Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

до лабораторної роботи № 8 з

курсу

*Прикладне програмувння*  на

тему:

**« Комплексна лабораторна робота »**

Виконав: студент ОІ-24

**Стащишин Юрій**

Прийняв:

**Чорненький В. Я.**

Львів – 2024

***Лабораторна робота № 8***

**Тема:** Комплексна лабораторна робота

**Мета роботи:** Додайте логгер з логуванням основних дій та виняткових ситуацій. При цьому дані логу мають зберігатися у файлі, а критичні помилки – додатково надсилатися на email.

**Варіант - 13**

**Код програми**

package truck;  
  
import truck.command.\*;  
import truck.model.\*;  
import truck.model.Package;  
import org.apache.logging.log4j.LogManager;  
import org.apache.logging.log4j.Logger;  
  
import java.util.InputMismatchException;  
import java.util.List;  
import java.util.Scanner;  
  
public class ConsoleMenu {  
 private static final Logger *logger* = LogManager.*getLogger*(ConsoleMenu.class);  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 Truck truck = new Truck(2000);  
 Shop shop = new Shop();  
  
 LoadTruckCommand loadCommand = new LoadTruckCommand(truck);  
 Command searchCommand = new SearchCoffeeCommand(truck);  
 Command sortCommand = new SortCoffeeCommand(truck);  
 AddCoffeeCommand 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) {  
 *logger*.info("Displaying menu options.");  
 System.*out*.println();  
 System.*out*.println("Меню:");  
 System.*out*.println("1. Завантажити фургон кавою");  
 System.*out*.println("2. Показати відсортовану каву в фургоні");  
 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. Завнтажити дані про каву з файлу");  
 System.*out*.println("7. Вихід");  
  
 System.*out*.print("Оберіть опцію: ");  
  
 int choice = -1;  
 while (true) {  
 try {  
 choice = scanner.nextInt();  
 if (choice < 1 || choice > 7) {  
 System.*out*.println("Будь ласка, виберіть число від 1 до 7.");  
 } else {  
 break;  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next();  
 }  
 }  
  
 switch (choice) {  
 case 1:  
 Coffee load = *selectCoffee*(scanner, shop);  
 if (load != null) {  
 *logger*.info("Loading coffee into the truck: {}", load.getName());  
 System.*out*.print("Введіть суму на яку хочете зробити замовлення: ");  
 int sum = scanner.nextInt();  
 int quantity = (int) (sum / load.getPrice());  
 if (quantity > load.getQuantity()) {  
 *logger*.warn("Not enough coffee in the shop: {}", load.getName());  
 System.*out*.println("Недостатньо кави в магазині");  
 break;  
 }  
 loadCommand.setCoffee(load, quantity);  
 loadCommand.execute();  
 } else {  
 *logger*.error("Coffee selection failed.");  
 System.*out*.println("Помилка: Кава не вибрана.");  
 }  
 break;  
 case 2:  
 *logger*.info("Sorting coffee in the truck.");  
 carrier.sortCoffee();  
 break;  
 case 3:  
 *logger*.info("Searching coffee.");  
 carrier.searchCoffee();  
 break;  
 case 4:  
 Coffee coffee = *createCoffee*(scanner);  
 if (coffee != null) {  
 *logger*.info("Adding new coffee to the shop: {}", coffee.getName());  
 addCoffeeCommand.setCoffee(coffee);  
 seller.addCoffee();  
 }  
 break;  
 case 5:  
 *logger*.info("Displaying available coffee in the shop.");  
 seller.showCoffee();  
 break;  
 case 6:  
 *logger*.info("Loading coffee data from file.");  
 shop.loadCoffeesFromFile("NotCoffees.txt");  
 break;  
 case 7:  
 *logger*.info("Exiting the program.");  
 exitCommand.execute();  
 break;  
 default:  
 *logger*.warn("Invalid choice entered: {}", choice);  
 System.*out*.println("Невірний вибір, спробуйте ще раз.");  
 }  
 }  
  
 scanner.close();  
 *logger*.info("Program terminated.");  
 }  
  
private static Coffee selectCoffee(Scanner scanner, Shop shop) {  
 *logger*.info("Selecting coffee from available options.");  
 System.*out*.println("Доступна кава в магазині:");  
 List<Coffee> availableCoffee = shop.getAvailableCoffee();  
  
 if (availableCoffee.isEmpty()) {  
 *logger*.warn("No coffee available in the shop.");  
 System.*out*.println("Кава відсутня.");  
 return null;  
 }  
  
 // Відображення кави з індексами для вибору  
 for (int i = 0; i < availableCoffee.size(); i++) {  
 System.*out*.printf("%d. %s%n", i + 1, availableCoffee.get(i));  
 }  
  
 int choice = -1;  
 while (true) {  
 System.*out*.print("Оберіть номер кави для завантаження у фургон: ");  
 try {  
 choice = scanner.nextInt();  
 if (choice > 0 && choice <= availableCoffee.size()) {  
 return availableCoffee.get(choice - 1);  
 } else {  
 *logger*.warn("Invalid coffee choice: {}", choice);  
 System.*out*.println("Невірний вибір. Будь ласка, виберіть номер від 1 до " + availableCoffee.size() + ".");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next();  
 }  
 }  
 }  
  
 private static Coffee createCoffee(Scanner scanner) {  
 *logger*.info("Starting coffee creation process.");  
 System.*out*.println("Виберіть тип кави: ");  
 System.*out*.println("1. Grain Coffee");  
 System.*out*.println("2. Ground Coffee");  
 System.*out*.println("3. Instant Coffee");  
  
 int typeChoice = -1;  
 while (true) {  
 try {  
 typeChoice = scanner.nextInt();  
 if (typeChoice < 1 || typeChoice > 3) {  
 System.*out*.println("Невірний вибір. Будь ласка, виберіть тип кави від 1 до 3.");  
 *logger*.warn("Invalid coffee type selection: {}", typeChoice);  
 } else {  
 break; // Вихід з циклу, якщо введено коректне число  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 scanner.nextLine(); // Очищення буфера після введення числа  
  
 System.*out*.print("Введіть назву кави: ");  
 String name = scanner.nextLine();  
  
 double weight = -1;  
 while (weight < 0) {  
 System.*out*.print("Введіть вагу кави (в грамах): ");  
 try {  
 weight = scanner.nextDouble();  
 if (weight < 0) {  
 *logger*.warn("Invalid weight input: {}", weight);  
 System.*out*.println("Вага не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну вагу.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double price = -1;  
 while (price < 0) {  
 System.*out*.print("Введіть ціну кави (в грн): ");  
 try {  
 price = scanner.nextDouble();  
 if (price < 0) {  
 *logger*.warn("Invalid price input: {}", price);  
 System.*out*.println("Ціна не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну ціну.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double volume = -1;  
 while (volume < 0) {  
 System.*out*.print("Введіть об'єм кави (в літрах): ");  
 try {  
 volume = scanner.nextDouble();  
 if (volume < 0) {  
 *logger*.warn("Invalid volume input: {}", volume);  
 System.*out*.println("Об'єм не може бути негативним. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректний об'єм.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 // Вибір типу упаковки  
 System.*out*.println("Виберіть тип упаковки: ");  
 System.*out*.println("1. Банка");  
 System.*out*.println("2. Пакет");  
  
 int packagingChoice = -1;  
 while (true) {  
 try {  
 packagingChoice = scanner.nextInt();  
 if (packagingChoice < 1 || packagingChoice > 2) {  
 *logger*.warn("Invalid packaging choice: {}", packagingChoice);  
 System.*out*.println("Невірний вибір типу упаковки. Будь ласка, виберіть 1 або 2.");  
 } else {  
 break; // Вихід з циклу, якщо введено коректне число  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 String packagingMaterial = (packagingChoice == 1) ? "Банка" : "Пакет";  
  
 double packageWeight = -1;  
 while (packageWeight < 0) {  
 System.*out*.print("Введіть вагу упаковки (в грамах): ");  
 try {  
 packageWeight = scanner.nextDouble();  
 if (packageWeight < 0) {  
 *logger*.warn("Invalid package weight input: {}", packageWeight);  
 System.*out*.println("Вага упаковки не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну вагу упаковки.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double packageVolume = -1;  
 while (packageVolume < 0) {  
 System.*out*.print("Введіть об'єм упаковки (в літрах): ");  
 try {  
 packageVolume = scanner.nextDouble();  
 if (packageVolume < 0) {  
 *logger*.warn("Invalid package volume input: {}", packageVolume);  
 System.*out*.println("Об'єм упаковки не може бути негативним. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректний об'єм упаковки.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 int quantity = -1;  
 while (quantity < 0) {  
 System.*out*.print("Введіть кількість упаковок: ");  
 try {  
 quantity = scanner.nextInt();  
 if (quantity < 0) {  
 *logger*.warn("Invalid quantity input: {}", quantity);  
 System.*out*.println("Кількість не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну кількість упаковок.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 Package packaging = new Package(packagingMaterial, packageWeight, packageVolume);  
  
 Coffee coffee = null;  
 switch (typeChoice) {  
 case 1:  
 coffee = new GrainCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 case 2:  
 coffee = new GroundCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 case 3:  
 coffee = new InstantCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 default:  
 *logger*.error("Invalid coffee type choice: {}", typeChoice);  
 System.*out*.println("Невірний вибір типу кави.");  
 }  
  
 if (coffee != null) {  
 *logger*.info("Coffee created: {}", coffee);  
 }  
  
 return coffee;  
 }  
}

**log4j2**

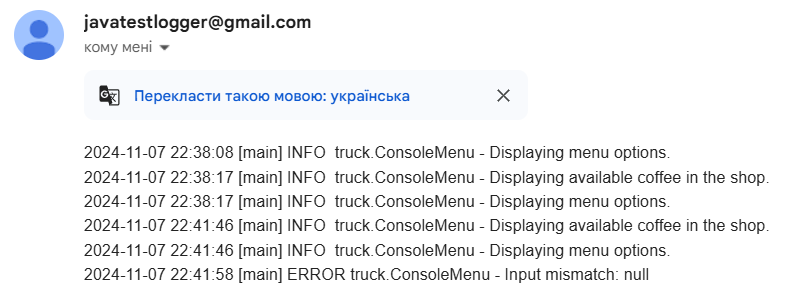
<?xml version="1.0" encoding="UTF-8"?>  
<Configuration status="WARN">  
 <Appenders>  
 <!-- Console Appender -->  
 <Console name="ConsoleAppender" target="SYSTEM\_OUT">  
 <PatternLayout pattern="%msg%n"/>  
 </Console>  
  
 <!-- File Appender -->  
 <File name="FileAppender" fileName="logs/app.log" append="true">  
 <PatternLayout pattern="%d{yyyy-MM-dd HH:mm:ss} [%t] %-5level %logger{36} - %msg%n"/>  
 </File>  
  
 <!-- SMTP Appender -->  
 <SMTP name="EmailAppender" subject="Critical Error in Application"  
 to="yurastashchyshyn@gmail.com" from="javatestlogger@gmail.com"  
 smtpHost="smtp.gmail.com" smtpPort="465"  
 smtpUsername="javatestlogger@gmail.com" smtpPassword="fqcy tttx blxq luqj"  
 smtpProtocol="smtps" smtpDebug="true">  
 <PatternLayout pattern="%d{yyyy-MM-dd HH:mm:ss} [%t] %-5level %logger{36} - %msg%n"/>  
 <Filters>  
 <ThresholdFilter level="ERROR" onMatch="ACCEPT" onMismatch="DENY"/>  
 </Filters>  
 </SMTP>  
 </Appenders>  
  
 <Loggers>  
 <Root level="info">  
 <AppenderRef ref="ConsoleAppender"/>  
 <AppenderRef ref="FileAppender"/>  
 <AppenderRef ref="EmailAppender"/>  
 </Root>  
 </Loggers>  
</Configuration>

**pom.xml**

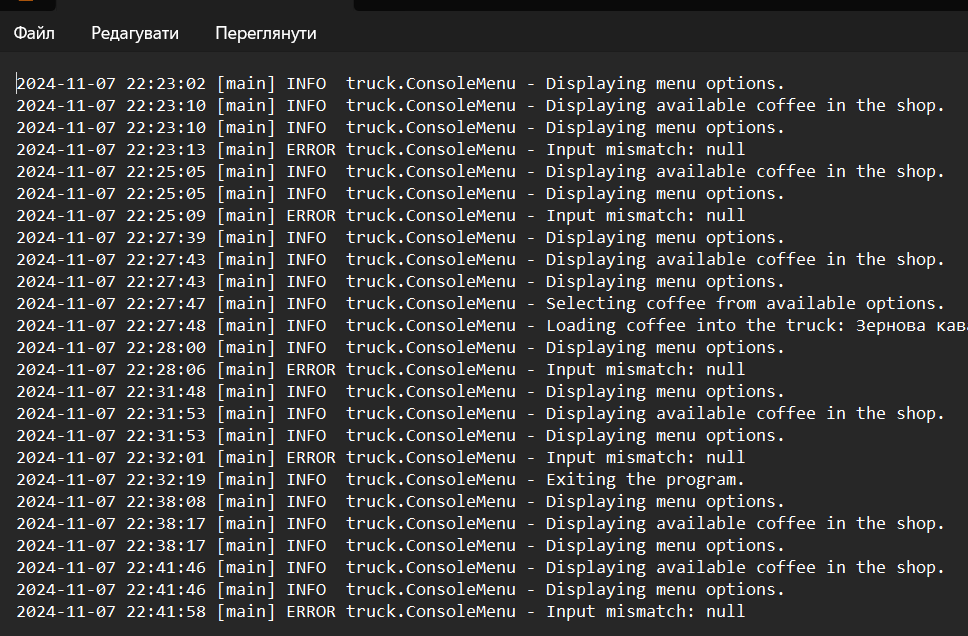
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>org.example</groupId>  
 <artifactId>Lab\_7\_Junit\_Test</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <dependencies>  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter-api</artifactId>  
 <version>5.8.2</version> <!-- Use the latest version -->  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.apache.logging.log4j</groupId>  
 <artifactId>log4j-api</artifactId>  
 <version>2.20.0</version> <!-- Use the latest stable version -->  
 </dependency>  
 <dependency>  
 <groupId>org.apache.logging.log4j</groupId>  
 <artifactId>log4j-core</artifactId>  
 <version>2.20.0</version>  
 </dependency>  
 <dependency>  
 <groupId>javax.mail</groupId>  
 <artifactId>javax.mail-api</artifactId>  
 <version>1.6.2</version>  
 </dependency>  
 <dependency>  
 <groupId>com.sun.mail</groupId>  
 <artifactId>javax.mail</artifactId>  
 <version>1.6.2</version>  
 </dependency>  
 <dependency>  
 <groupId>jakarta.activation</groupId>  
 <artifactId>jakarta.activation-api</artifactId>  
 <version>1.2.2</version> <!-- Updated to the correct dependency -->  
 </dependency>  
 </dependencies>  
  
 <properties>  
 <maven.compiler.source>22</maven.compiler.source>  
 <maven.compiler.target>22</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
</project>

package truck;  
  
import truck.command.\*;  
import truck.model.\*;  
import truck.model.Package;  
import org.apache.logging.log4j.LogManager;  
import org.apache.logging.log4j.Logger;  
  
import java.util.InputMismatchException;  
import java.util.List;  
import java.util.Scanner;  
  
public class ConsoleMenu {  
 private static final Logger *logger* = LogManager.*getLogger*(ConsoleMenu.class);  
  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 Truck truck = new Truck(2000);  
 Shop shop = new Shop();  
  
 LoadTruckCommand loadCommand = new LoadTruckCommand(truck);  
 Command searchCommand = new SearchCoffeeCommand(truck);  
 Command sortCommand = new SortCoffeeCommand(truck);  
 AddCoffeeCommand 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) {  
 *logger*.info("Displaying menu options.");  
 System.*out*.println();  
 System.*out*.println("Меню:");  
 System.*out*.println("1. Завантажити фургон кавою");  
 System.*out*.println("2. Показати відсортовану каву в фургоні");  
 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. Завнтажити дані про каву з файлу");  
 System.*out*.println("7. Вихід");  
  
 System.*out*.print("Оберіть опцію: ");  
  
 int choice = -1;  
 while (true) {  
 try {  
 choice = scanner.nextInt();  
 if (choice < 1 || choice > 7) {  
 System.*out*.println("Будь ласка, виберіть число від 1 до 7.");  
 } else {  
 break;  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next();  
 }  
 }  
  
 switch (choice) {  
 case 1:  
 Coffee load = *selectCoffee*(scanner, shop);  
 if (load != null) {  
 *logger*.info("Loading coffee into the truck: {}", load.getName());  
 System.*out*.print("Введіть суму на яку хочете зробити замовлення: ");  
 int sum = scanner.nextInt();  
 int quantity = (int) (sum / load.getPrice());  
 if (quantity > load.getQuantity()) {  
 *logger*.warn("Not enough coffee in the shop: {}", load.getName());  
 System.*out*.println("Недостатньо кави в магазині");  
 break;  
 }  
 loadCommand.setCoffee(load, quantity);  
 loadCommand.execute();  
 } else {  
 *logger*.error("Coffee selection failed.");  
 System.*out*.println("Помилка: Кава не вибрана.");  
 }  
 break;  
 case 2:  
 *logger*.info("Sorting coffee in the truck.");  
 carrier.sortCoffee();  
 break;  
 case 3:  
 *logger*.info("Searching coffee.");  
 carrier.searchCoffee();  
 break;  
 case 4:  
 Coffee coffee = *createCoffee*(scanner);  
 if (coffee != null) {  
 *logger*.info("Adding new coffee to the shop: {}", coffee.getName());  
 addCoffeeCommand.setCoffee(coffee);  
 seller.addCoffee();  
 }  
 break;  
 case 5:  
 *logger*.info("Displaying available coffee in the shop.");  
 seller.showCoffee();  
 break;  
 case 6:  
 *logger*.info("Loading coffee data from file.");  
 shop.loadCoffeesFromFile("NotCoffees.txt");  
 break;  
 case 7:  
 *logger*.info("Exiting the program.");  
 exitCommand.execute();  
 break;  
 default:  
 *logger*.warn("Invalid choice entered: {}", choice);  
 System.*out*.println("Невірний вибір, спробуйте ще раз.");  
 }  
 }  
  
 scanner.close();  
 *logger*.info("Program terminated.");  
 }  
  
private static Coffee selectCoffee(Scanner scanner, Shop shop) {  
 *logger*.info("Selecting coffee from available options.");  
 System.*out*.println("Доступна кава в магазині:");  
 List<Coffee> availableCoffee = shop.getAvailableCoffee();  
  
 if (availableCoffee.isEmpty()) {  
 *logger*.warn("No coffee available in the shop.");  
 System.*out*.println("Кава відсутня.");  
 return null;  
 }  
  
 // Відображення кави з індексами для вибору  
 for (int i = 0; i < availableCoffee.size(); i++) {  
 System.*out*.printf("%d. %s%n", i + 1, availableCoffee.get(i));  
 }  
  
 int choice = -1;  
 while (true) {  
 System.*out*.print("Оберіть номер кави для завантаження у фургон: ");  
 try {  
 choice = scanner.nextInt();  
 if (choice > 0 && choice <= availableCoffee.size()) {  
 return availableCoffee.get(choice - 1);  
 } else {  
 *logger*.warn("Invalid coffee choice: {}", choice);  
 System.*out*.println("Невірний вибір. Будь ласка, виберіть номер від 1 до " + availableCoffee.size() + ".");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next();  
 }  
 }  
 }  
  
 private static Coffee createCoffee(Scanner scanner) {  
 *logger*.info("Starting coffee creation process.");  
 System.*out*.println("Виберіть тип кави: ");  
 System.*out*.println("1. Grain Coffee");  
 System.*out*.println("2. Ground Coffee");  
 System.*out*.println("3. Instant Coffee");  
  
 int typeChoice = -1;  
 while (true) {  
 try {  
 typeChoice = scanner.nextInt();  
 if (typeChoice < 1 || typeChoice > 3) {  
 System.*out*.println("Невірний вибір. Будь ласка, виберіть тип кави від 1 до 3.");  
 *logger*.warn("Invalid coffee type selection: {}", typeChoice);  
 } else {  
 break; // Вихід з циклу, якщо введено коректне число  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 scanner.nextLine(); // Очищення буфера після введення числа  
  
 System.*out*.print("Введіть назву кави: ");  
 String name = scanner.nextLine();  
  
 double weight = -1;  
 while (weight < 0) {  
 System.*out*.print("Введіть вагу кави (в грамах): ");  
 try {  
 weight = scanner.nextDouble();  
 if (weight < 0) {  
 *logger*.warn("Invalid weight input: {}", weight);  
 System.*out*.println("Вага не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну вагу.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double price = -1;  
 while (price < 0) {  
 System.*out*.print("Введіть ціну кави (в грн): ");  
 try {  
 price = scanner.nextDouble();  
 if (price < 0) {  
 *logger*.warn("Invalid price input: {}", price);  
 System.*out*.println("Ціна не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну ціну.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double volume = -1;  
 while (volume < 0) {  
 System.*out*.print("Введіть об'єм кави (в літрах): ");  
 try {  
 volume = scanner.nextDouble();  
 if (volume < 0) {  
 *logger*.warn("Invalid volume input: {}", volume);  
 System.*out*.println("Об'єм не може бути негативним. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректний об'єм.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 // Вибір типу упаковки  
 System.*out*.println("Виберіть тип упаковки: ");  
 System.*out*.println("1. Банка");  
 System.*out*.println("2. Пакет");  
  
 int packagingChoice = -1;  
 while (true) {  
 try {  
 packagingChoice = scanner.nextInt();  
 if (packagingChoice < 1 || packagingChoice > 2) {  
 *logger*.warn("Invalid packaging choice: {}", packagingChoice);  
 System.*out*.println("Невірний вибір типу упаковки. Будь ласка, виберіть 1 або 2.");  
 } else {  
 break; // Вихід з циклу, якщо введено коректне число  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректне число.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 String packagingMaterial = (packagingChoice == 1) ? "Банка" : "Пакет";  
  
 double packageWeight = -1;  
 while (packageWeight < 0) {  
 System.*out*.print("Введіть вагу упаковки (в грамах): ");  
 try {  
 packageWeight = scanner.nextDouble();  
 if (packageWeight < 0) {  
 *logger*.warn("Invalid package weight input: {}", packageWeight);  
 System.*out*.println("Вага упаковки не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну вагу упаковки.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 double packageVolume = -1;  
 while (packageVolume < 0) {  
 System.*out*.print("Введіть об'єм упаковки (в літрах): ");  
 try {  
 packageVolume = scanner.nextDouble();  
 if (packageVolume < 0) {  
 *logger*.warn("Invalid package volume input: {}", packageVolume);  
 System.*out*.println("Об'єм упаковки не може бути негативним. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректний об'єм упаковки.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 int quantity = -1;  
 while (quantity < 0) {  
 System.*out*.print("Введіть кількість упаковок: ");  
 try {  
 quantity = scanner.nextInt();  
 if (quantity < 0) {  
 *logger*.warn("Invalid quantity input: {}", quantity);  
 System.*out*.println("Кількість не може бути негативною. Спробуйте ще раз.");  
 }  
 } catch (InputMismatchException e) {  
 *logger*.error("Input mismatch: {}", e.getMessage());  
 System.*out*.println("Помилка: Введіть коректну кількість упаковок.");  
 scanner.next(); // Очищення буфера  
 }  
 }  
  
 Package packaging = new Package(packagingMaterial, packageWeight, packageVolume);  
  
 Coffee coffee = null;  
 switch (typeChoice) {  
 case 1:  
 coffee = new GrainCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 case 2:  
 coffee = new GroundCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 case 3:  
 coffee = new InstantCoffee(name, weight, price, volume, packaging, quantity);  
 break;  
 default:  
 *logger*.error("Invalid coffee type choice: {}", typeChoice);  
 System.*out*.println("Невірний вибір типу кави.");  
 }  
  
 if (coffee != null) {  
 *logger*.info("Coffee created: {}", coffee);  
 }  
  
 return coffee;  
 }  
}

**Помилки надіслані на пошту**



**Дані логу збережені у файл**



**Висновок**

В ході цієї лабораторної роботи я навчився додавати логгер з логуванням основних дій та виняткових ситуацій. Зберіг дані логу у файлі, а критичні помилки – додатково надіслав на email.

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