Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**ЗВІТ**

Про виконання лабораторної роботи **№ 5**

**«Реалізація консольного меню з використанням паттерну “Команда”»**

з дисципліни **«Прикладне програмування»**

**Лектор:**

доц. кафедри АСУ

Скорохода О.В.

**Виконав:**

студ. групи КН-203

Репей А.В.

**Прийняв:**

асис. кафедри АСУ

Мельник Р. В.

«\_\_\_» \_\_\_\_\_\_ 2022 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2022

**Тема:** Реалізація консольного меню з використанням паттерну “Команда”

**Мета:** Реалізувати консольне меню до індивідуального завдання з використанням паттерну «Команда». На цьому етапі не потрібно реалізовувати функціонал конкретних пунктів меню – тільки створити «кістяк» вашої програми

**Посилання на репозиторій Github**: https://github.com/temerep/labs\_ap/tree/main/knight\_ammunition

**Хід виконання лабораторної роботи:**

Створити консольну програму на мові програмування Java “Лицарі”.

Визначити ієрархію амуніції лицаря. Екіпірувати лицаря.

Підрахувати вартість. Здійснити сортування амуніції на основі ваги. Знайти елементи амуніції, що відповідають заданому діапазону параметрів ціни.

Окрім цього, програма повинна відповідати таким додатковим вимогам:  
• Використовувати можливості ООП: класи, успадкування, поліморфізм, інкапсуляцію.  
• Кожний клас повинен мати назву, яка повністю описує його суть, і інформативний склад.  
• Успадкування потрібно використовувати тільки тоді, коли воно має сенс.  
• При записі програми потрібно використовувати домовленості щодо оформлення коду java code convention.  
• Класи потрібно грамотно розкласти по пакетах.  
• Для збереження параметрів ініціалізації потрібно використовувати файли.  
• Повинне бути консольне меню для вибору різних дій, наприклад: «завантажити з файлу», «довідка», «вихід» тощо.

**Додаткове завдання**

Реалізуйте консольне меню до вашого завдання з використанням паттерну  
«Команда». На цьому етапі не потрібно реалізовувати функціонал конкретних пунктів меню – тільки створити «кістяк» вашої програми.

**Діаграма**

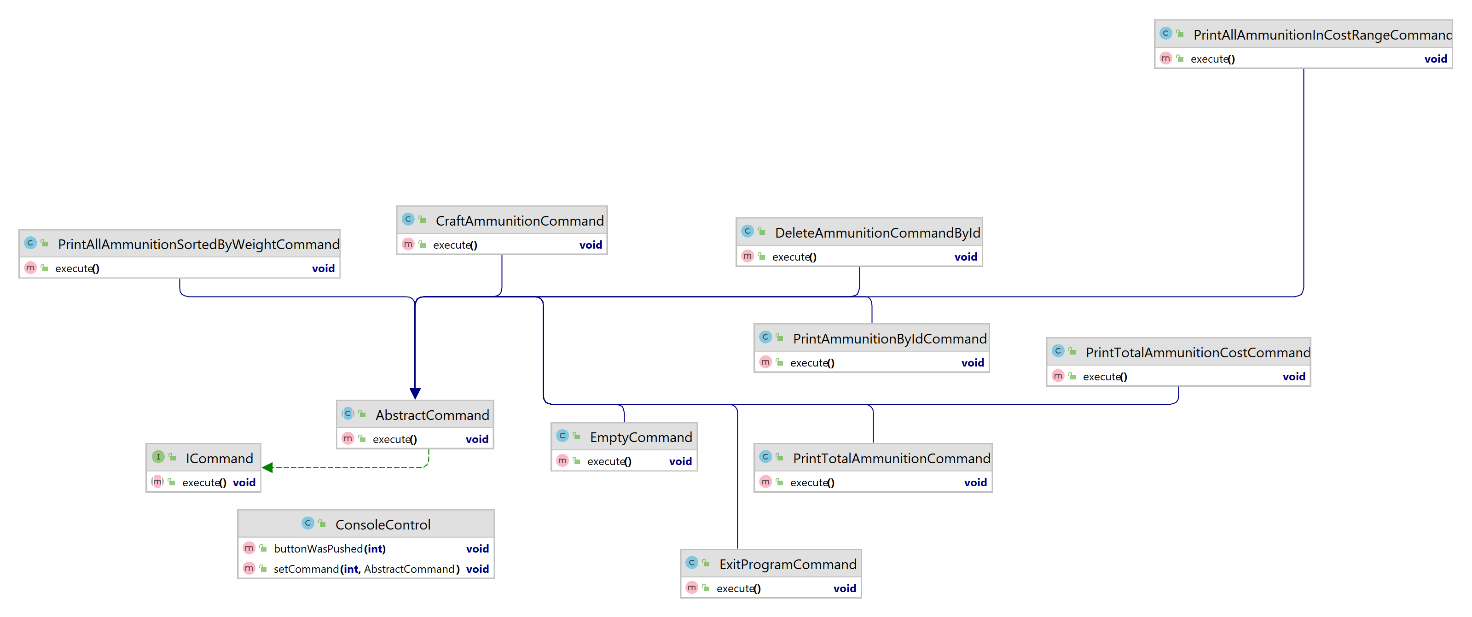


Рисунок 3 Діаграма класів для класів із пакету Command

Базовий інтерфейс ICommand із єдиним методом - void execute().

Executor-ом або Invoker-oм виступає клас ConsoleControl, який містить методи по обробці вхідних сигналів і інтерпретації їх у виконання відповідної команди – void buttonWasPushed(int) та конфігурації команд у відповідні слоти-приймачі – setCommand (int slot, AbstractCommand command).

**Код програми**

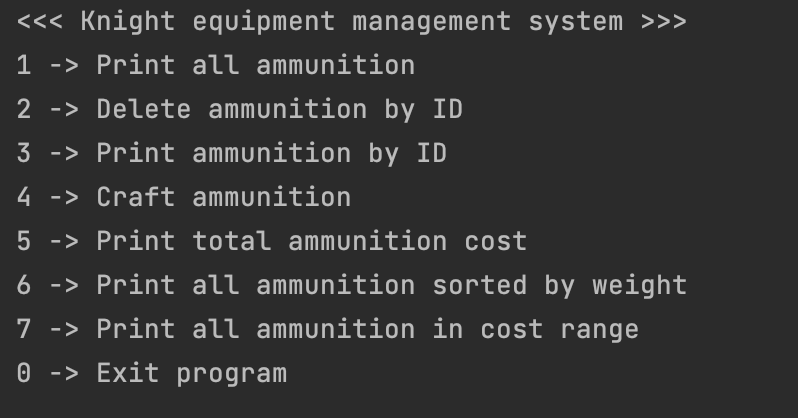
***App.java***

public static ConsoleControl setupConsoleControl() {  
 ConsoleControl consoleControl = new ConsoleControl();  
 AmmunitionFactory ammunitionFactory = new AmmunitionFactoryImpl();  
 AmmunitionController ammunitionController = new AmmunitionController(new AmmunitionService(new AmmunitionDao<>()));  
 ExitProgramCommand exitProgramCommand = new ExitProgramCommand(ammunitionController);  
 CraftAmmunitionCommand craftAmmunitionCommand = new CraftAmmunitionCommand(ammunitionController, ammunitionFactory);  
 DeleteAmmunitionCommandById deleteAmmunitionCommandById = new DeleteAmmunitionCommandById(ammunitionController);  
 PrintAllAmmunitionInCostRangeCommand printAllAmmunitionInCostRangeCommand = new PrintAllAmmunitionInCostRangeCommand(ammunitionController);  
 PrintAllAmmunitionSortedByWeightCommand printAllAmmunitionSortedByWeightCommand = new PrintAllAmmunitionSortedByWeightCommand(ammunitionController);  
 PrintTotalAmmunitionCommand printTotalAmmunitionCommand = new PrintTotalAmmunitionCommand(ammunitionController);  
 PrintTotalAmmunitionCostCommand printTotalAmmunitionCostCommand = new PrintTotalAmmunitionCostCommand(ammunitionController);  
 PrintAmmunitionByIdCommand printAmmunitionByIdCommand = new PrintAmmunitionByIdCommand(ammunitionController);  
 consoleControl.setCommand(0, exitProgramCommand);  
 consoleControl.setCommand(1, printTotalAmmunitionCommand);  
 consoleControl.setCommand(2, deleteAmmunitionCommandById);  
 consoleControl.setCommand(3, printAmmunitionByIdCommand);  
 consoleControl.setCommand(4, craftAmmunitionCommand);  
 consoleControl.setCommand(5, printTotalAmmunitionCostCommand);  
 consoleControl.setCommand(6, printAllAmmunitionSortedByWeightCommand);  
 consoleControl.setCommand(7, printAllAmmunitionInCostRangeCommand);  
 ammunitionController.initView();  
 return consoleControl;  
}

***AmmunitionView.java***

public void initView() {  
 System.*out*.println("<<< Knight equipment management system >>>");  
 updateView();  
}  
  
public void updateView() {  
 StringBuilder stringBuilder = new StringBuilder()  
 .append("1 -> Print all ammunition\n")  
 .append("2 -> Delete ammunition by ID\n")  
 .append("3 -> Print ammunition by ID\n")  
 .append("4 -> Craft ammunition\n")  
 .append("5 -> Print total ammunition cost\n")  
 .append("6 -> Print all ammunition sorted by weight\n")  
 .append("7 -> Print all ammunition in cost range\n")  
 .append("0 -> Exit program\n");  
 System.*out*.println(stringBuilder);  
}

**Результат виконання програми**

****

**Висновок**

На лабораторній роботі я реалізував консольне меню до свого завдання з використанням паттерну “Команда” та побудував діаграму.