Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ



**Звіт**

З лабораторної роботи №6

Варіант№8

З дисципліни: «Кросплатформні засоби програмування»

На тему: «ПАРАМЕТРИЗОВАНЕ ПРОГРАМУВАННЯ»

Виконав: ст. гр. КІ-304

Кісіль В. І.

Прийняв:

Олексів М.В.

Львів 2023

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

**Теоретичні відомості**

Параметризоване програмування є аналогом шаблонів у С++. 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.

**Визначення простого параметризованого класу**

Параметризований клас – це клас з однією або більше змінними типу.

Синтаксис оголошення параметризованого класу:

[public] class НазваКласу {…}

Іменувати змінні параметризованих типів прийнято великими літерами. У бібліотеці Java використовується літера Е для позначення типу колекції, К і V – для типів ключа і значення таблиці, Т, U, S та сусідні літери – для позначення довільних типів.

Приклад оголошення параметризованого класу:

class GenericClass { public GenericClass(T first, U second)

{ this.first = first;

this.second = second; }

public void setFirst(T first)

{ this.first = first; }

public T getFirst()

{ return first; }

...

private T first; private U second; }

Тут T і U – це змінні параметризованих типів, що використовуються по всьому тілу класу для специфікації типу повернення методів, типів полів і локальних змінних. При створенні об’єкту параметризованого класу замість них підставляються реальні типи, що визначаються в трикутних дужках у місці створення об’єкту параметризованого класу.

Синтаксис створення об’єкту параметризованого класу:

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

Приклад створення об’єкту параметризованого класу:

GenericClass obj = new GenericClass ();

**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

9

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.

**Варіант завдання:**

**8. Коробка**

**Код програми:**

Book.java

package ki\_304.Kisil.Lab6;

public class Book implements IObj

{

private String bookName;

private int usedTimes;

private int size;

public Book(String bName, int bUsed, int bSize)

{

bookName = bName;

usedTimes = bUsed;

size = bSize;

}

public int getSize()

{

return size;

}

public int compareTo(IObj p)

{

Integer s = size;

return s.compareTo(p.getSize());

}

public void print()

{

System.out.print("Book: " + bookName + ", Times used: " + usedTimes +

", Book Size: " + size + ";\n");

}

}

Box.java

package ki\_304.Kisil.Lab6;

import java.util.\*;

import java.util.ArrayList;

public class Box <T extends IObj>

{

private ArrayList<T> arr;

public Box()

{

arr = new ArrayList<T>();

}

public void clear()

{

arr = new ArrayList<T>();

}

public T findMin()

{

if (!arr.isEmpty())

{

T min = arr.get(0);

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

{

if ( arr.get(i).compareTo(min) < 0 )

min = arr.get(i);

}

return min;

}

return null;

}

public void sortSize()

{

Collections.sort(arr);

}

public void AddData(T data)

{

arr.add(data);

System.out.print("Element added: ");

data.print();

}

public void printContent()

{for(int i=0; i<arr.size();i++)

{IObj res = arr.get(i);

res.print();}

}

public void DeleteData(int i)

{

arr.remove(i);

}

}

BoxDriver

package ki\_304.Kisil.Lab6;

public class BoxDriver {

public static void main(String[] args)

{

Box <? super IObj> myBox = new Box <IObj>();

myBox.AddData(new Notebook("Weekend" , 32568741));

myBox.clear();

myBox.AddData(new Notebook("People" , 2547814));

myBox.AddData(new Book("Windows", 0, 852124325));

myBox.AddData(new Book("JRE" , 0, 67894257));

IObj res = myBox.findMin();

System.out.print("The smallest object in Box is: \n");

res.print();

System.out.print("The list of objects in Box: \n");

myBox.printContent();

System.out.print("The list of objects in Box: \n");

myBox.sortSize();

myBox.printContent();

}

}

IObj

package ki\_304.Kisil.Lab6;

public interface IObj extends Comparable<IObj>

{

public int getSize();

public void print();

}

Notebook

package ki\_304.Kisil.Lab6;

public class Notebook implements IObj

{

private String notebookName;

private int notebookSize;

public Notebook(String nName, int nSize)

{

notebookName = nName;

notebookSize = nSize;

}

public int getSize()

{

return notebookSize;

}

public int compareTo(IObj p)

{

Integer s = notebookSize;

return s.compareTo(p.getSize());

}

public void print()

{

System.out.print("Notebook Name: " + notebookName + ", Photo File Size: " + notebookSize + ";\n");

}

}

**Результат роботи програми:**

Element added: Notebook Name: Weekend, Photo File Size: 32568741;

Element added: Notebook Name: People, Photo File Size: 2547814;

Element added: Book: Windows, Times used: 0, Book Size: 852124325;

Element added: Book: JRE, Times used: 0, Book Size: 67894257;

The smallest object in Box is:

Notebook Name: People, Photo File Size: 2547814;

The list of objects in Box:

Notebook Name: People, Photo File Size: 2547814;

Book: Windows, Times used: 0, Book Size: 852124325;

Book: JRE, Times used: 0, Book Size: 67894257;

The list of objects in Box:

Notebook Name: People, Photo File Size: 2547814;

Book: JRE, Times used: 0, Book Size: 67894257;

Book: Windows, Times used: 0, Book Size: 852124325;

**Висновок:** Я оволодів навиками параметризованого програмування мовою Java.