**Лабораторна робота №6**

**Серіалізація/десеріалізація об'єктів. Бібліотека класів користувача**

**Мета:**Тривале зберігання та відновлення стану об'єктів. Ознайомлення з принципами серіалізації/десеріалізації об'єктів. Використання бібліотек класів користувача.

**1 ВИМОГИ**

1. Реалізувати і продемонструвати тривале зберігання/відновлення раніше розробленого контейнера за допомогою серіалізації/десеріалізації.
2. Обмінятися відкомпільованим (без початкового коду) службовим класом (UtilityClass) рішення задачі л.р. №3 з іншим студентом (визначає викладач).
3. Продемонструвати послідовну та вибіркову обробку елементів розробленого контейнера за допомогою власного і отриманого за обміном службового класу.
4. Реалізувати та продемонструвати порівняння, сортування та пошук елементів у контейнері.
5. Розробити консольну програму та забезпечити діалоговий режим роботи з користувачем для демонстрації та тестування рішення.

**1.1Розробник**

- П.І.Б:Наймитенко С.І.

- Група: КІТ-119а

- Варіант 15

**1.3 Задача**

Ввести текст. Текст розбити на речення. Для кожного речення вивести слова, у яких перша та остання літери співпадають. Результат вивести у вигляді таблиці.

**2 ОПИС ПРОГРАМИ**

**2.1**Було використано наступні засоби:

* Iterator<String> iterator = MyContainer.getIterator() – Ітератор;
* Scannersc = newScanner(System.in) – для введення обраних опцій користувачем з клавіатури;
* FileOutputStreamfs = newFileOutputStream("Serial.ser");ObjectOutputStreamos = newObjectOutputStream(fs);os.writeObject(mc);os.close() – серіалізація;
* FileInputStreamfis = newFileInputStream("Serial.ser");ObjectInputStreamois = newObjectInputStream(fis);MyContainervar = (MyContainer) ois.readObject();ois.close() – десеріалізація;

**2.2 Ієрархія та структура класів**

Було створено 3 класита використано клас розроблений у л.р. №3:

* publicclassMyContainer – клас, що реалізує методи контейнеру.
* privateclassMyIterator – клас, що реалізує методи ітератора.
* public class Main – містить лише метод main.

**Важливі фрагменти програми**

**Клас Container**

package ua.khpi.oop.naimytenko06;

import java.io.Serializable;

import java.util.Iterator;

public class Container implements Serializable {

private String [] container;

private int size;

public String toString() // повертає вміст контейнера у вигляді рядка;

{

String str = "";

for (String string : container) {

str += string + " ";

}

return str;

}

public void add(String str) //додає вказаний елемент до кінця контейнеру;

{

int size = container.length;

String [] new\_container = new String[size+1];

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

new\_container[i]=container[i];

}

new\_container[size]=str;

size++;

container = new\_container;

}

public void clear() //видаляє всі елементи з контейнеру;

{

for (int i = 0; i < container.length; i++) {

container[i]=null;

}

size =0;

}

public boolean remove(String str) // видаляє перший випадок вказаного елемента з контейнера;

{

boolean flag = false;

String [] new\_container = new String[size-1];

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

if(container[i].equals(str))

flag = true;

}

if(flag) {

for(int i=0,j=0;i<size;i++) {

if(container[i].equals(str))

i++;

new\_container[j]=container[i];

j++;

}

size--;

container = new\_container;

return flag;

}

else

{

return flag;

}

}

public String[] toArray() //повертає масив, що містить всі елементи у контейнері;

{

return container;

}

public int size() //повертає кількість елементів у контейнері;

{

return size;

}

public boolean containsAll(Container cont) //повертає true, якщо контейнер містить всі елементи з зазначеного у параметрах;

{

int count = 0;

for (int i = 0; i < container.length; i++) {

for (int j = 0; j < cont.container.length; j++) {

if(cont.container[j].equals(container[i]))

count++;

}

}

if(count == cont.container.length)

return true;

else

return false;

}

public boolean contains(String str) //повертає true, якщо контейнер містить вказаний елемент;

{

boolean flag = false;

for (String string : container) {

if(string.equals(str))

flag=true;

}

return flag;

}

public void Sort() {

String temp;

for(int a = 0; a < size - 1; a++) {

for(int b = a + 1; b < container.length; b++)

{

if(container[a].compareTo(container[b]) > 0)

{

temp = container[a];

container[a] = container[b];

container[b] = temp;

}

}

}

}

public Container(String... str) {

if(str.length!=0) {

size = str.length;

container = new String[size];

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

container[i]=str[i];

}

}

}

public Iterator<String> getIterator() {

return new My\_iterator<String>();

}

public class My\_iterator<String> implements Iterator {

int index;

@Override

public boolean hasNext() {

return index < size ? true : false;

}

@Override

public Object next() {

return container[index++];

}

/\*Method that removes from the underlying collection the last element returned by this iterator\*/

@Override

public void remove() {

Container.this.remove(container[--index]);

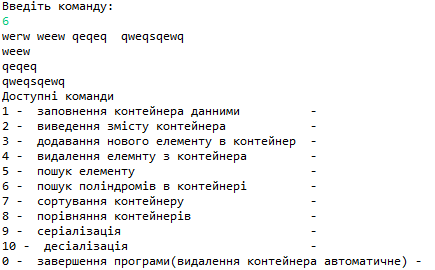
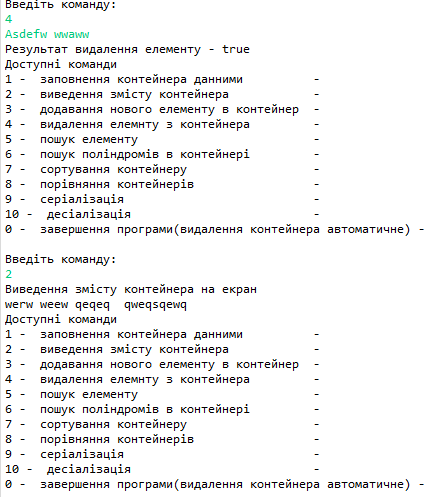
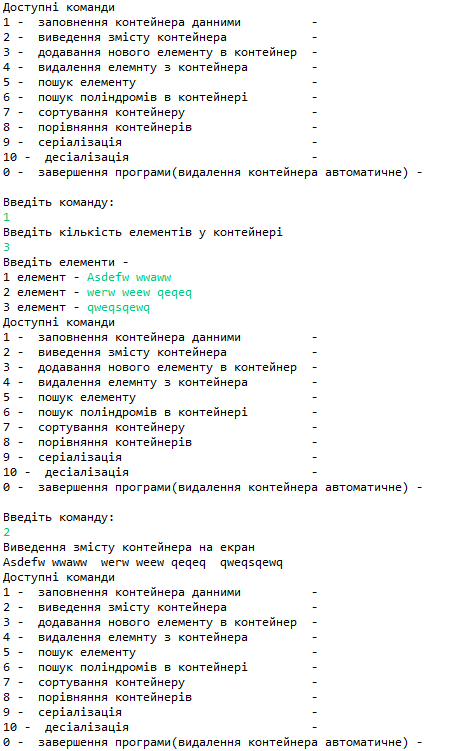
}

}

}

**3 ВИСНОВКИ**

Результат роботи програми:



**ВИСНОВКИ**

У результаті виконання лабораторної роботи було набуто навичок роботи з серіалізацією\десеріалізацією. з розробки бібліотеки класів користувача у середовищі JavaEclipse.