Міністерство освіти і науки України

Національний технічний університет

«Харківський політехнічний інститут»

Кафедра «Обчислювальна техніка та програмування»

**ЗВІТ**

Про виконання лабораторної роботи № 5

«[Серіалізація/десеріалізація об'єктів. Бібліотека класів користувача](https://oop-khpi.gitlab.io/#tasks)»

Керівник: Жилін Володимир Анатолійович

Виконавець: студентка гр. КІТ-320 Шпилька Д. О.

Харків 2021

Тема роботи: [Серіалізація/десеріалізація об'єктів. Бібліотека класів користувача](https://oop-khpi.gitlab.io/#tasks)

Мета:

* 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](https://oop-khpi.gitlab.io/%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F/task05/) за допомогою [серіалізації/десеріалізації](https://docs.oracle.com/javase/8/docs/technotes/guides/serialization/index.html).
2. Обмінятися відкомпільованим (без початкового коду) службовим класом (Utility Class) рішення [задачі л.р. №3](https://oop-khpi.gitlab.io/%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F/task03/#_4) 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.

Код: **package** ua.khpi.oop.Shpylka06;

**import** java.io.\*;

**import** java.util.\*;

**class** main{

**public** **static** **void** main(String[] args){

Menu our\_menu = **new** Menu();

our\_menu.menu\_start();

}

}

**class** Menu{

String s;

**int** x, y;

**void** menu\_start() {

Scanner input = **new** Scanner(System.***in***);

Container cont = **new** Container();

**for** (;;){

System.***out***.println("Choose an option:" + "\n1) Input data" + "\n2) Show data" + "\n3) Make ALL calculations"

+ "\n4) Make INDEX calculations" + "\n5) Show array" + "\n6) Clear array" + "\n7) Contains your word"

+ "\n8) Delete your word" + "\n9) Serialize" + "\n10) Deserialize" + "\n11) Sort array" + "\n12) Compare 2 words in array by index"

+ "\n13) Show array size" + "\n14) Exit" + "Your choice:");

**int** choice = input.nextInt();

**if** (choice < 1 || choice >= 14){

System.***out***.println("\n>Wrong input!");

System.*exit*(1);

}

**else** **if** (choice == 1){

System.***out***.println("\n>Enter data:");

input.nextLine();

s = input.nextLine();

cont.add(s);

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 2){

System.***out***.println(s);

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 3){

cont.remover\_all();

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 4){

System.***out***.println("\n>Enter data:");

input.nextLine();

x = input.nextInt();

cont.remover\_index(x);

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 5){

System.***out***.println(cont.to\_string());

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 6){

cont.clear();

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 7){

System.***out***.println("\n>Enter data:");

input.nextLine();

s = input.nextLine();

System.***out***.println(cont.contains(s));

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 8){

System.***out***.println("\n>Enter data:");

input.nextLine();

s = input.nextLine();

**if** (cont.remove(s)){

System.***out***.println("\n>Done!");

} **else**{

System.***out***.println(">Not done!");

}

}

**else** **if** (choice == 9){

**try** {

cont.serialize();

} **catch** (IOException e) {

e.printStackTrace();

}

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 10){

**try** {

cont.deserialize();

} **catch** (ClassNotFoundException | IOException e) {

e.printStackTrace();

}

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 11){

cont.sort();

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 12){

System.***out***.println("\n>Enter first index: ");

input.nextLine();

x = input.nextInt();

System.***out***.println("\n>Enter second index:");

input.nextLine();

y = input.nextInt();

System.***out***.println(cont.compare\_strings(x, y));

System.***out***.println("\n>Done!");

}

**else** **if** (choice == 13){

System.***out***.println(cont.size());

System.***out***.println("\n>Done!");

}

}

}

}

**class** Helper{

**static** String remover(String S){

String result = S.trim().replaceAll("[^a-zA-Z-\\s-,.!?;:]", "");

result = result.replaceAll(" +", " ");

result = result.replaceAll("(?<=[,.!?;:])(?!$)", " ");

**return** result;

}

}

**class** Container **implements** Iterable<String> {

String[] our\_strings = {"Vol12vo", "BM W", "Fo $ rd", "Maz& da"};

String s;

@Override

**public** Iterator<String> iterator(){

Iterator<String> it = **new** Iterator<String>() {

**private** **int** currentIndex = 0;

@Override

**public** **boolean** hasNext() {

**return** currentIndex < our\_strings.length && our\_strings[currentIndex] != **null**;

}

@Override

**public** String next() {

**return** our\_strings[currentIndex++];

}

@Override

**public** **void** remove() {

**throw** **new** UnsupportedOperationException();

}

};

**return** it;

}

String to\_string(){

StringBuilder temp\_buffer = **new** StringBuilder();

**for** (String our\_string : our\_strings) {

temp\_buffer.append(our\_string);

temp\_buffer.append(',');

}

**return** temp\_buffer.toString();

}

**void** add(String string){

s = string;

**int** i;

**int** length = our\_strings.length;

String[] new\_arr = **new** String[length + 1];

**for** (i = 0; i < length; i++){

new\_arr[i] = our\_strings[i];

}

new\_arr[length] = string;

our\_strings = new\_arr;

}

**void** clear(){

our\_strings = **new** String[0];

}

Object[] to\_array(){

Object[] resulting\_object = **new** Object[our\_strings.length];

**int** i = 0;

**for** (String S : our\_strings){

resulting\_object[i] = S;

i++;

}

**return** resulting\_object;

}

**int** size(){

**return** our\_strings.length + 1;

}

**boolean** contains(String string){

**for** (String S : our\_strings){

**if** (Objects.*equals*(S, string)){

**return** **true**;

}

}

**return** **false**;

}

**boolean** contains\_all(Container container){

**return** **this** == container;

}

**boolean** remove(String string) {

**int** length = our\_strings.length;

String[] new\_arr = **new** String[length - 1];

**for** (**int** i = 0, k = 0; i < length; i++){

**if** (our\_strings[i].equals(string)) {

**continue**;

}

new\_arr[k++] = our\_strings[i];

}

our\_strings = new\_arr;

**return** **true**;

}

**void** remover\_all(){

**int** i;

**int** length = our\_strings.length;

String[] new\_arr = **new** String[length];

**for** (i = 0; i < length; i++){

our\_strings[i] = Helper.*remover*(our\_strings[i]);

new\_arr[i] = our\_strings[i];

}

our\_strings = new\_arr;

}

**void** remover\_index(**int** index){

**int** i;

**int** length = our\_strings.length;

String[] new\_arr = **new** String[length];

**for** (i = 0; i < length; i++){

**if** (i == index){

our\_strings[i] = Helper.*remover*(our\_strings[i]);

}

new\_arr[i] = our\_strings[i];

}

our\_strings = new\_arr;

}

**void** serialize() **throws** IOException {

String name = "Serialized.ser";

Object obj = **this**.to\_string();

FileOutputStream fileOut = **new** FileOutputStream(name);

ObjectOutputStream out = **new** ObjectOutputStream(fileOut);

out.writeObject(obj);

out.close();

fileOut.close();

System.***out***.println("\nData written to: " + name);

}

**void** deserialize() **throws** IOException, ClassNotFoundException {

FileInputStream fileIn = **new** FileInputStream("Serialized.ser");

ObjectInputStream in = **new** ObjectInputStream(fileIn);

Object obj = in.readObject();

System.***out***.println(obj.toString());

}

**void** sort(){

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

**for** (**int** j = i + 1; j < **this**.size(); j++){

**if**(our\_strings[i].toLowerCase().compareTo(our\_strings[j].toLowerCase()) > 0){

String temp = our\_strings[i];

our\_strings[i] = our\_strings[j];

our\_strings[j] = temp;

}

}

}

}

**boolean** compare\_strings(**int** index1, **int** index2){

**return** our\_strings[index1].equals(our\_strings[index2]);

}

}

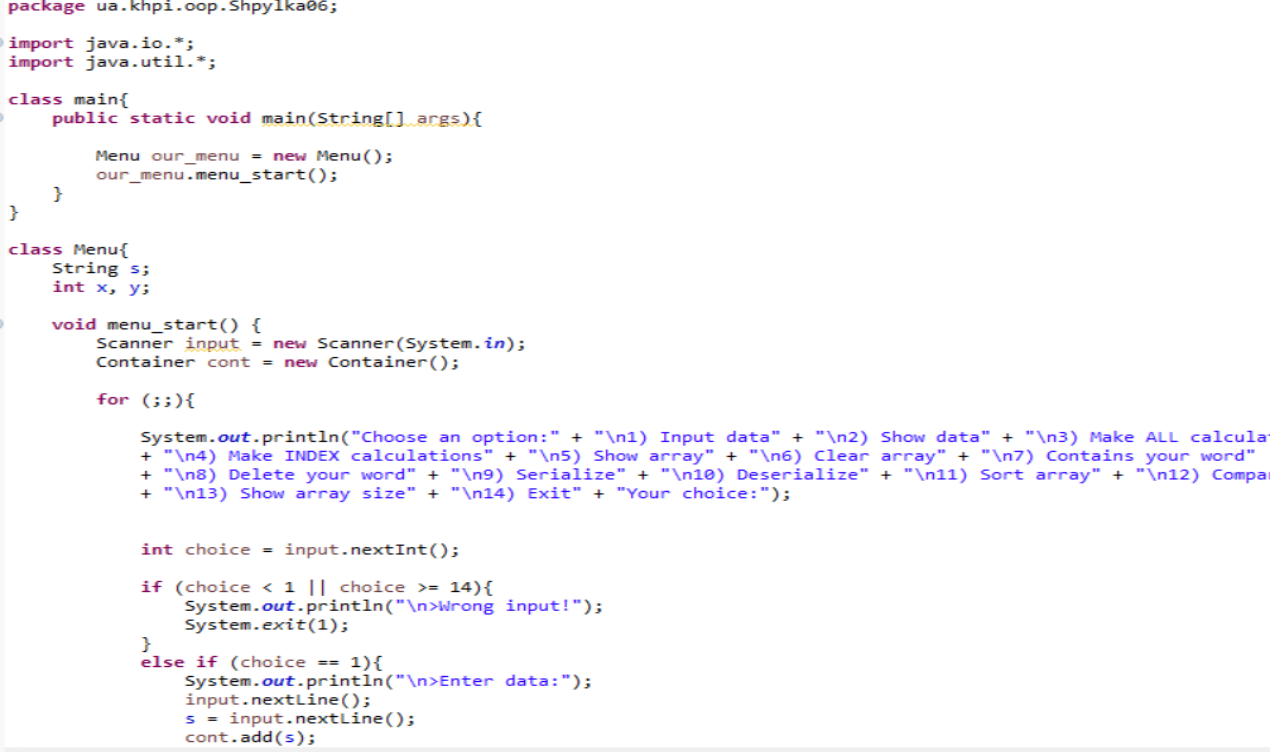


рис.1

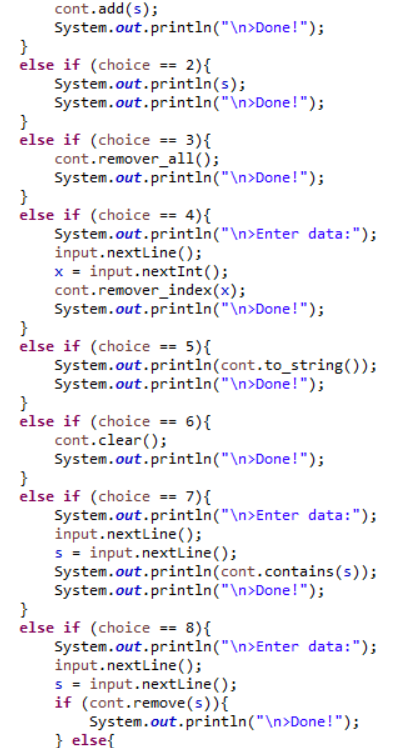


рис.2

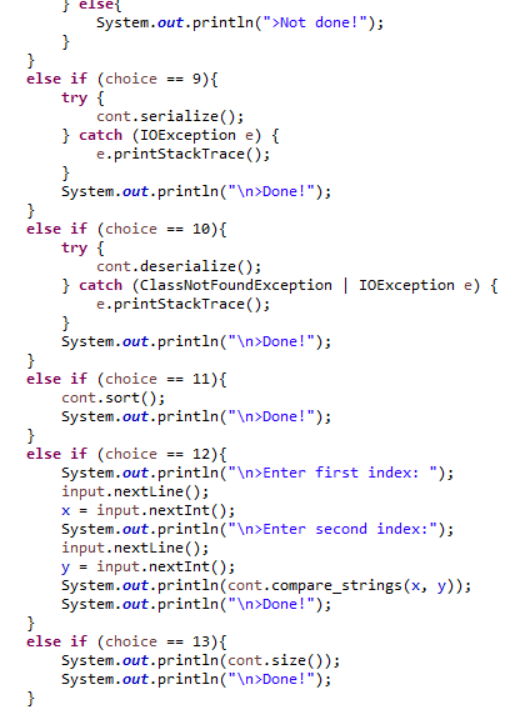


рис.3



рис.4

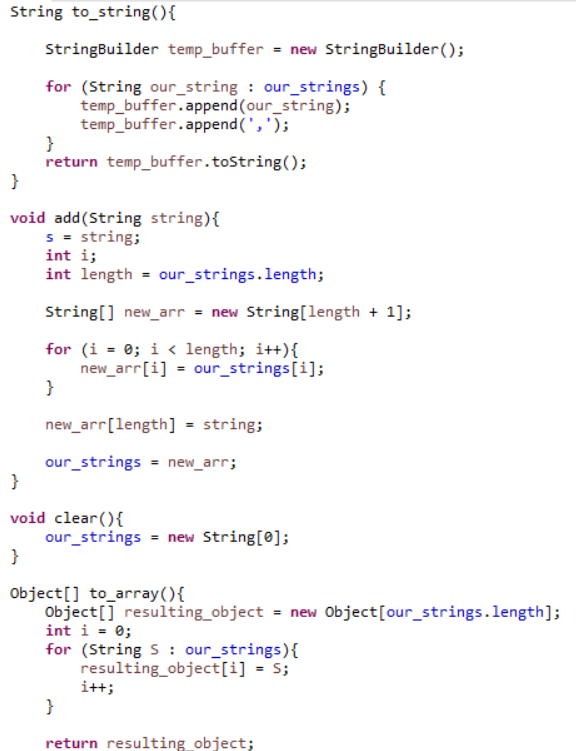
****

рис.5

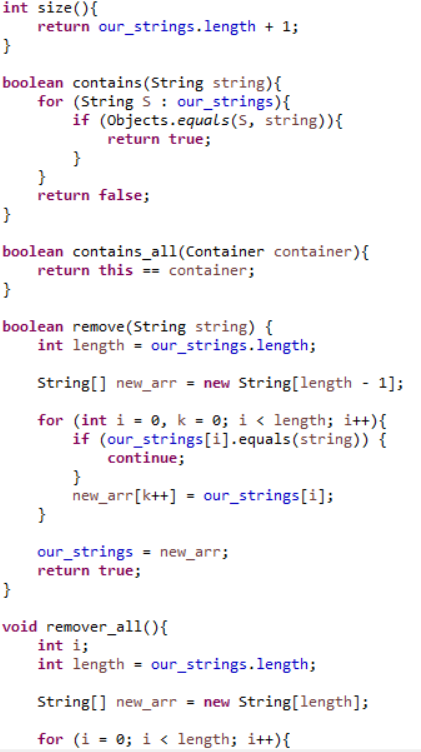


рис.6

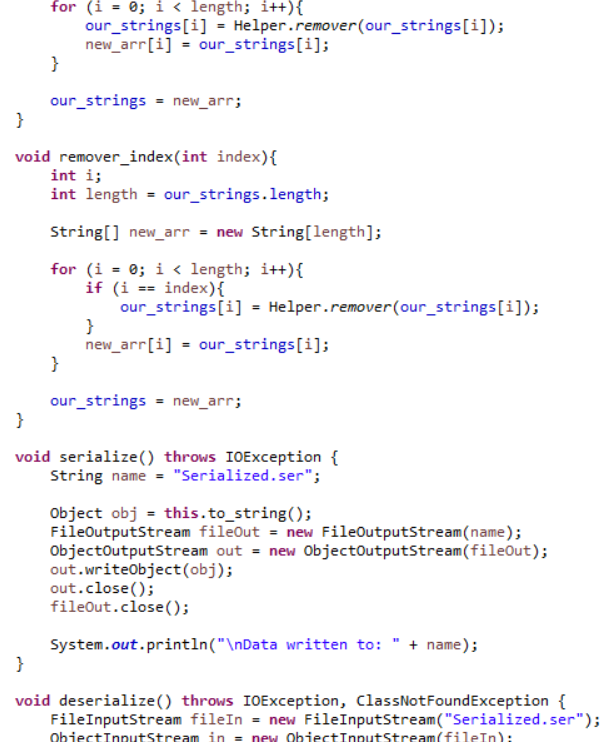


рис.7



рис.8

Висновок: Було набуло навичок тривалого зберігання та відновлення стану об'єктів, ознайомлено з принципами серіалізації/десеріалізації об'єктів, використання бібліотек класів користувача.