МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра системи штучного інтелекту

Лабораторна робота 2

з дисципліни

“Об'єктно орієнтоване програмування”

Виконав:

студент групи КН-109

Гладун Ярослав

Викладач:

Мочурад Л. І.

Львів - 2018 р.

Тема: Розробка власних контейнерів. Ітератори. Серіалізація/десеріалізація об’єктів. Бібліотека класів користувача

Мета:

- Набуття навичок розробки власних контейнерів.
- Використання ітераторів.
- Тривале зберігання та відновлення стану об’єктів.
- Ознайомлення з принципами серіалізації/десеріалізації об’єктів.
- Використання бібліотек класів користувача.

Теоретичні відомості:

Клас визначає абстрактні характеристики деякої сутності, включаючи характеристики самої сутності (її атрибути або властивості) та дії, які вона здатна виконувати (її поведінки, методи або можливості). Наприклад, клас Собака може характеризуватись рисами, притаманними всім собакам, зокрема: порода, колір хутра, здатність гавкати. Класи вносять модульність та структурованість в об'єктно-орієнтовану програму. Як правило, клас має бути зрозумілим для не-програмістів, що знаються на предметній області, що, у свою чергу, значить, що клас повинен мати значення в контексті. Також, код реалізації класу має бути досить самодостатнім. Властивості та методи класу, разом називаються його членами.

Об'єкт Окремий екземпляр класу (створюється після запуску програми і ініціалізації полів класу). Клас Собака відповідає всім собакам шляхом опису їхніх спільних рис; об'єкт Рекс є одним окремим собакою, окремим варіантом значень характеристик. Собака має хутро; Рекс має коричнево-біле хутро. Об'єкт Рекс є екземпляром (примірником) класу Собака. Сукупність значень атрибутів окремого об'єкта називається станом. На основі класу Собака можна, також, створити інший об'єкт Сірко, який відрізнятиметься від об'єкта Рекс своїм станом (наприклад кольором хутра). Обидва об'єкта (Сірко і Рекс) є екземплярами класу Собака.

Метод Можливості об'єкта. Оскільки Сірко — Собака, він може гавкати. Тому гавкати() є одним із методів об'єкта Сірко. Він може мати й інші методи, зокрема: місце(), або їсти(). В межах програми, використання

методу має впливати лише на один об'єкт; всі Собаки можуть гавкати, але треба щоб гавкав лише один окремий собака.

Варіант 5.

Умова: 1. Розробити клас-контейнер, що ітерується для збереження початкових даних Вашого варіанту завдання з попередньої роботи (Прикладні задачі. Список з 1-15 варіантів)у вигляді масиву рядківз можливістю додавання, видалення і зміни елементів. 2. В контейнері реалізувати та продемонструвати наступні методи: ○ String toString() повертає вміст контейнера у вигляді рядка; ○ void add(String string) додає вказаний елемент до кінця контейнеру; ○ void clear() видаляє всі елементи з контейнеру; ○ boolean remove(String string) видаляє перший випадок вказаного елемента з контейнера; ○ Object[] toArray() повертає масив, що містить всі елементи у контейнері; ○ int size() повертає кількість елементів у контейнері; ○ boolean contains(String string) повертає true, якщо контейнер містить вказаний елемент; ○ boolean containsAll(Container container) повертає true, якщо контейнер містить всі елементи з зазначеного у параметрах; ○ public Iterator iterator() повертає ітератор відповідно до Interface Iterable. 3. В класі ітератора відповідно до Interface Iterator реалізувати методи: ○ public boolean hasNext(); ○ public String next(); ○ public void remove(). 4. Продемонструвати роботу ітератора за допомогою циклів while и for each. 5. Забороняється використання контейнерів (колекцій) і алгоритмів з Java Collections Framework. 6. Реалізувати і продемонструвати тривале зберігання/відновлення розробленого контейнера за допомогою серіалізації/десеріалізації. 7. Обмінятися відкомпільованим (без початкового коду) службовим класом (Utility Class) рішення одного варіанту задачі (Прикладні задачі. Список з 1-15 варіантів)з сусіднім номером. 1 міняється з 2, 2 з 3, 3 з 4, 4 з 5 і т.д. Останній, 15 міняється з 1 варіантом і далі аналогічно. 8. Продемонструвати послідовну та вибіркову обробку елементів розробленого контейнера за допомогою власного і отриманого за обміном службового класу. 9. Реалізувати та продемонструвати порівняння,

сортування та пошук елементів у контейнері. 10.Розробити консольну програму та забезпечити діалоговий режим роботи з користувачем для демонстрації та тестування рішення.

Розв'язок:

```
package com.company;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.util.Scanner;

public class Main {

    private static final String filepath = "StringList.txt";

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);
        String choice;
        StringList stringList = new StringList();

        do {
            System.out.println("MENU:");
            System.out.println("a. Add to list.");
            System.out.println("b. Remove from list.");
            System.out.println("c. Print list.");
            System.out.println("d. Print number of strings.");
            System.out.println("e. Check contain element.");
            System.out.println("f. Clean list.");
            System.out.println("g. Write to file.");
            System.out.println("h. Read from file.");
            System.out.println("i. Sort.");

            System.out.println("j. Exit.");
            System.out.print("Do choice: ");

            choice = in.nextLine();

            switch (choice) {
```

```
                case "a":

System.out.println("\n=======================================")
;
                    System.out.print("Enter string: ");
                    stringList.add(in.nextLine());

System.out.println("=======================================\n")
;
                    break;
                case "b":

System.out.println("\n=======================================")
;
                    System.out.print("Enter string: ");
                    if (stringList.remove(in.nextLine())) {
                        System.out.println("String removed.");
                    } else {
                        System.out.println("List hasn't
string.");
                    }

System.out.println("=======================================\n")
;
                    break;
                case "c":

System.out.println("\n=======================================")
;
                    for (String str :
                            stringList) {
                        System.out.println("||    " + str);
                    }

System.out.println("=======================================\n")
;
                    break;
                case "d":

System.out.println("\n=======================================")
;
```

```
                    System.out.printf("List size = %d;\n",
stringList.size());

System.out.println("=======================================\n")
;
                    break;
                case "e":

System.out.println("\n=======================================")
;
                    if (stringList.contains(in.nextLine())) {
                        System.out.println("List contains.");
                    } else {
                        System.out.println("List doesn't
contain.");
                    }

System.out.println("=======================================\n")
;
                    break;
                case "f":

System.out.println("\n=======================================")
;
                    stringList.clear();
                    System.out.println("List cleared.");

System.out.println("=======================================\n")
;
                    break;
                case "g":

System.out.println("\n=======================================")
;
                    WriteObjectToFile(filepath, stringList);
                    System.out.println("List wrote.");

System.out.println("=======================================\n")
;
                    break;
                case "h":
```

```
System.out.println("\n=======================================")
;
                    stringList = (StringList)
ReadObjectFromFile(filepath);
                    System.out.println("List read.");

System.out.println("=======================================\n")
;
                case "i":

System.out.println("\n=======================================")
;
                    stringList.sort();
                    System.out.println("List sorted.");

System.out.println("=======================================\n")
;
                    break;
                case "j":
                    break;
                default:
                    System.out.println("Menu hasn't variant.");
                    break;
            }


        } while (!choice.equals("j"));

    }

    public static void WriteObjectToFile(String filepath,
Object serObj) {

        try {

            FileOutputStream fileOut = new
FileOutputStream(filepath);
            ObjectOutputStream objectOut = new
ObjectOutputStream(fileOut);
            objectOut.writeObject(serObj);
            objectOut.close();
```

```
            System.out.println("The Object  was succesfully
written to a file");

        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public static Object ReadObjectFromFile(String filepath) {

        try {

            FileInputStream fileIn = new
FileInputStream(filepath);
            ObjectInputStream objectIn = new
ObjectInputStream(fileIn);

            Object obj = objectIn.readObject();

            System.out.println("The Object has been read from
the file");
            objectIn.close();
            return obj;

        } catch (Exception ex) {
            ex.printStackTrace();
            return null;
        }
    }
}
```

Висновок: Я навчився записувати та зчитувати з файла власно створені об’єкти.