Тема: Основи введення/виведення Java SE.

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

**1 ВИМОГИ  
1.1 Розробник**

Інформація про розробника:

- Когутенко Олександр Олексійович;

- КІТ-119Д;

- 11 варіант.

**1.2 Загальне завдання**

* 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/).
* Забороняється використання [стандартного протокола серіалізації](https://docs.oracle.com/javase/8/docs/platform/serialization/spec/serialTOC.html).
* Продемонструвати використання моделі [Long Term Persistence](https://docs.oracle.com/javase/tutorial/javabeans/advanced/longpersistence.html).
* Забезпечити діалог з користувачем у вигляді простого текстового меню.
* При збереженні та відновленні даних забезпечити діалоговий режим вибору директорії з відображенням вмісту і можливістю переміщення по підкаталогах.

2 ОПИС ПРОГРАМИ

2.1 Засоби ООП

Використовуеться наслідування, інтерфейс, поліморфізм.

2.2 Ієрархія та структура класів

Використовую 9 классів: Array, ArrayIterator, HelperClass, InteractiveConsole, Start, SaveArray, Date, Shops, ConsoleFile.

Array використовую як інтерфейс для класу контейнеру.

ArrayIterator використовую як особисту реалізацію ітератора.

HelperClass допоміжний класс для розрахунків.

InteractiveConsole клас для налагодженого спілкування програми з користувачем.

Start клас який має точку входу у програму.

SaveArray клас контейнер який має все необхідні методи маніпулятори.

Date клас використовуется для збереження дати.

Shops клас приклодної галузі.

ConoleFile клас за допомогою якого користувач може наблюдати за відображенням вмісту каталогів.

2.3 Важливі фрагменти програми

package ua.khpi.oop.kogutenko08;  
  
import java.io.Console;  
import java.io.File;  
import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.Scanner;  
  
public class ConsoleFile {  
  
 public static Scanner *in* = new Scanner(System.*in*);  
  
  
 public static int dialogOut() throws IOException, InterruptedException  
 {  
 System.*out*.print("\n\n Оберіть команду:"  
 +"\n 1 - Місце перебування"  
 +"\n 2 - Файли в директорії"  
 +"\n 3 - Вийти із директорії"  
 +"\n 4 - Вибрати файл на перевірку"  
 +"\n 5 - Створити директорію у цій дерикторії"  
 +"\n 6 - Перейти за адресою:"  
 +"\n 7 - Ім'я збереженого файлу у який збережімо колекцію"  
 +"\n\n>>>: ");  
 return *in*.nextInt();  
 }  
  
 public static File MenuFillOut() ///функціє проводить координування по можливостям програм  
 {  
 File file = new File("D:/eclips-workspace/kogutenko-oleksandr/src/ua/khpi/oop/");  
 while(true)///нескінченний цикл який дозволяє працювати програмі  
 {  
 try {  
 int key = *dialogOut*();  
  
 switch(key)///пошук введеної команди  
 {  
 case 1:  
 System.*out*.println("\n Шлях: " + file.getPath()); //getAbsolutePath  
 break;  
 case 2:  
 int a = 0;  
 System.*out*.print("Файли: " + file.getPath() + "\n");  
 for (File file2 : file.listFiles())  
 {  
 if ( a % 5 == 0 )  
 System.*out*.print("\n");  
 a++;  
 System.*out*.printf("%-25s ", file2.getName());  
 } ;///////////  
 break;  
 case 3:  
 file = file.getParentFile();  
 break;  
 case 4:  
 System.*out*.print("Ведіть назву файлу: ");  
 *in*.nextLine();  
 file = new File(file.getAbsolutePath() + "/" + *in*.nextLine());  
 if (file.isFile() == true)  
 return file;  
 if(file.isDirectory() == true);  
 else {  
 System.*out*.print("незнайдено");  
 file = file.getParentFile();  
 }  
 break;  
 case 5:  
 System.*out*.print("Ведіть назву директрії: ");  
 String s = *in*.nextLine();  
 File dir = new File(file.getPath()+"/" + *in*.nextLine());  
 System.*out*.println("Створення директорії: " + dir.mkdirs());  
 break;  
 case 6:  
 System.*out*.print("Ведіть адресу: ");  
 String s3 = *in*.nextLine();  
 s3 = *in*.nextLine();  
 String s2 = new String();  
 for (int i = 0; i < s3.length(); i++)  
 {  
 if (s3.charAt(i)=='\\')  
 s2 += "/";  
 else  
 s2 += s3.charAt(i);  
 }  
 file = new File(s2);  
 break;  
  
 case 7:  
 System.*out*.print("Ведіть назву файлу:");  
 *in*.nextLine();  
 return new File(file.getAbsolutePath() + "/" + *in*.nextLine());  
 }  
 }  
 catch(Exception e)  
 {  
 System.*out*.println(" EROR EROR EROR EROR EROR EROR EROR EROR \n");  
 System.*out*.print("\n\n\nТрапилась помилка. Але тепер все добре!!\n\n");  
 System.*out*.println(e);  
 }  
 }  
  
 }  
  
  
 public static int dialogIn() throws IOException, InterruptedException  
 {  
 System.*out*.print("\n\n Оберіть команду:"  
 +"\n 1 - Місце перебування"  
 +"\n 2 - Файли в директорії"  
 +"\n 3 - Перейти в директорію"  
 +"\n 4 - Вийти із директорії"  
 +"\n 5 - Відкрити файл для зчитування і створення колекції"  
 +"\n 6 - Перейти за адресою:"  
 +"\n\n>>>: ");  
  
 return *in*.nextInt();  
 }  
  
 public static File MenuFillIn() ///функціє проводить координування по можливостям програм  
 {  
 File file = new File("D:/eclips-workspace/kogutenko-oleksandr/src/ua/khpi/oop/");  
 while(true)///нескінченний цикл який дозволяє працювати програмі  
 {  
 try {  
 int key = *dialogIn*();  
  
 switch(key)///пошук введеної команди  
 {  
 case 1:  
 System.*out*.println("\nШлях: " + file.getPath()); //getAbsolutePath  
 break;  
 case 2:  
 int a = 0;  
 System.*out*.print("Файли: \n" + file.getPath() + "\n");  
 for (File file2 : file.listFiles())  
 {  
 if ( a % 4 == 0)  
 System.*out*.print("\n");  
 a++;  
 System.*out*.printf("%-30s ",file2.getName());  
 } ///////////  
 break;  
 case 3:  
 System.*out*.print("Ведіть назву файлу:");  
 file = new File(file.getAbsolutePath() + "/" + *in*.nextLine());  
 if (file.isDirectory() == false)  
 {  
 System.*out*.print("Дирикторія незнайдена");  
 file = file.getParentFile();  
 }  
 break;  
 case 4:  
 file = file.getParentFile();  
 break;  
 case 5:  
 System.*out*.print("Ведіть назву файлу: ");  
 *in*.nextLine();  
 file = new File(file.getAbsolutePath() + "/" + *in*.nextLine());  
 if (file.isFile() == true)  
 return file;  
 break;  
 case 6:  
 {  
 System.*out*.print("Ведіть адресу: ");  
 String s3 = *in*.nextLine();  
 String s2 = new String();  
 for (int i=0; i < s3.length(); i++)  
 {  
 if (s3.charAt(i)=='\\')  
 s2+="/";  
 else  
 s2 += s3.charAt(i);  
 }  
 file = new File(s2);  
 }  
 break;  
 }  
 }  
 catch(Exception e)  
 {  
 System.*out*.println(" EROR EROR EROR EROR EROR EROR EROR EROR EROR \n");  
 System.*out*.print("\n\n\nТрапилась помилка. Але тепер все добре!!\n\n");  
 System.*out*.println(e);}  
 }  
 }  
}

package ua.khpi.oop.kogutenko08;  
  
import java.io.\*;  
import java.util.Scanner;  
import java.beans.XMLDecoder;  
import java.beans.XMLEncoder;  
import java.io.BufferedInputStream;  
import java.io.BufferedOutputStream;  
import java.io.File;  
import java.io.FileInputStream;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.ObjectOutputStream;  
  
*/\*\*  
 \* The type Helper class with console.  
 \*/*public class HelperClass extends Object{  
  
 .......  
 public void serialization()  
 {  
 //File file = ConsoleFile.MenuFillOut();  
 try{  
 XMLEncoder encoder = new XMLEncoder(  
 new BufferedOutputStream(  
 new FileOutputStream( ConsoleFile.*MenuFillOut*()+".xml")));  
  
 encoder.writeObject(save.size());  
  
 for(Shops shop : save)  
 encoder.writeObject(shop);  
  
  
 encoder.close();  
 } catch (FileNotFoundException e) {  
 e.printStackTrace();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void deserializtion() {  
 //Array<Shop> container = new SaveArray<>();  
 File file = ConsoleFile.*MenuFillIn*();///pathname  
 try {  
 FileInputStream fis = new FileInputStream(file);///pathname  
 ObjectInputStream ois = new ObjectInputStream(fis);  
  
 int count = ois.readInt();  
  
 for(int i = 0; i < count; i++)  
 {  
 Shops shops = (Shops)ois.readObject();  
 save.add(shops);  
 }  
 ois.close();  
 }  
 catch(FileNotFoundException e) {e.printStackTrace();}  
 catch (IOException e) {e.printStackTrace(); }  
 catch (ClassNotFoundException e) {e.printStackTrace(); }  
 }  
 public void deserializtionXML()  
 {  
 try{  
 XMLDecoder decoder = new XMLDecoder(  
 new BufferedInputStream(  
 new FileInputStream(ConsoleFile.*MenuFillIn*())));  
  
 int count = (int) decoder.readObject();  
  
 for(int i = 0; i < count; i++)  
 {  
 Shops shops = (Shops)decoder.readObject();  
 save.add(shops);  
 }  
 decoder.close();  
  
 }  
 catch(FileNotFoundException e) {e.printStackTrace();}  
 }  
}

**3 ВАРІАНТИ ВИКОРИСТАННЯ**

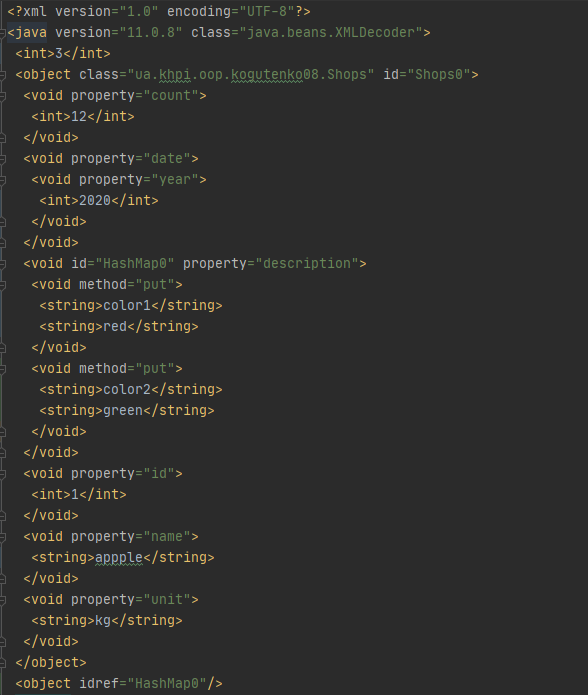


Рисунок 8.1 - інформація початкового файлу

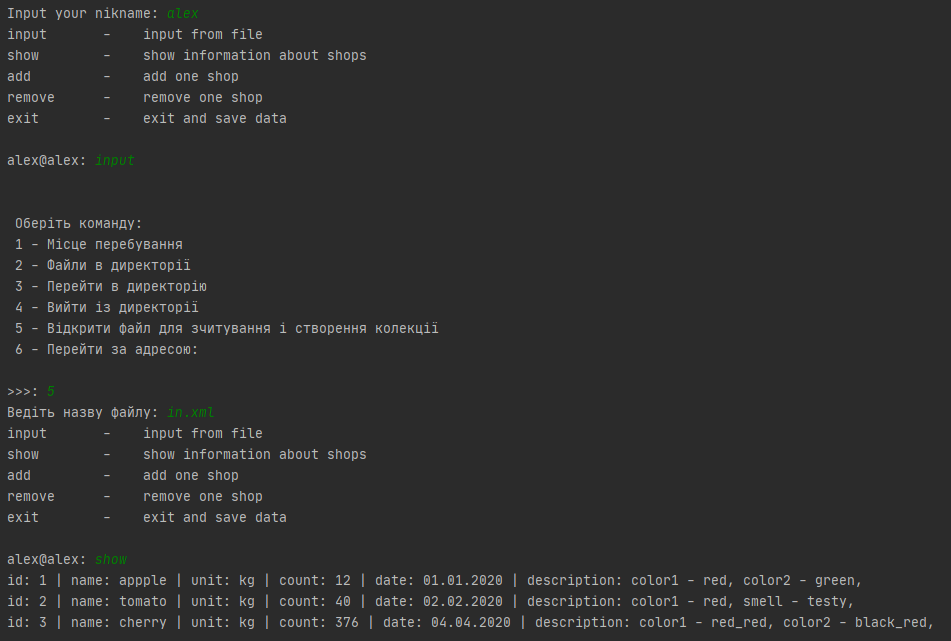


Рисунок 8.2 - інформація з файлу

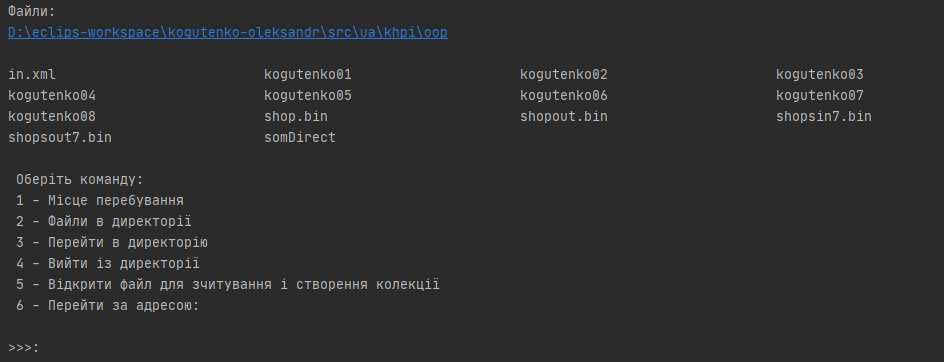


Рисунок 8.3 - файли дерикторії

**ВИСНОВКИ**

Оволодів навичками управління введенням/виведенням данних з використанням класів платформи Java SE.