**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **ІКНІ**

Кафедра **ПЗ**

### ЗВІТ

до лабораторної роботи № 2

**На тему:** *“Робота з колекціями в Java”*

**З дисципліни:** *“Кросплатформне програмування”*

**Лектор:**

доцент каф. ПЗ

Дяконюк Л. М.

**Виконав:**

ст. гр. ПЗ-35

Сточанський М.А.

**Прийняв:**

асист. каф. ПЗ

Шкраб Р.Р.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2020 р.

∑= \_\_\_\_ .

Львів – 2020

**Тема роботи:** Робота з колекціями в Java.

**Мета роботи:** Освоїти навики роботи з колекціями мови програмування Java.

**Теоретичні відомості**

Колекції в мові програмування Java поділяються на 3 основні типи:

* List – список впорядкованих елементів та може зберігати дублікати.
* Queue – список, який працює за принципом FIFO (first-in-first-out).
* Set – невпорядкований набір даних, який тримає тільки унікальні об’єкти.

****

**Індивідуальне завдання**

Задано інформацію про продукти, яка включає назву, дату виготовлення, кінцеву дату споживання та ціну.

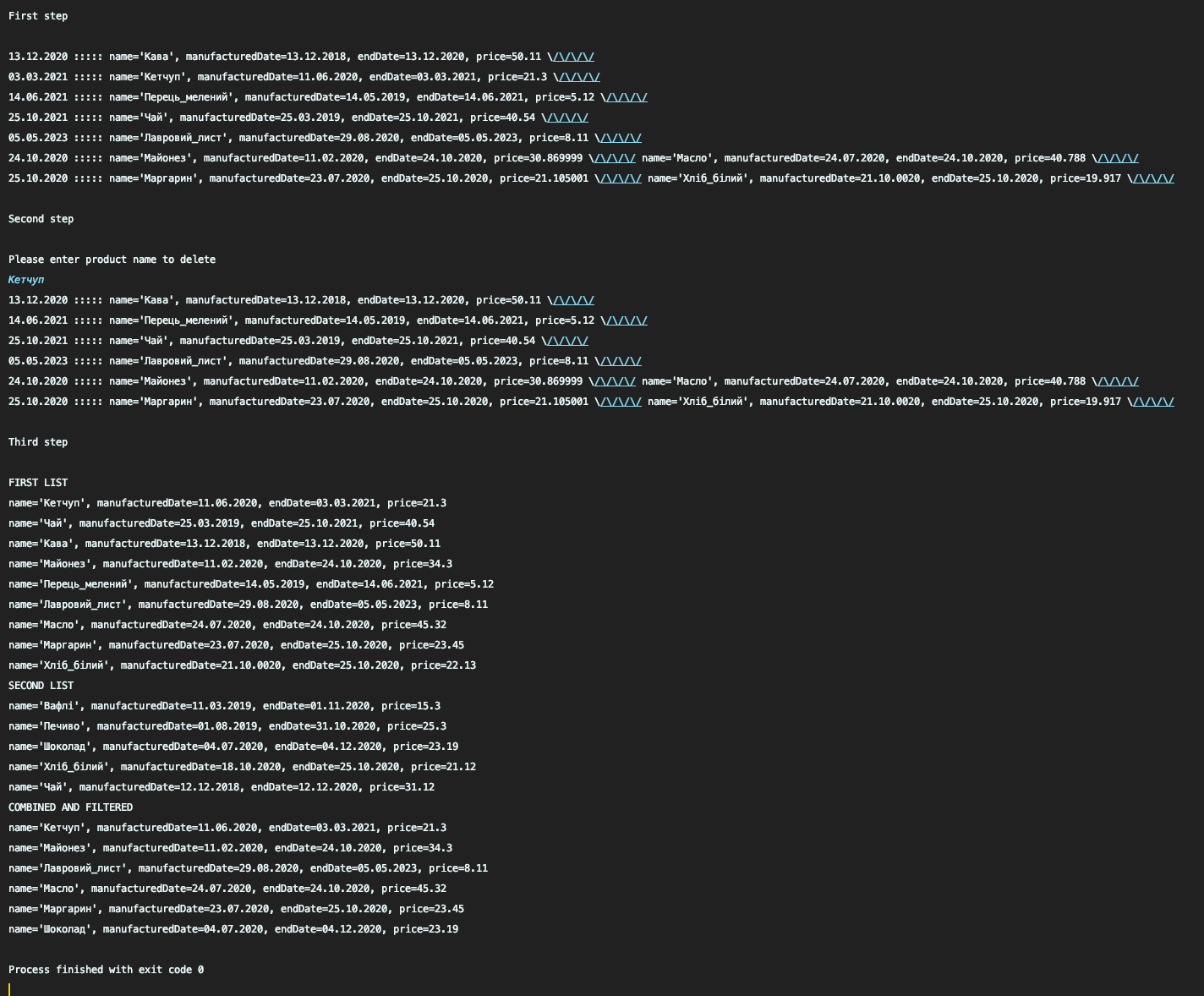
1. Здійснити переоцінку всіх продуктів на 10 відсотків, кінцева дата споживання яких є через 3 дні від біжучої дати. Видрукувати таблицю, в якій в першій колонці розташовані кінцеві дати, в другій – колекція всіх продуктів з заданою кінцевою датою.
2. Вилучити з попередньої таблиці всі продукти з заданою назвою.
3. З 2 різних файлів зчитати 2 вихідні набори інформації про продукти. Визначити спільну колекцію унікальних за назвою продуктів, які є тільки в першому, або тільки в другому наборі, вилучивши з неї елементи дата виготовлення яких є не в біжучому році.

**Код програми**

**Executor.java**

*package* execution;  
  
*import* data.dateFormatter.DateFormatter;  
*import* data.products.ProductsRepository;  
*import* data.products.ProductsRepositoryImpl;  
*import* data.products.models.Product;  
  
*import* java.util.\*;  
*import* java.util.stream.Collectors;  
*import* java.util.stream.Stream;  
  
*public class* Executor {  
  
 *private final* ProductsRepository repository = *new* ProductsRepositoryImpl();  
  
 *private final* DateFormatter datesFormatter = DateFormatter.getInstance();  
  
 *private final* Printer printer = Printer.getInstance();  
 *private final* AppScanner scanner = *new* AppScanner();  
  
 *public void* execute() {  
 printer.printStep("First step");  
 Map<Date, List<Product>> map = performFirstStep();  
 printer.printMap(map);  
  
 printer.printStep("Second step");  
 performSecondStep(map);  
 printer.printMap(map);  
  
 printer.printStep("Third step");  
 List<Product> products = performThirdStep();  
 printer.printList(products, "Combined and filtered");  
 }  
  
 *private* Map<Date, List<Product>> performFirstStep() {  
 List<Product> products = repository.getProductsFromFirstFile();  
  
 products.forEach(product -> {  
 *long* currentTime = System.currentTimeMillis();  
 *if* (datesFormatter.daysBetweenTimestamps(product.getEndDate().getTime(), currentTime) < 3) {  
 product.setPrice((*float*) (product.getPrice() - product.getPrice() \* 0.1));  
 }  
 });  
  
 *return* products.stream().collect(Collectors.groupingBy(Product::getEndDate));  
 }  
  
 *private void* performSecondStep(Map<Date, List<Product>> input) {  
  
 printer.printLine("Please enter product name to delete");  
  
 String nameToDelete = scanner.getLine();  
  
 input.forEach((date, products) -> products.removeIf(product -> product.getName().equals(nameToDelete)));  
  
 List<Date> keysToRemove = input.entrySet().stream()  
 .filter(entry -> entry.getValue().isEmpty())  
 .map(Map.Entry::getKey)  
 .collect(Collectors.toList());  
  
 keysToRemove.forEach(input::remove);  
 }  
  
 *private* List<Product> performThirdStep() {  
 List<Product> firstList = repository.getProductsFromFirstFile();  
 List<Product> secondList = repository.getProductsFromSecondFile();  
  
 printer.printList(firstList, "First list");  
 printer.printList(secondList, "Second list");  
  
 Calendar calendar = Calendar.getInstance();  
 *int* currentYear = calendar.get(Calendar.YEAR);  
  
 *return* Stream.concat(firstList.stream(), secondList.stream())  
 .filter(item -> !(listContainsItem(firstList, item) && listContainsItem(secondList, item)))  
 .filter(item -> {  
 calendar.setTime(item.getManufacturedDate());  
 *return* calendar.get(Calendar.YEAR) == currentYear;  
 }  
 ).collect(Collectors.toList());  
  
 }  
 *private boolean* listContainsItem(List<Product> list, Product product) {  
 *return* list.stream().anyMatch(item -> item.compareTo(product) == 0);  
 }  
}

**Протокол виконання**

****

**Висновок**

У результаті виконання лабораторної роботи було розроблено програму, яка опрацьовує різні колекції мови Java, а саме перетворює список в мапу, видаляє задані елементи з мапи та об’єднує 2 списки різних продуктів, виключаючи ті, які повторюються та фільтруючи результат.