

Міністерство освіти і науки України
Національний університет „Львівська політехніка”
Кафедра ЕОМ



Звіт

до лабораторної роботи № 6
з дисципліни: «Кросплатформні засоби програмування»
«Параметризоване програмування»
Варіант - 1

Виконав:
Студент групи КІ-306
Баран В. М.
Прийняв:
Іванов Ю. С.

Львів 2023

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. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
4. Дати відповідь на контрольні запитання.

Варіант завдання:

1. Масив

Код програми:

File App.java

```
package KI34.Vishchur.lab7;

public class App {
    /**
     * @param args
     * @throws Exception
     */
    public static void main(String[] args) throws Exception
    {
        // Ініціалізація масиву
        Array<? super Shape> mall = new Array();
        // Додаємо фігури
        mall.add(new Sphere(1000, 100,100, "Sphere1"));
        mall.add(new Cube(1000, 10,10, "Cube1" ));
        // Виводимо
        mall.showAll();
        // Виводимо першу
        System.out.println(mall.get(0).getName());
        // Сукупний об'єм
        System.out.println(mall.getTotalVolume());
        // Найбільша фігура
        System.out.println(mall.getLargestFigure().getName());
    }
}
```

```
}
```

File Array.java

```
package KI34.Vishchur.lab7;

import java.util.ArrayList;
/**
 * Class Array
 * @version 1.0
 */
public class Array<T extends Shape> {
    private ArrayList<T> storage = new ArrayList();
    /**
     * Method returns an item by the index
     * @param index Element index
     * @throws Exception
     */
    public T get(int index) throws Exception
    {
        try{
            return storage.get(index);
        }
        catch (Exception e)
        {
            throw e;
        }
    }
    /**
     * Method adds an item
     * @param item Item to be added
     * @throws Exception
     */
    public void add(T item) throws Exception
    {
        try {
            storage.add(item);
        }
        catch (Exception e)
        {
            throw e;
        }
    }
    /**
     * Method shows storage content
     */
    public void showAll()
```

```

    {
        for(int i = 0; i < storage.size(); i++)
        {
            try{
                storage.get(i).printData();
            }
            catch (Exception e)
            {
                System.out.println(e);
            }
        }
    }
}

/**
 * Method returns total volume
 */
public int getTotalVolume()
{
    int result = 0;
    for(int i = 0; i < storage.size(); i++)
    {
        result += storage.get(i).getVolume();
    }
    return result;
}

/**
 * Method returns the biggest item
 */
public T getLargestFigure()
{
    int largestIdx = 0;

    for(int i = 0; i < storage.size(); i++)
    {
        if(i == 0)
        {
            continue;
        }
        if(storage.get(i).getVolume() >
storage.get(largestIdx).getVolume())
        {
            largestIdx = i;
        }
    }
    return storage.get(largestIdx);
}
}

```

File Shape.java

```
package KI34.Vishchur.lab7;

public class Shape {
    private int volume;
    private int width;
    private int height;
    private String name;
    /**
     * Constructor
     * @param volume
     * @param width
     * @param height
     * @param name
     */
    public Shape(int volume, int width, int height, String
name)
    {
        this.volume = volume;
        this.width = width;
        this.height = height;
        this.name = name;
    }
    /**
     * Method prints shape data
     */
    public void printData()
    {
        System.out.println("Name: " + name);
        System.out.println("Volume: " + volume);
        System.out.println("Width: " + width);
        System.out.println("Height: " + height);
    }
    /**
     * Method returns shape height
     */
    public int getHeight(){ return height; }
    /**
     * Method returns shape volume
     */
    public int getVolume(){ return volume; }
    /**
     * Method returns shape width
     */
    public int getWidth() { return width; }
```

```

    /**
     * Method returns shape name
     */
    public String getName() { return name; }
}

```

File Cube.java

```

package KI34.Vishchur.lab7;

public class Cube extends Shape {
    /**
     * Constructor
     * @param volume
     * @param width
     * @param height
     * @param name
     */
    public Cube(int volume, int width, int height, String
name)
    {
        super(volume, width, height, name);
    }
}

```

File Sphere.java

```

package KI34.Vishchur.lab7;

public class Sphere extends Shape {
    /**
     * Constructor
     * @param volume
     * @param width
     * @param height
     * @param name
     */
    public Sphere(int volume, int width, int height, String
name)
    {
        super(volume, width, height, name);
    }
}

```

Результат виконання роботи:

```
Name: Sphere1
Volume: 1000
Width: 100
Height: 100
Name: Cube1
Volume: 1000
Width: 10
Height: 10
Sphere1
2000
Sphere1
```

Фрагмент згенерованої документації:

PACKAGE

CLASS

TREE

INDEX

HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD

DETAIL: FIELD | CONSTR | METHOD

SEARCH

Package org.example

Class App

java.lang.Object

org.example.App

public class App

extends Object

Constructor Summary

Constructors

Constructor	Description
App()	

Method Summary

All MethodsStatic MethodsConcrete Methods
























Modifier and Type	Method	Description
static void	main(String[] args)	

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

App

	index-files	13.11.2023 11:57	Папка файлів	
	legal	13.11.2023 11:57	Папка файлів	
	org	13.11.2023 11:57	Папка файлів	
	resources	13.11.2023 11:57	Папка файлів	
	script-dir	13.11.2023 11:57	Папка файлів	
	allclasses-index.html	13.11.2023 11:57	Chrome HTML Do...	4 КБ
	allpackages-index.html	13.11.2023 11:57	Chrome HTML Do...	3 КБ
	copy.svg	13.11.2023 11:57	Microsoft Edge HT...	1 КБ
	element-list	13.11.2023 11:57	Файл	1 КБ
	help-doc.html	13.11.2023 11:57	Chrome HTML Do...	9 КБ
	index.html	13.11.2023 11:57	Chrome HTML Do...	2 КБ
	link.svg	13.11.2023 11:57	Microsoft Edge HT...	2 КБ
	member-search-index.js	13.11.2023 11:57	JavaScript Source ...	2 КБ
	module-search-index.js	13.11.2023 11:57	JavaScript Source ...	1 КБ
	overview-tree.html	13.11.2023 11:57	Chrome HTML Do...	4 КБ
	package-search-index.js	13.11.2023 11:57	JavaScript Source ...	1 КБ
	script.js	13.11.2023 11:57	JavaScript Source ...	9 КБ
	search.html	13.11.2023 11:57	Chrome HTML Do...	4 КБ
	search.js	13.11.2023 11:57	JavaScript Source ...	17 КБ
	search-page.js	13.11.2023 11:57	JavaScript Source ...	11 КБ
	stylesheet.css	13.11.2023 11:57	Каскадна таблиц...	32 КБ
	tag-search-index.js	13.11.2023 11:57	JavaScript Source ...	1 КБ
	type-search-index.js	13.11.2023 11:57	JavaScript Source ...	1 КБ

Відповіді на контрольні запитання

1. Дайте визначення терміну «параметризоване програмування».

- це підхід до програмування, що дозволяє створювати класи і методи, які можна використовувати з різними типами даних, надаючи більшу гнучкість і безпеку типів у програмах.

2. Розкрийте синтаксис визначення простого параметризованого класу.

```
public class НазваКласу<параметризованийТип> {  
    // Тіло класу }
```

3. Розкрийте синтаксис створення об'єкту параметризованого класу. -

```
НазваКласу<перелікТипів> зміннаКласу = new  
НазваКласу<перелікТипів>(параметри);
```

4. Розкрийте синтаксис визначення параметризованого методу.

```
- public <параметризованийТип> типПовернення назваМетоду(параметри) {  
    // Тіло методу }
```

5. Розкрийте синтаксис виклику параметризованого методу.

```
- (НазваКласу|НазваОб'єкту).<перелікТипів>назваМетоду(параметри);
```

6. Яку роль відіграє встановлення обмежень для змінних типів?

- дозволяє заборонити використання деяких типів або вимагати, щоб тип підставлений за замовчуванням був підкласом або реалізував певний інтерфейс.

7. Як встановити обмеження для змінних типів?

- за допомогою ключового слова `extends` для суперкласу або інтерфейсу, від яких має походити реальний тип.

8. Розкрийте правила спадкування параметризованих типів.

- о - Всі класи, створені з параметризованого класу, незалежні один від одного.
- о - Зазвичай немає залежності між класами, створеними з різними параметрами типів.

9. Яке призначення підстановочних типів?

- використовуються для забезпечення безпеки типів при використанні параметризованих класів та методів. Вони дозволяють визначити, які типи можна використовувати замість параметризованих типів.

10. Застосування підставних типів.

- <?> (unbounded wildcard) дозволяє читати об'єкти з колекції без змінення її.
- - <? extends Тип> (bounded wildcard) дозволяє читати об'єкти з колекції, але забороняє додавання в неї нових об'єктів.
- - <? super Тип> (lower bounded wildcard) дозволяє додавати об'єкти в колекцію, але забороняє їх читання.

Висновок

У ході виконання даної лабораторної роботи, я отримала важливі навички параметризованого програмування мовою Java. Ознайомилась з різними аспектами мови, такими як використання параметрів у методах, створення та використання класів та інтерфейсів.