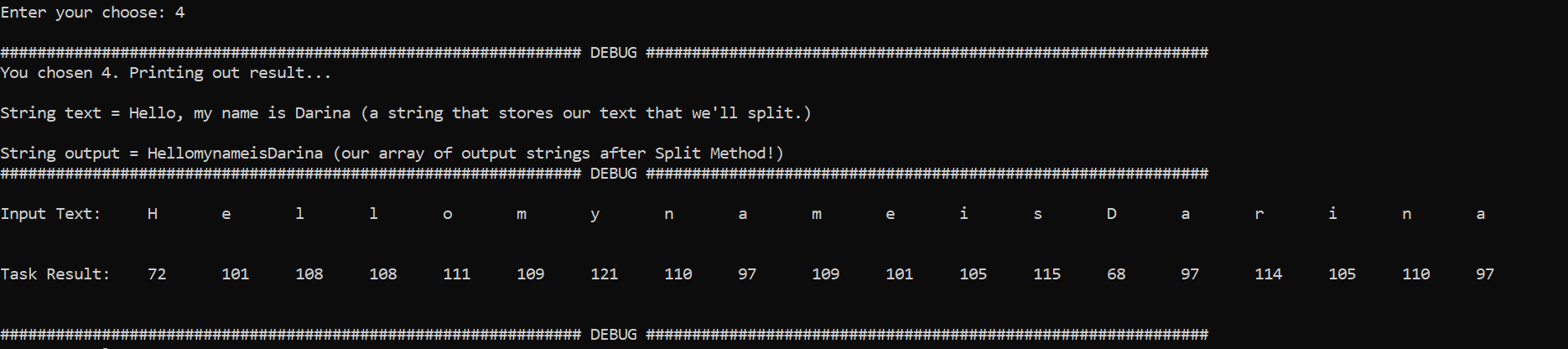
****

Рисунок 4.5 – Використання консольної команди -h

****

а)



б)

Рисунок 4.6 – Використання консольної команди –d: а) робота у Debug режимі; б) відображення додаткових даних у процесі роботи програми

**ВАРІАНТИ ВИКОРИСТАННЯ**

Програму можна використовувати для заміни символів (десяткових цифр, латинського алфавіту, розділових знаків, керуючих символів) їх числовим кодом. Програму можна запускати в режимі Debug, задля відладки та перегляду значень тимчасових змінних для полегшення роботи з програмою. Також можна запускати програму у режимі Help, для виведення інформація про автора програми та допоміжних функцій програми.

**ВИСНОВОК**

Отримала практичні навички реалізації діалогового режиму роботи з користувачем в консольних програмах. Розроблено клас, що реалізує діалоговий режим роботи, а також клас, який оброблює консольні команди. Використала пакет java.util.Scanner, importjava.util.ArrayList, import java.util.List. Програма виконується без помилок.