Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



Звіт

з лабораторної роботи № 6

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «Параметризоване програмування»

**Виконав:**

студент групи КІ-307

Трофименко Є.Е.

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Мета роботи:** оволодіти навиками параметризованого програмування мовою Java.

**Завдання (варіант № 22)**

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.



**Вихідний код програми**

*Main.java*

package ki307.Trofymenko.Lab6;

/\*\*

\* Основний клас програми

\* **@version** 1.0

\*/

public class Main

{

/\*\*

\* Конструктор за замовченням

\*/

public Main() {}

/\*\*

\* Вхідний метод програми

\* **@param** args визначення методу main

\*/

public static void main(String[] args)

{

Suitcase<? super Item> suitcase = new Suitcase<>();

suitcase.AddItem(new Clothes(29, "Sweeter", "M"));

suitcase.AddItem(new Clothes(32, "Hoodie", "XL"));

suitcase.AddItem(new Shoes(65, "Laser"));

suitcase.AddItem(new Shoes(40, "Textil"));

suitcase.AddItem(new Clothes(21, "T-Shirt", "L"));

suitcase.AddItem(new Shoes(15, "Coton"));

suitcase.AddItem(new Clothes(45, "Hoodie", "L"));

var min= suitcase.FindMin();

System.***out***.print("\nMin element is: ");

min.PrintInfo();

}

}

*Shoes.java*

package ki307.Trofymenko.Lab6;

public class Shoes implements Item

{

private double volume;

private String material;

public Shoes(double volume, String material)

{

this.volume = volume;

this.material = material;

}

public void setVolume(double volume) {

this.volume = volume;

}

public String getMaterial() {

return material;

}

public void setMaterial(String material) {

this.material = material;

}

*@Override*

public double getVolume()

{

return volume;

}

*@Override*

public void PrintInfo()

{

System.***out***.println("Clothes: volume - " + volume + "; material - " + material + ";");

}

}

*Clothes.java*

package ki307.Trofymenko.Lab6;

public class Clothes implements Item

{

private double volume;

private String typeOfClothes;

private String size;

public Clothes(double volume, String typeOfClothes, String size)

{

this.volume = volume;

this.typeOfClothes = typeOfClothes;

this.size = size;

}

public void setVolume(double volume)

{

this.volume = volume;

}

public String getTypeOfClothes()

{

return typeOfClothes;

}

public void setTypeOfClothes(String typeOfClothes)

{

this.typeOfClothes = typeOfClothes;

}

public String getSize()

{

return size;

}

public void setSize(String size)

{

this.size = size;

}

*@Override*

public double getVolume()

{

return volume;

}

*@Override*

public void PrintInfo()

{

System.***out***.println("Clothes: volume - " + volume + "; type of clothes - " + typeOfClothes + "; size - " + size + ";");

}

}

*Item.java*

package ki307.Trofymenko.Lab6;

public interface Item

{

double getVolume();

void PrintInfo();

}

*Suitcase.java*

package ki307.Trofymenko.Lab6;

import java.util.ArrayList;

import java.util.List;

public class Suitcase<T extends Item>

{

private final List<T> arr;

public Suitcase()

{

arr = new ArrayList<T>();

}

public T FindMin()

{

if (arr.isEmpty())

{

return null;

}

else

{

T min = arr.get(0);

for (int i = 1; i < arr.size(); i++)

{

if (arr.get(i).getVolume() < min.getVolume())

{

min = arr.get(i);

}

}

return min;

}

}

public void AddItem(T item)

{

arr.add(item);

var index = arr.indexOf(item);

System.***out***.print("Element was added: index - " + index + "; ");

item.PrintInfo();

}

public void DeleteItem(int i)

{

System.***out***.print("Element on index " + i + " was deleted: ");

arr.get(i).PrintInfo();

arr.remove(i);

}

public void PrintAllItems()

{

for(T el : arr)

{

el.PrintInfo();

}

}

}

**Результат виконання програми**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Фрагмент згенерованої документації**

Изображение выглядит как текст, программное обеспечение, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, чек, снимок экрана

Автоматически созданное описание

**Відповіді на контрольні запитання**

1. Дайте визначення терміну «параметризоване програмування».

*Параметризоване програмування – програмування, суть якого полягає у написанні коду, що можна багаторазово застосовувати з об’єктами різних класів.*

1. Розкрийте синтаксис визначення простого параметризованого класу.

*Синтаксис оголошення параметризованого класу:*

*[public] class НазваКласу <параметризованийТип{,параметризованийТип}>*

*{...}*

1. Розкрийте синтаксис створення об’єкту параметризованого класу.

*Синтаксис створення об’єкту параметризованого класу:*

*НазваКласу < перелікТипів > = new НазваКласу < перелікТипів > (параметри);*

1. Розкрийте синтаксис визначення параметризованого методу.

*Синтаксис оголошення параметризованого методу:*

*Модифікатори <параметризованийТип{,параметризованийТип}> типПовернення назваМетоду(параметри);*

1. Розкрийте синтаксис виклику параметризованого методу.

*Синтаксис виклику параметризованого методу:*

*(НазваКласу|НазваОб’єкту).[<перелікТипів>]НазваМетоду(параметри);*

1. Яку роль відіграє встановлення обмежень для змінних типів?

*Наприклад, може бути ситуація, коли метод у процесі роботи викликає з-під об’єкта параметризованого типу метод, що визначається у деякому інтерфейсі. У такому випадку немає ніякої гарантії, що цей метод буде реалізований у кожному класі, що передається через змінну типу. Щоб вирішити цю проблему у мові Java можна задати обмеження на множину можливих типів, що можуть бути підставлені замість параметризованого типу.*

1. Як встановити обмеження для змінних типів?

*Для цього після змінної типу слід використати ключове слово extends і вказати один суперклас, або довільну кількість інтерфейсів (через знак &), від яких має походити реальний тип, що підставляється замість параметризованого типу.*

1. Розкрийте правила спадкування параметризованих типів.
2. *Всі класи, що утворені з одного і того ж параметризованого класу з використанням різних значень змінних типів є незалежними.*
3. *Завжди можна перетворити параметризований клас у «сирий» клас.*

*3) Параметризовані класи можуть розширювати або реалізовувати інші параметризовані класи.*

1. Яке призначення підстановочних типів?

*Підстановочні типи були введені у мову Java для збільшення гнучкості жорсткої існуючої системи параметризованих типів.*

1. Застосування підстановочних типів.

*Підстановочні типи застосовуються у вигляді параметру типу, що передається у трикутних дужках при утворені реального типу з параметризованого типу, наприклад, у методі main.*

**Висновок**

На данній лабораторній роботі я оволодів навиками параметризованого програмування мовою Java. Також ознайомився з параметризованими методами, встановленням обмежень для змінних типів та підстановчими типами. Створив параметризований клас, що реалізує предметну область задану варіантом.