**Звіт**

**Лабораторна работа 6. Серіалізація/десеріалізація об'єктів. Бібліотека класів користувача**

**Мета роботи**:

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

**ВИМОГИ**

1. Реалізувати і продемонструвати тривале зберігання/відновлення раніше розробленого контейнера за допомогою серіалізації/десеріалізації.
2. Обмінятися відкомпільованим (без початкового коду) службовим класом (Utility Class) 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. **Розробник**: Каркуша Дмитро Андрійович, КІТ119-а, варіант №10.
6. **ОПИС ПРОГРАМИ**
   1. **Засоби ООП**: клас, метод класу, поле класу.
   2. **Ієрархія та структура класів:** один публічний клас Main та публічний клас Container, у полі якого знаходиться приватний класс MyIterator.
   3. **Важливі фрагменти програми:**

public class Container implements Serializable {

private String[] array;

private int size;

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

{

String newArray = "";

for (String string : array)

{

newArray += string + " ";

}

return newArray;

}

public void addElement(String string) //додає вказаний елемент до кінця контейнеру;

{

String newArray[] = new String[size + 1];

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

{

newArray[i] = array[i];

}

newArray[size] = string;

size++;

array = newArray;

}

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

{

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

array[i]=null;

}

size =0;

}

public boolean removeElement(String string) // видаляє перший випадок вказаного елемента з контейнера;

{

boolean flag = false;

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

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

if(array[i].equals(string))

flag = true;

}

if(flag) {

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

if(array[i].equals(string))

i++;

new\_array[j]=array[i];

j++;

}

size--;

array = new\_array;

return flag;

}

else

{

return flag;

}

}

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

{

return array;

}

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

{

return size;

}

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

{

int count = 0;

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

for (int j = 0; j < arr.array.length; j++) {

if(arr.array[j].equals(array[i]))

count++;

}

}

if(count == arr.array.length)

return true;

else

return false;

}

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

{

boolean flag = false;

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

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

flag=true;

}

return flag;

}

public Container(String... str) {

if(str.length!=0) {

size = str.length;

array = new String[size];

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

array[i]=str[i];

}

}

}

public void Sort() {

String temp;

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

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

{

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

{

temp = array[a];

array[a] = array[b];

array[b] = temp;

}

}

}

}

public Iterator<String> getIterator()

{

return new MyIterator<String>();

}

private class MyIterator<String> implements Iterator {

int index;

@Override

public boolean hasNext() {

return index < size ? true : false;

}

@Override

public Object next() {

return array[index++];

}

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

@Override

public void removeElement() {

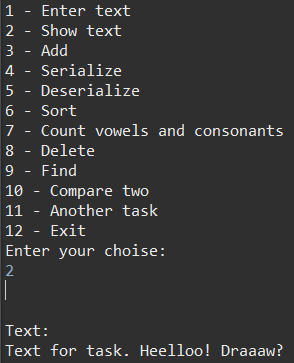
Container.this.remove(array[--index]);

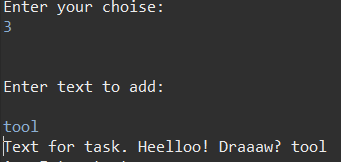
}

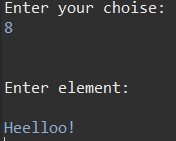
}

}

**Результат роботи програми.**









**Висновки**

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи с серіалізацією та десеріалізацією об’єктів .

Програма протестована, виконується без помилок.