**Основи введення/виведення Java SE**

**Мета роботи**: Оволодіння навичками управління введенням/виведенням даних з використанням класів платформи Java SE.

**Вимоги**

1. Забезпечити можливість збереження і відновлення масива об'єктів рішення завдання [лабораторної роботи №7](https://oop-khpi.gitlab.io/%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F/task07/).
2. Забороняється використання [стандартного протокола серіалізації](https://docs.oracle.com/javase/8/docs/platform/serialization/spec/serialTOC.html).
3. Продемонструвати використання моделі [Long Term Persistence](https://docs.oracle.com/javase/tutorial/javabeans/advanced/longpersistence.html).
4. Забезпечити діалог з користувачем у вигляді простого текстового меню.
5. При збереженні та відновленні даних забезпечити діалоговий режим вибору директорії з відображенням вмісту і можливістю переміщення по підкаталогах.

**Особисте завдання**

Адресна книга

Запис: П.І.Б .; дата народження; телефони (кількість не обмежена); адреса; дата і час редагування.

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

**Опис програми**

**Засоби ООП**: клас, метод класу.

**Структура класів:** Публічний клас Main, клас даних Person, утилітарний клас AddressBook та Dir.

**Важливі фрагменти програми:**

package ua.khpi.oop.bilyi08;

import java.io.File;

import java.io.FilenameFilter;

import java.util.Scanner;

public class Dir {

public static String directoryForDownload() {

Scanner scan = new Scanner(System.in);

String path = new String("C:\\Users\\Acer\\eclipse-workspace\\bilyi vadim");

File dir = new File(path);

String choose = new String();

do {

String [] list = dir.list(new FilenameFilter() {

@Override

public boolean accept(File f, String name)

{

if(new File(f, name).isDirectory())

return true;

else

return(name.endsWith("xml"));

}

});

System.out.println();

for(String temp : list) {

System.out.println(temp);

}

System.out.println();

System.out.println("Choose: ");

choose = scan.nextLine();

if (choose.equals("quit"))

break;

if (choose.equals("..."))

{

path = path.substring(0,path.lastIndexOf('\\'));

dir = new File(path);

continue;

}

path = path.concat("\\"+choose);

dir = new File(path);

if(!dir.exists())

{

System.out.println("\nNot found: " + choose);

path = path.substring(0,path.length()-(choose.length()+1));

dir = new File(path);

continue;

}

if(dir.isFile())

{

return path;

}

}while(true);

return new String("DONTDOWNLOAD");

}

public Dir() {

super();

// TODO Auto-generated constructor stub

};

}

public void save(String fileName) throws FileNotFoundException {

XMLEncoder encoder = new XMLEncoder(new BufferedOutputStream(new FileOutputStream(fileName)));

encoder.writeObject(mass);

encoder.close();

System.out.println("Serialization successful\n");

}

@SuppressWarnings("unchecked")

public void download(String fileName) throws FileNotFoundException {

XMLDecoder d = new XMLDecoder(new BufferedInputStream(new FileInputStream(fileName)));

mass = (ArrayList<Person>) d.readObject();

d.close();

System.out.println("Serialization successful\n");

}

void delete(int n) {

if(n<mass.size())

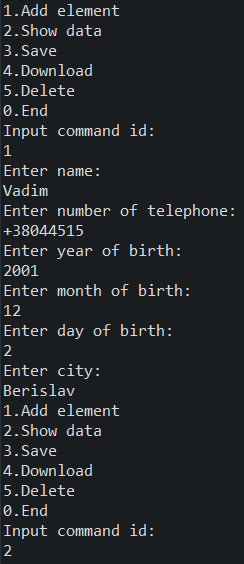
mass.remove(n);

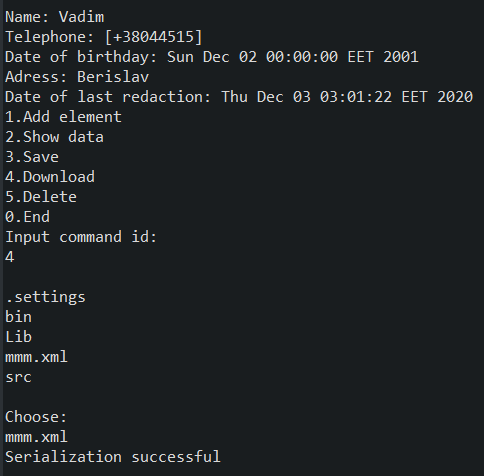
else

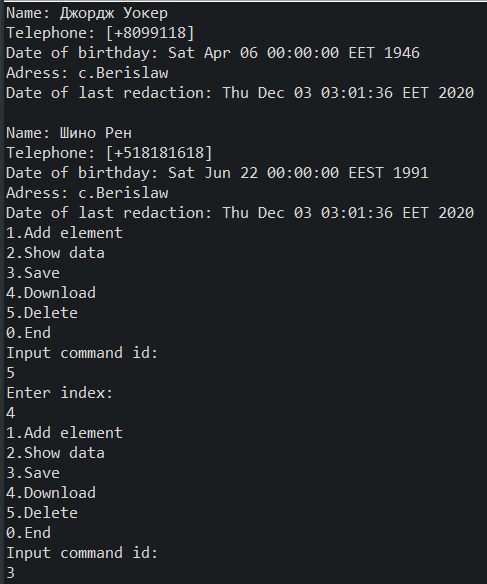
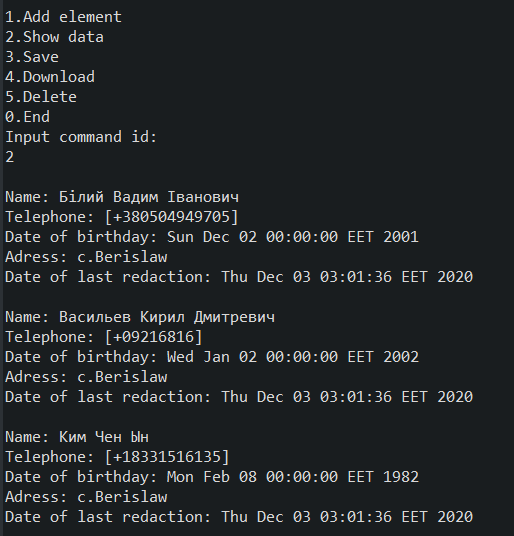
System.out.println("Element don`t exists");

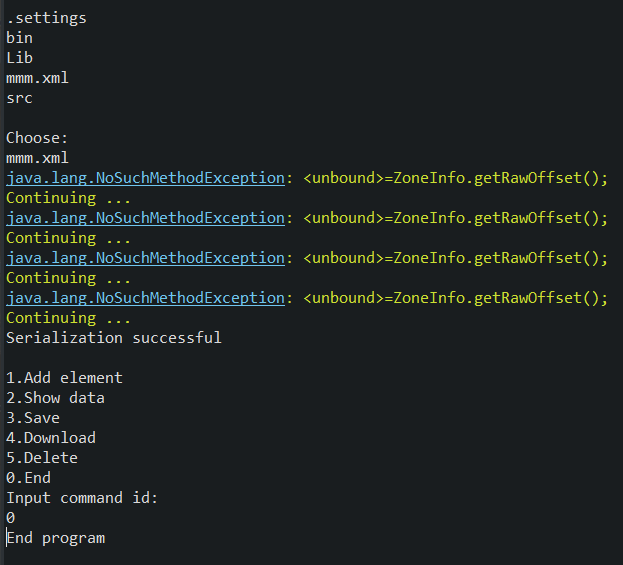
}

**Результати роботи**









**Висновки**

Оволодів навичками серіалізації об’єкту за допомогою використання моделі Long Term Persistence.