Міністерство освіти і науки України

Національний університет "Львівська політехніка"

Звіт з лабораторної роботи №6

з курсу “Кросплатформні засоби програмування”

Параметризовне програмування

Виконав: студент гр. KI-306

Шаповал Віталій

Прийняв: к.т.н. Олексів М.В.

Львів 2024 р.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Параметризоване програмування є аналогом шаблонів у C++. Воно полягає у написанні коду, що можна багаторазово застосовувати з об'єктами різних класів. Користувачів параметризованого програмування можна поділити на 3 рівні кваліфікації:

1. ті, що користуються готовими класами;
2. ті, що користуються готовими класами і вміють виправляти помилки, що виникають при їх використанні;
3. ті, що пишуть власні параметризовані класи.

Для успішного застосування параметризованого програмування слід навчитися розуміти помилки, що генерує середовище при компіляції програми, що можуть стосуватися, наприклад, неоднозначності визначення спільного суперкласу для всіх переданих об'єктів. З іншої сторони необхідно передбачити захист від присвоєння об'єктів параметризованого класу, що містять об'єкти підкласу об'єктам параметризованого класу, що містять об'єкти суперкласу і дозволити зворотні присвоєння. Для вирішення цієї проблеми у мові Java введено так звані підстановочні типи. Це далеко не всі «підводні камені», що виникають при застосуванні параметризованого програмування.

Варіант № 28 Антресолі

Завдання:

1. Створити параметризований клас, що реалізує предметну область задану варіантом. Клас має містити мінімум 4 методи опрацювання даних включаючи розміщення та виймання елементів. Парні варіанти реалізують пошук мінімального елементу, непарні – максимального. Написати на мові Java та налагодити програму-драйвер для розробленого класу, яка мстить мінімум 2 різні класи екземпляри яких розмішуються у екземплярі розробленого класу-контейнеру. Програма має розміщуватися в пакеті Група.Прізвище.Lab6 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.
2. Автоматично згенерувати документацію до розробленого пакету.
3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
5. Дати відповідь на контрольні запитання.

Код програми:

Файл: ./ce306/shapoval/lab6/Cabinet.java:

```
package ce306.shapoval.lab6;

import java.util.*;

/**
 * The <code>Cabinet</code> class represents a cabinet with a generic type.
 * It allows adding, removing, and searching for items.
 *
 * @param <T> the type of items stored in the cabinet
 */
public class Cabinet<T extends Item> {
    private List<T> items;

    /**
     * Constructor to initialize the cabinet.
     */
    public Cabinet() {
        items = new ArrayList<>();
    }

    /**
     * Adds an item to the cabinet.
     * @param item The item to be added
     */
    public void addItem(T item) {
        items.add(item);
    }

    /**
     * Removes an item from the cabinet.
     * @param item The item to be removed
     * @return <code>true</code> if the item was removed, <code>false</code>
otherwise
     */
    public boolean removeItem(T item) {
        return items.remove(item);
    }

    /**
     * Searches for the minimum item in the cabinet (for even variants).
     * @return The minimum item, or <code>null</code> if the cabinet is empty
     */
    public T findMin() {
        if (items.isEmpty()) {
            return null;
        }
        T min = items.get(0);
        for (T item : items) {
            if (item.compareTo(min) < 0) {
                min = item;
            }
```

```
        }
        return min;
    }

    /**
     * Returns the list of items in the cabinet.
     * @return List of items
     */
    public List<T> getItems() {
        return items;
    }
}
```

Файл: ./ce306/shapoval/lab6/Item.java:

```
package ce306.shapoval.lab6;

/**
 * The <code>Item</code> class represents an item with a name and volume.
 */
public class Item implements Comparable<Item> {
    private String name;
    private double volume;

    /**
     * Constructor to initialize an item with a name and volume.
     *
     * @param name   The name of the item
     * @param volume The volume of the item
     */
    public Item(String name, double volume) {
        this.name = name;
        this.volume = volume;
    }

    /**
     * Gets the name of the item.
     *
     * @return The name of the item
     */
    public String getName() {
        return name;
    }

    /**
     * Gets the volume of the item.
     *
     * @return The volume of the item
     */
    public double getVolume() {
        return volume;
    }

    /**
```

```
     * Compares this item with another item based on volume.
     *
     * @param other The other item to compare with
     * @return A negative integer, zero, or a positive integer as this item
     *         is less than, equal to, or greater than the specified item
     */
    @Override
    public int compareTo(Item other) {
        return Double.compare(this.volume, other.volume);
    }

    /**
     * Returns a string representation of the item.
     *
     * @return A string representation of the item
     */
    @Override
    public String toString() {
        return String.format("Item{name='%s', volume=%.2f}", name, volume);
    }
}
```

Файл: ./CabinetApp.java:

```
import ce306.shapoval.lab6.*;
import java.util.Scanner;

/**
 * The <code>CabinetApp</code> class serves as a driver program to demonstrate
the functionality
 * of the <code>Cabinet</code> class. It allows users to interact with the
cabinet to add,
 * remove, and find items based on their volume.
 */
public class CabinetApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Cabinet<Item> cabinet = new Cabinet<>(); // Create a cabinet to store
items

        boolean exit = false;

        while (!exit) {
            // Displaying menu options
            System.out.println("===== Cabinet Menu =====");
            System.out.println("1. Add item");
            System.out.println("2. Remove item");
            System.out.println("3. Find minimum item");
            System.out.println("4. Display all items");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline character
```

```
            switch (choice) {
                case 1:
                    // Adding an item
                    System.out.print("Enter item name: ");
                    String name = scanner.nextLine();
                    System.out.print("Enter item volume: ");
                    double volume = scanner.nextDouble();
                    Item newItem = new Item(name, volume);
                    cabinet.addItem(newItem);
                    System.out.println("Item added: " + newItem);
                    break;

                case 2:
                    // Removing an item
                    System.out.print("Enter item name to remove: ");
                    String itemNameToRemove = scanner.nextLine();
                    Item itemToRemove = null;
                    for (Item item : cabinet.getItems()) {
                        if (item.getName().equalsIgnoreCase(itemNameToRemove)) {
                            itemToRemove = item;
                            break;
                        }
                    }
                    if (itemToRemove != null &&
cabinet.removeItem(itemToRemove)) {
                        System.out.println("Item removed: " + itemToRemove);
                    } else {
                        System.out.println("Item not found.");
                    }
                    break;

                case 3:
                    // Finding the minimum item
                    Item minItem = cabinet.findMin();
                    if (minItem != null) {
                        System.out.println("Minimum item: " + minItem);
                    } else {
                        System.out.println("Cabinet is empty.");
                    }
                    break;

                case 4:
                    // Displaying all items
                    System.out.println("Items in the cabinet:");
                    for (Item item : cabinet.getItems()) {
                        System.out.println(item);
                    }
                    break;

                case 5:
                    // Exiting
                    System.out.println("Exiting.");
```

```
                    exit = true;
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }

        scanner.close();
    }
}
```

Виконання програми:

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 4

Items in the cabinet:

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 1

Enter item name: Petro

Enter item volume: 0.08

Item added: Item{name='Petro', volume=0.08}

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 4

Items in the cabinet:

Item{name='Petro', volume=0.08}

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 1

Enter item name: Ivan

Enter item volume: 100

Item added: Item{name='Ivan', volume=100.00}

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 4

Items in the cabinet:

Item{name='Petro', volume=0.08}

Item{name='Ivan', volume=100.00}

===== Cabinet Menu =====

1. Add item

2. Remove item

3. Find minimum item

4. Display all items

5. Exit

Enter your choice: 5

Exiting.

Висновок

У цій лабораторній роботі я створив параметризований клас для реалізації предметної області відповідно до варіанту завдання. Клас містить щонайменше чотири методи обробки даних, включаючи розміщення та виймання елементів. У моєму рішенні я реалізував пошук максимального елемента, що відповідає непарному варіанту, а також створив програму-драйвер, яка демонструє роботу розробленого класу з принаймні двома різними класами. Програма була організована в пакеті Група.Прізвище.Lab6, що відповідає вимогам до структури. Я додав коментарі до коду, що дозволили автоматично згенерувати документацію для пакета. Завантаживши код на GitHub відповідно до методичних вказівок, я закріпив свої навички роботи з системою контролю версій. Ця лабораторна робота дозволила мені покращити знання роботи з параметризованими класами та вмінь у написанні документованого коду.