

№4 Тема: Інтерактивні консольні програми для платформи Java SE.

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

1 Індивідуальне завдання

1.1 Розробник

Студент Малохвій Едуард Едуардович, КІТ-26А, Варіант 8 (Завдання №8).

1.2 Вимоги

- 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.
- 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.

1.3 Завдання

Ввести текст. У тексті знайти всі пари слів, з яких одне є обігом (словом навпаки) іншого (наприклад: "abc"-"cba", "def"-"fed"). Результат вивести у вигляді таблиці.

2 Розробка програми

2.1 Засоби ООП

Під час вирішенн поставленної задачі було використано паттерн Command для інкапсуляції операцій під виглядом команд та патер Decorator для розширення функціоналу команд для надання їм властивостей виведення детальної інформації для подальшої відладки. Для більш гнучкішої реалізації фабрики команд використано рефлексію.

2.2 Ієрархія та структура класів

Проект містить наступні пакети:

- shell - містить інтерактивну оболонку та її допоміжні класи.
- command - містить у собі перелік команд, та фабрику для їх створення.

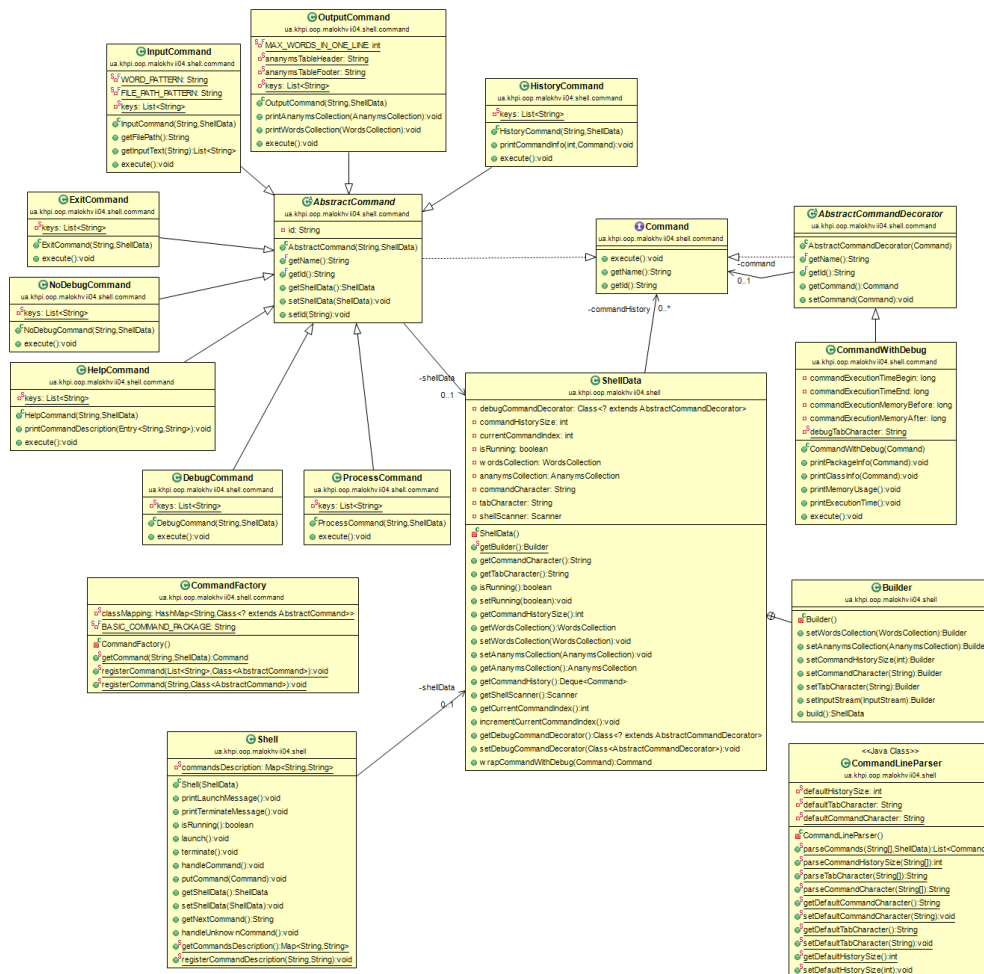


Рис. 1 - Діаграма класів із пакету shell

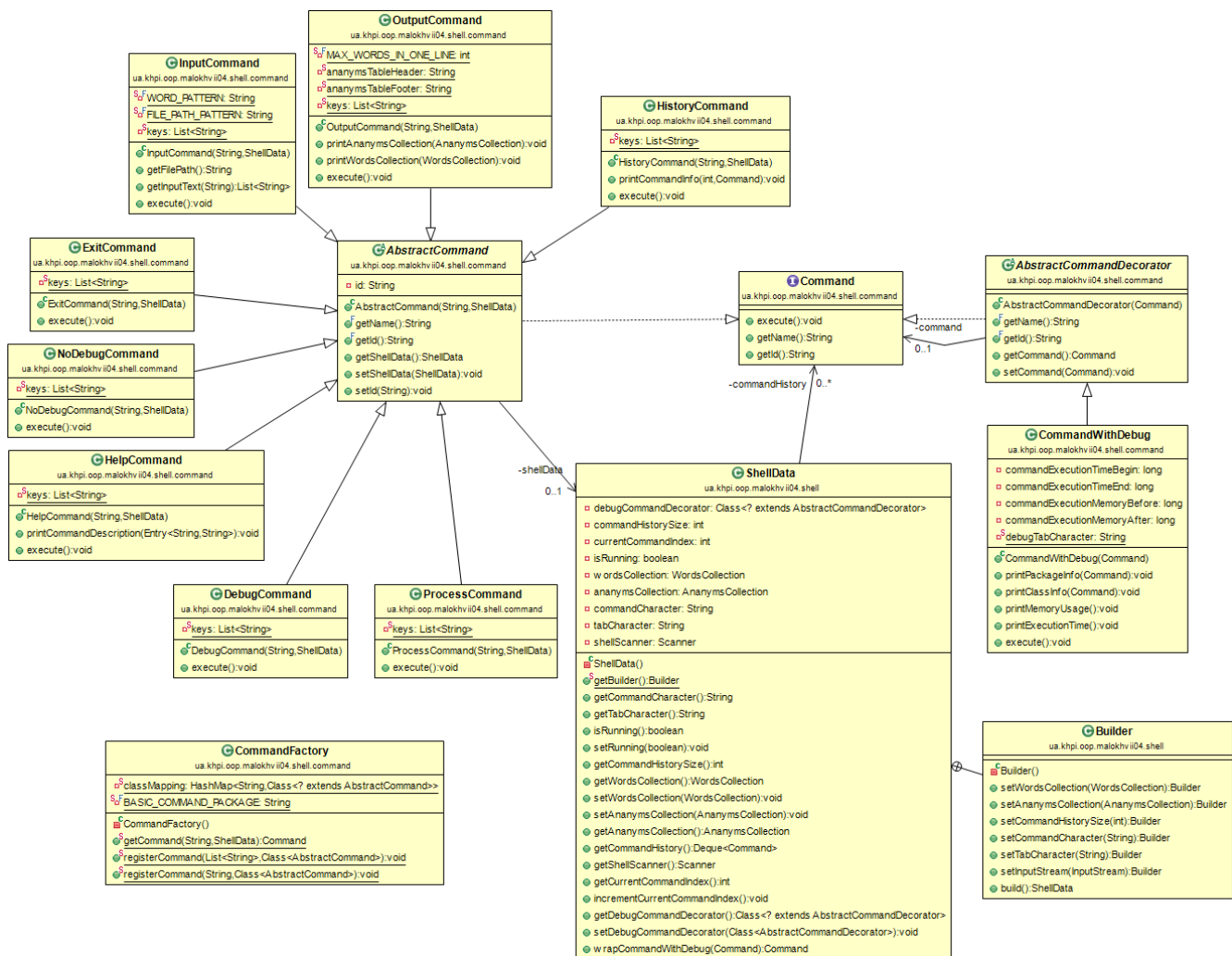


Рис. 2 - Діаграма класів із пакету command

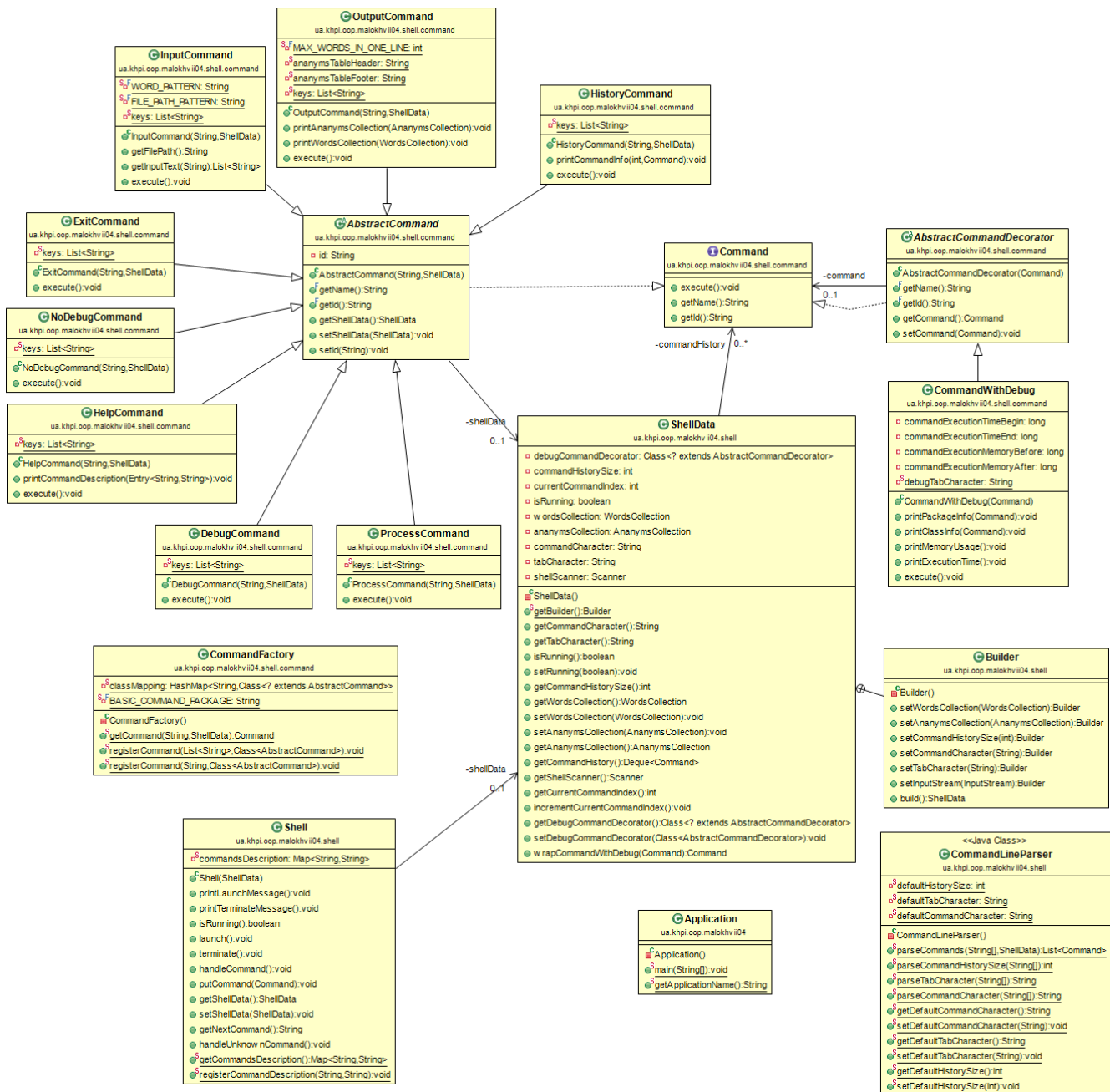


Рис. 3 - Загальна діаграма класів

2.3 Опис програми

Згідно завдання реалізовано обробку параметрів отриманих через командний рядок, реалізовано довідку для усіх команд, реалізовано необхідний перелік команд.

2.4 Важливі фрагменти програми

Нижче наведено фрагмент точки входу програми

(https://sourceforge.net/p/kit26a-cpp/code/HEAD/tree/malokhvii_eduard/src/ua/khpi/oop/malokhvii04/).

```
package ua.khpi.oop.malokhvii04;

import java.io.File;
import java.io.IOException;
import java.net.URISyntaxException;

import ua.khpi.oop.malokhvii03.text.AnanymsCollection;
import ua.khpi.oop.malokhvii03.text.WordsCollection;
import ua.khpi.oop.malokhvii04.shell.CommandLineParser;
import ua.khpi.oop.malokhvii04.shell.Shell;
import ua.khpi.oop.malokhvii04.shell.ShellData;
import ua.khpi.oop.malokhvii04.shell.command.Command;
import ua.khpi.oop.malokhvii04.shell.command.CommandFactory;

public final class Application {

    public static void main(final String[] args) throws IOException {
        WordsCollection wordsCollection = new WordsCollection();
        AnanymsCollection ananymsCollection = new AnanymsCollection();

        ShellData shellData = ShellData.getBuilder()
            .setWordsCollection(wordsCollection)
            .setAnanymsCollection(ananymsCollection)
            .setCommandHistorySize(
                CommandLineParser.parseCommandHistorySize(args))
            .setCommandCharacter(
                CommandLineParser.parseCommandCharacter(args))
            .setTabCharacter(CommandLineParser.parseTabCharacter(args))
            .setInputStream(System.in).build();

        Shell shell = new Shell(shellData);
        shell.launch();

        for (Command command : CommandLineParser.parseCommands(args,
            shellData)) {
            shell.putCommand(command);
            shell.handleCommand();
        }

        while (shell.isRunning()) {
            Command command = CommandFactory.getCommand(shell.getNextCommand(),
                shell.getShellData());
            if (command != null) {
                shell.putCommand(command);
                shell.handleCommand();
            } else {
                shell.handleUnknownCommand();
            }
        }
        shell.terminate();
    }

    public static String getApplicationName() {
```

```

        File file = null;
        try {
            file = new File(Application.class.getProtectionDomain()
                .getCodeSource().getLocation().toURI());
        } catch (URISyntaxException exception) {

        }
        return file.getName();
    }
}

```

3 Результати роботи

Нижче наведено виведення обчислень у вигляді інтерактивної консолі.

```

Ed@malokhvii-ee MINGW32 /d/home/projects/ua.khpi.oop.malokhvii
$ java -jar ananymys.jar
Shell was launched
... Description: Used to find in the text ananymys, such as "def" - "fed"
... An ananym is a word whose spelling is derived by reversing the spelling
... of another word. It is therefore a special type of anagram

>>> -h

Usage: ananymys.jar
... -debug -d          Switch on debug information
... -exit -e           End interactive console session
... -help -h           Get help on all the interactive console commands
... -history -hs       Show commands call history
... -input -i          Input of input data in the form of text files
... -no-debug -nd      Switch off debug information
... -output -o         Output of an anagram search result in the input text
... -process -p        Search anagrams in the indexed input text

>>> -i

Please, enter file path with input text:
... D:\home\projects\ua.khpi.oop.malokhvii\src\ua\khpi\oop\malokhvii03\text-259.txt

>>> -p

>>> -o

List of words loaded from text file:
... [A]: As
... [C]: Compiler Classpath Community Corporation
... [G]: GNU General Gosling
... [I]: It IcedTea
... [J]: JVM Java James
... [L]: License
... [M]: May Microsystems
... [O]: Others Oracle

```

Рис. 3 - Фрагмент демонстраційної програми

```
>>> -0

List of words loaded from text file:
... [A]: As
... [C]: Compiler Classpath Community Corporation
... [G]: GNU General Gosling
... [I]: It IcedTea
... [J]: JVM Java James
... [L]: License
... [M]: May Microsystems
... [O]: Others Oracle
... [P]: Public Process
... [S]: Sun
... [T]: The
... [W]: Web WORA
... [a]: as at all are and any avaJ also applets anywhere acquired
... application alternative applications architecture
... [b]: by but bew been based browser bytecode
... [c]: can core code class computer compiled compiler component compilers compliance concurrent
...
... [d]: dna derives detroper designed detroppus developed developers dependencies
... [e]: eht ehT era esu eno eerf either elcar0
... [f]: fo for few from fewer facilities
... [g]: general gnimmarginorp
... [h]: has have htiw htob hcihw hguohtla
... [i]: in is it its intended implementation implementations
... [l]: low let level licenses language libraries laicremmoc
... [m]: most much meaning machine machines
... [n]: no ni need noillim noisrev
... [o]: of on once object original oriented originally
... [p]: plugin purpose possible platform platforms proprietary programming
... [r]: rof run rehto revres reilrae ralupop released reference relicensed regardless recompilation
...
... [s]: si sA such sisab since syntax support standard snoisrev segaungal seinapmoc
... srepleved snoitacilppa specifically specifications
... [t]: to the that than them tsom these tneilc tsetal typically technologies
...
... [u]: under
```

Рис. 4 - Фрагмент демонстраційної програми

```
Result of searching anonyms in text:
... Amount of anonyms: 39

Table of anonyms from text:
... |-----|
... | Word | Reversed word |
... |-----|-----|
... | As | sA |
... | Java | avaJ |
... | Oracle | elcar0 |
... | The | ehT |
... | are | era |
... | and | dna |
... | avaJ | Java |
... | applications | snoitacilppa |
... | by | yb |
... | dna | and |
... | developers | srepleved |
... | eht | the |
... | ehT | The |
... | era | are |
... | elcar0 | Oracle |
... | fo | of |
... | for | rof |
... | gnimmarginorp | programming |
... | htiw | with |
... | hcihw | which |
... | in | ni |
... | is | si |
... | level | level |
... | most | tsom |
... | no | on |
... | ni | in |
... | of | fo |
... | on | no |
... | programming | gnimmarginorp |
... | rof | for |
```

Рис. 5 - Фрагмент демонстраційної програми

```

>>> -d

Switch on debug infomation

>>> -hs

[Debug]: 23d337f9-4b42-4673-adf3-f684378fabe4
    at ua.khpi.oop.malokhvii04.shell.CommandWithDebug.execute(CommandWithDebug.java:72)
    at ua.khpi.oop.malokhvii04.shell.Shell.handleCommand(Shell.java:55)
    at ua.khpi.oop.malokhvii04.Application.main(Application.java:46)

... Package name: ua.khpi.oop.malokhvii04.shell
... Package vendor: null
... Package version: null

... Simple name: HistoryCommand
... Canonical name: ua.khpi.oop.malokhvii04.shell.HistoryCommand
... Class loader: sun.misc.Launcher$AppClassLoader

... Start execution HistoryCommand

... Index: 4
... Name: HistoryCommand
... Id: 23d337f9-4b42-4673-adf3-f684378fabe4

... Index: 3
... Name: DebugCommand
... Id: ee78e063-b6c7-42d0-a3c1-a3a5108b2d73

... Index: 2
... Name: ProcessCommand
... Id: 24563aba-94a6-49de-9d46-7f633c46bc88

... Index: 1
... Name: InputCommand
... Id: bc2ff1b3-9845-4f5b-b2a2-c3e0afc76c25

... Finish execution HistoryCommand

```

Рис. 6 - Фрагмент демонстраційної програми

```

Ed@malokhvii-ee MINGW32 /d/home/projects/ua.khpi.oop.malokhvii
$ java -jar ananymys.jar -d -h -tc=???
Shell was launched
??? Description: Used to find in the text ananymys, such as "def" - "fed"
??? An ananym is a word whose spelling is derived by reversing the spelling
??? of another word. It is therefore a special type of anagram

Switch on debug infomation

[Debug]: 308e42ef-6721-485b-beld-721232d42845
    at ua.khpi.oop.malokhvii04.shell.CommandWithDebug.execute(CommandWithDebug.java:72)
    at ua.khpi.oop.malokhvii04.shell.Shell.handleCommand(Shell.java:55)
    at ua.khpi.oop.malokhvii04.Application.main(Application.java:38)

??? Package name: ua.khpi.oop.malokhvii04.shell
??? Package vendor: null
??? Package version: null

??? Simple name: HelpCommand
??? Canonical name: ua.khpi.oop.malokhvii04.shell.HelpCommand
??? Class loader: sun.misc.Launcher$AppClassLoader

??? Start execution HelpCommand

Usage: ananymys.jar
??? -debug -d          Switch on debug information
??? -exit -e          End interactive console session
??? -help -h          Get help on all the interactive console commands
??? -history -hs      Show commands call history
??? -input -i         Input of input data in the form of text files
??? -no-debug -nd     Switch off debug information
??? -output -o        Output of an anagram search result in the input text
??? -process -p       Search anagrams in the indexed input text

??? Finish execution HelpCommand

??? Total memory before execution: 128974848 bytes

```

Рис. 7 - Фрагмент демонстраційної програми

Висновки

У ході виконання лабораторної роботи були покращені навички розробки інтерактивних консольних програм мовою Java. Реалізовано діалоговий режим роботи з користувачем в консольній програмі.