Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

до лабораторної роботи № 5 з

курсу

*Прикладне програмувння*  на

тему:

**« Комплексна лабораторна робота »**

Виконав: студент ОІ-24

**Стащишин Юрій**

Прийняв:

**Чорненький В. Я.**

Львів – 2024

***Лабораторна робота № 5***

**Тема:** Комплексна лабораторна робота

**Мета роботи:** Навчитися використовувати патерн «Команда»

**Варіант - 13**

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.

**Фургон кави.** Завантажити фургон певного об’єму вантажем на певну суму з різних видів кави, що може знаходитися у різних фізичних станах (зернова, мелена, розчинна в банках і пакетиках). Враховувати об’єм кави разом з упаковкою. Здійснити сортування товарів на основі співвідношення ціни та ваги. Знайти товар у фургоні, що відповідає заданому діапазону параметрів якості.

**Код програми**

package truck.model;  
  
// Reciver  
public class Truck {  
 // Метод для сортування кави на основі співвідношення ціни та ваги  
 public void sortCoffee(){  
 System.*out*.println("Сортування кави...");  
 // Далі логіка  
 }  
  
 // Метод для знаходження товару у фургоні  
 public void searchCoffee(){  
 System.*out*.println("Пошук товару...");  
 // Далі логіка  
 }  
  
 // Метод для завантаження фургону кавою  
 public void loadTruck(){  
 System.*out*.println("Завантажуємо фургон кавою...");  
 // далі логіка  
 }  
}

package truck.model;  
  
public class Shop {  
 public void addCoffee(){  
 System.*out*.println("Додаємо каву...");  
 }  
  
 public void showAvailableCoffee(){  
 System.*out*.println("Кава на вибір...");  
 }  
}

package truck.model;  
import truck.command.\*;  
  
public class Seller {  
 Command add;  
 Command show;  
  
 public Seller(Command show, Command add) {  
 this.add = add;  
 this.show = show;  
 }  
  
 public void addCoffee(){  
 add.execute();  
 }  
  
 public void showCoffee(){  
 show.execute();  
 }  
}

package truck.model;  
import truck.command.\*;  
  
// Invoker  
public class Carrier {  
 Command load;  
 Command search;  
 Command sort;  
  
 public Carrier(Command load, Command search, Command sort) {  
 this.load = load;  
 this.search = search;  
 this.sort = sort;  
 }  
  
 public void loadTruck(){  
 load.execute();  
 }  
  
 public void searchCoffee(){  
 search.execute();  
 }  
  
 public void sortCoffee(){  
 sort.execute();  
 }  
}

package truck.model;  
  
public class Coffee {  
 private String name;  
 private double weight;  
 private double price;  
  
}

package truck.model;  
  
public class GrainCoffee extends Coffee{  
}

package truck.model;  
  
public class GroundCoffee extends Coffee{  
}

package truck.model;  
  
public class InstantCoffee extends Coffee{  
}

package truck.model;  
  
// Клас для збереження даних про упаковку  
public class Package {  
}

package truck.command;  
  
public interface Command {  
 void execute();  
}

package truck.command;  
  
import truck.model.Shop;  
  
public class AddCoffeeCommand implements Command{  
 Shop shop;  
  
 public AddCoffeeCommand(Shop shop) {  
 this.shop = shop;  
 }  
  
 @Override  
 public void execute(){  
 shop.addCoffee();  
 }  
}

package truck.command;  
  
import truck.model.Shop;  
  
public class ShowAvailableCoffeeCommand implements Command{  
 Shop shop;  
  
 public ShowAvailableCoffeeCommand(Shop shop) {  
 this.shop = shop;  
 }  
  
 @Override  
 public void execute(){  
 shop.showAvailableCoffee();  
 }  
}

package truck.command;  
  
import truck.model.Truck;  
  
public class SortCoffeeCommand implements Command{  
 Truck truck;  
  
 public SortCoffeeCommand(Truck truck) {  
 this.truck = truck;  
 }  
  
 @Override  
 public void execute(){  
 truck.sortCoffee();  
 }  
}

package truck.command;  
  
import truck.model.Truck;  
  
public class SearchCoffeeCommand implements Command{  
 Truck truck;  
  
 public SearchCoffeeCommand(Truck truck) {  
 this.truck = truck;  
 }  
  
 @Override  
 public void execute(){  
 truck.searchCoffee();  
 }  
}

package truck.command;  
  
import truck.model.Truck;  
  
public class LoadTruckCommand implements Command{  
 Truck truck;  
  
 public LoadTruckCommand(Truck truckManager) {  
 this.truck = truckManager;  
 }  
  
 @Override  
 public void execute(){  
 truck.loadTruck();  
 }  
}

package truck.command;  
  
public class LoadFromFileCommand implements Command{  
 @Override  
 public void execute(){  
 System.*out*.println("Завантаження даних з файлу...");  
 // Далі логіка  
 }  
}

package truck.command;  
  
public class ExitCommand implements Command{  
 @Override  
 public void execute(){  
 System.*out*.println("Вихід з програми...");  
 System.*exit*(0);  
 }  
}

package truck;  
  
import truck.command.\*;  
import truck.model.\*;  
  
import java.util.Scanner;  
  
public class ConsoleMenu{  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 Truck truck = new Truck();  
 Shop shop = new Shop();  
  
 Command loadCommand = new LoadTruckCommand(truck);  
 Command searchCommand = new SearchCoffeeCommand(truck);  
 Command sortCommand = new SortCoffeeCommand(truck);  
 Command addCoffeeCommand = new AddCoffeeCommand(shop);  
 Command showCoffeeCommand = new ShowAvailableCoffeeCommand(shop);  
 Command exitCommand = new ExitCommand();  
  
 Carrier carrier = new Carrier(loadCommand, searchCommand, sortCommand);  
 Seller seller = new Seller(showCoffeeCommand, addCoffeeCommand);  
  
 boolean running = true;  
 while (running) {  
 System.*out*.println("Меню:");  
 System.*out*.println("1. Завантажити фургон кавою");  
 System.*out*.println("2. Сортувати каву");  
 System.*out*.println("3. Шукати каву");  
 System.*out*.println("4. Додати каву в магазин");  
 System.*out*.println("5. Показати доступну каву в магазині");  
 System.*out*.println("6. Вихід");  
  
 System.*out*.print("Оберіть опцію: ");  
 int choice = scanner.nextInt();  
  
 switch (choice) {  
 case 1:  
 carrier.loadTruck();  
 break;  
 case 2:  
 carrier.sortCoffee();  
 break;  
 case 3:  
 carrier.searchCoffee();  
 break;  
 case 4:  
 seller.addCoffee();  
 break;  
 case 5:  
 seller.showCoffee();  
 break;  
 case 6:  
 exitCommand.execute();  
 break;  
 default:  
 System.*out*.println("Невірний вибір, спробуйте ще раз.");  
 }  
 }  
 scanner.close();  
 }  
}

**Висновок**

Завдяки цій лабораторній роботі я навчився використовувати патерн «Команда». В ході виконання лабораторної роботи я реалізував консольне меню до мого завдання з використанням патерну «Команда». Конкретні пункти меню не були реалізовані, було створено лише кістяк програми.

Посилання на Git: https://github.com/Float36/labs