Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

ЛАБОРАТОРНА РОБОТА № 8

з дисципліни «Основи програмування»

Тема: Вкладені та внутрішні класи

Виконали:

Студентки групи ІА-31

Горлач Д., Макасєєва М.,

Соколова П.

Перевірив:

Степанов А.

Київ-2023

Тема: Вкладені та внутрішні класи

Мета: Ознайомитись з наданими нам матеріалами, пригадати інформацію надану на лекціях з даної теми, також навчившись з минулих лабораторних робіт правильно використовувати знання та реалізовувати за допомогою них завдання, виконати поставлену нам задачу. А саме, скористатись новими знаннями та реалізувати за допомогою них подане завдання.

Хід роботи:

У цій роботі використали класи з лабораторної роботи № 8 з першого семестру. Для того, щоб відсортувати за допомогою методу Arrays.sort(array, comparator), отримали масив класів Product із списку Product і передали в метод Arrays.sort першим параметром, а другим параметром передали лямбда вираз компаратора, і порівнювали поле name між двома продуктами за допомогою методу compareTo у стрінгів. Результатом є мінус один, якщо перший об’єкт String лексикографічно передує рядку другого об'єкту, якщо навпаки, то повертає один, а якщо вони однакові - повертає нуль. Так само реалізуємо компаратор внутрішнім класом в класі Product і реалізуємо таку ж саму логіку по перевірці поля name у двох об'єктів. І передаємо цей компаратор в конструктор колекції TreeSet. Після чого додаємо весь список продуктів в колексію TreeSet. Для перевірки, що масив і TreeSet відсортовані однаково, викликаємо метод toArray у TreeSet, щоб отримати його елементи в вигляді масиву і використовуємо метод Arrays.equals(array1, array2) і передаємо туди відсортований масив і масив який отримали від TreeSet.

1. Пригадати як використовувати вкладені (nested) та внутрішні (inner) класи.
2. Для класів свого варіанту з л/р №8 першого семестру створити компаратори для того щоб була можливість сортувати об'єкти цих класів за допомогою Arrays.sort(T[] a, Comparator<? super T> c) та зберігати у колекції TreeSet<T>. Потрібно реалізовувати принаймні два компаратори: один як статичний вкладений клас, а інший як анонімний клас. Продемонструвати використання обох компараторів (відсортувати масив об'єктів та/або зберегти об'єкти в колекції TreeSet).

Рис. 1 завдання

1. Повторити теоретичні відомості
2. Проаналізувати предметну область завдання свого варіанту (табл.1)
3. Розробити базовий клас (відповідно до завдання можливо абстрактний клас або інтерфейс), клас-нащадок, а також допоміжні класи та/або інтерфейси за необхідністю. Відповідно до предметної області завдання передбачити відповідні методи бізнес-логіки, а також конструктори, сетери та/або гетери, методи equals() та toString(). Продемонструвати використання:
    - this;
    - super;
    - перевантаження (overloading) та заміщення(overriding) методів;
    - перевантаження (overloading) конструкторів.
4. Відповісти на контрольні запитання

9	Товар	Продовольчий товар	mutable

Рис. 2 завдання минулого сесместру та наш варіант

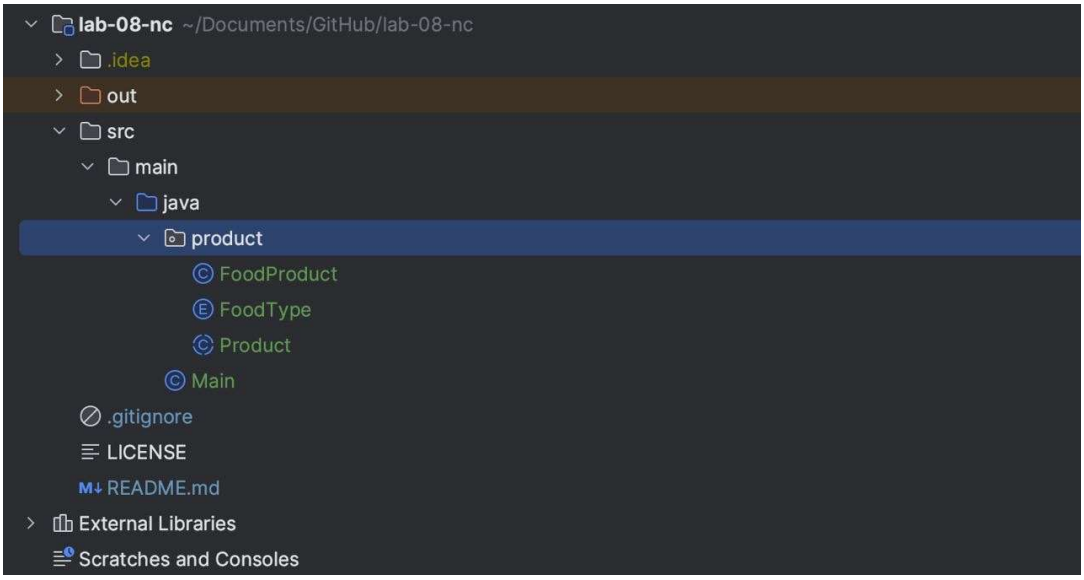

Рис 3. Структура програми

README.md | Main.java | .gitignore | FoodProduct.java | Product.java

```java
import product.FoodProduct;
import product.FoodType;
import product.Product;

import java.util.Arrays;
import java.util.List;
import java.util.Set;
import java.util.TreeSet;

new *
public class Main {

    new *
    public static void main(String[] args) {
        List<Product> products = List.of(
                new FoodProduct( name: "apple", quantity: 1, price: 0.5, calories: 10, FoodType.FRUIT),
                new FoodProduct( name: "pear", quantity: 1, price: 0.5, calories: 10, FoodType.FRUIT),
                new FoodProduct( name: "kiwi", quantity: 1, price: 0.5, calories: 10, FoodType.FRUIT),
                new FoodProduct( name: "blueberry", quantity: 1, price: 0.5, calories: 10, FoodType.FRUIT)
        );

        Product[] productArray = products.toArray(new Product[]{});
        Arrays.sort(productArray, (p1, p2) -> p1.getName().compareTo(p2.getName()));

        Set<Product> productSet = new TreeSet<>(new Product.ProductComparator());
        productSet.addAll(products);

        System.out.println(Arrays.equals(productArray, productSet.toArray()));
    }
}
```

Рис. 4 клас Main

```
true

Process finished with exit code 0
```

Рис. 5 output

Висновок:

Суть в тому, що якщо використовувати компаратор для відсортування масиву і використовувати той же компаратор в TreeSet, то порядок відсортованих елементів буде однаковий.