**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](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/" \l "_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.

**Розробник**: Білий Вадим Іванович КІТ-119а №3.

**Опис програми**

**Засоби ООП**: клас, метод классу.

**Структура класів:** один публічний клас Main, один утилітарний клас Helper, один клас колекція MyCollection

**Важливі фрагменти програми:**

public static void Menu() {

boolean prz = true;

Scanner scan = new Scanner(System.in);

MyCollection collection = new MyCollection();

MyCollection temp = new MyCollection();

while(prz){

System.out.println("1.Enter word\n2.Show data\n3.Start\n4.Start method other student\n5.Save\n6.Download\n7.Sorting\n8.Find\n9.Compare with saving\n0.End\nInput command id: ");

switch (scan.nextInt()) {

case 1:

System.out.println("Enter word: ");

scan.nextLine();

collection.add(scan.nextLine());

break;

case 2:

System.out.println(collection.toString());

break;

case 3:

Helper.start(collection.toString());

break;

case 4:

VowelsOrConsonants.splitTextInSentenses(new StringBuffer(collection.toString()));

break;

case 5:

System.out.println("Enter file name: ");

scan.nextLine();

try {

collection.Save(scan.nextLine());

} catch (IOException e) {

e.printStackTrace();

}

break;

case 6:

System.out.println("Enter file name: ");

scan.nextLine();

try {

collection.Download(scan.nextLine());

} catch (ClassNotFoundException | IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

break;

case 7:

collection.sort();

break;

case 8:

scan.nextLine();

System.out.println("Enter data:");

int index = collection.find(scan.nextLine());

if(index==-1)

System.out.println("Element don`t found");

else

System.out.println("Element index in mass: "+index);

break;

case 9:

System.out.println("Enter file name: ");

scan.nextLine();

try {

temp.Download(scan.nextLine());

} catch (ClassNotFoundException | IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

if(collection.equals(temp))

System.out.println("Are equal");

else

System.out.println("Not equal");

break;

default:

System.out.println("End program");

prz = false;

break;

}

}

scan.close();

}

public void Save(String fileName) throws IOException {

FileOutputStream outputStream= new FileOutputStream(fileName);

ObjectOutputStream objectOutputStream = new ObjectOutputStream(outputStream);

objectOutputStream.writeObject(this);

objectOutputStream.close();

}

public void Download(String fileName) throws IOException, ClassNotFoundException{

FileInputStream inStream= new FileInputStream(fileName);

ObjectInputStream objectInStream = new ObjectInputStream(inStream);

MyCollection temp = (MyCollection) objectInStream.readObject();

this.setMass(temp.getMass());

objectInStream.close();

}

**Висновки**

Оволодів серіалізацією та десеріалізацією, навчився підключати сторонні бібліотеки та jar файлы.