Лабораторна робота №8

Основи введення/виведення Java SE

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

1 ВИМОГИ

1. Забезпечити можливість збереження і відновлення масива об'єктів рішення завдання лабораторної роботи №7.
2. Забороняється використання стандартного протокола серіалізації.
3. Продемонструвати використання моделі Long Term Persistence.
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.

1.1 Розробник

- П.І.Б: Заночкин. Є. Д.
- Группа: КІТ-119а
- Варіант: 7

2 ОПИС ПРОГРАМИ

2.1 Було використано наступні засоби:

File folder = new File (absolutePath) – отримання адреси каталогу;

listFiles.length() – визначення довжини масиву назв каталогів та файлів,
XMLEncoder encoder = new XMLEncoder(new BufferedOutputStream(new FileOutputStream(file))),
encoder.writeObject(list.array), encoder.close() – серіалізація;

XMLDecoder decoder = new XMLDecoder(new BufferedInputStream(new FileInputStream(file))), list.array = (Client[]) decoder.readObject(),
decoder.close() – десеріалізація.

2.2 Ієрархія та структура класів

Було створено 2 класи:

- public class Main – містить метод main;
- public class ClientList – містить масив типу Client та методи його обробки.

Також було підключено класи Client, InfoAboutYourself та PartnerRequirements з попередньої лабораторної роботи.

2.3 Важливі фрагменти програми

Клас ClientList

```
public class ClientList
{
	private int size = 2;
	/*
	 * Масив типу Client
	 */
	Client array[] = new Client[size];
	/*
	 * Геттер та сеттер
	 */
	public int getSize()
	{
		return size;
	}
	public void setSize(int size)
	{
		this.size = size;
	}
	/*
	 * Метод printAll, що виводить інформацію про всіх клієнтів на екран
	 */
	public void printAll()
	{
		if(size > 0)
			for(int i = 0; i < array.length; i++)
			{
				System.out.println("ID - " + array[i].getId() + "\nRegistration date - " + array[i].getDate() + "\nGender - " + array[i].getClientGender() + "\n");
				System.out.println("Information about yourself:\nName - " + array[i].getInformation().getName() + "\nAge - " + array[i].getInformation().getAge() +
					"\nHeight - " + array[i].getInformation().getHeight() + "\nEye colour - " + array[i].getInformation().getEyeColour() +
					"\nHobby - " + array[i].getInformation().getClientHobby() + "\n");
				System.out.println("Partner requirements:\nGender - " + array[i].getRequirements().getPartnerGender() +
					"\nMin age - " + array[i].getRequirements().getMinAge() + "\nMax age - " + array[i].getRequirements().getMaxAge());
				System.out.println("----------------------------------------");
			}
		else
		{
			System.out.println("Empty list");
			System.out.println("----------------------------------------");
		}
	}
	/*
	 * Метод print, що виводить інформацію про одного клієнта на екран
	 */
	public void print(int num)
	{
		System.out.println("ID - " + array[num].getId() + "\nRegistration date - " + array[num].getDate() + "\nGender - " + array[num].getClientGender() + "\n");
		System.out.println("Information about yourself:\nName - " + array[num].getInformation().getName() + "\nAge - " + array[num].getInformation().getAge() +
			"\nHeight - " + array[num].getInformation().getHeight() + "\nEye colour - " + array[num].getInformation().getEyeColour() +
			"\nHobby - " + array[num].getInformation().getClientHobby() + "\n");
		System.out.println("Partner requirements:\nGender - " + array[num].getRequirements().getPartnerGender() +
```

```
"\nMin age - " + array[num].getRequirements().getMinAge() + "\nMax age - " + array[num].getRequirements().getMaxAge());
                System.out.println("----------------------------------------");
            }
            /*
             * Метод add, який додає нового клієнта
             */
            public void add(Client string)
            {
                Client newArr[] = new Client[size + 1];
                for (int i = 0; i < size; i++)
                    newArr[i] = array[i];
                size++;
                newArr[size - 1] = string;
                array = newArr;
            }
            /*
             * Метод remove, який видаляє одного з клієнтів
             */
            void remove(int num)
            {
                Client newArr[] = new Client[size - 1];
                for (int i = 0; i < num; i++)
                    newArr[i] = array[i];
                for (int i = num, j = num + 1; j < size; i++, j++)
                    newArr[i] = array[j];
                size--;
                array = newArr;
            }
            /*
             * Метод clear, який очищує список клієнтів
             */
            void clear()
            {
                size = 0;
                Client newArr[] = new Client[size];
                array = newArr;
            }
        }
```

Клас Main

```
        public class Main
        {
            public static void main(String[] args)
            {
                ClientList list = new ClientList();
                InfoAboutYourself info = new InfoAboutYourself("Yehor", 18, 185, "Blue", "Video games");
                PartnerRequirements requirements = new PartnerRequirements("Female", 18, 25);
                int ID = 1;
                list.array[0] = new Client("Male", ID++, "01.05.2018", info, requirements);
                info = new InfoAboutYourself("Kate", 17, 170, "Green", "Art, singing");
                requirements = new PartnerRequirements("Male", 18, 25);
                list.array[1] = new Client("Female", ID++, "12.12.2020", info, requirements);
                boolean endCheck = true;
                Scanner inInt = new Scanner(System.in);
                Scanner inStr = new Scanner(System.in);
                while (endCheck)
                {
                    System.out.println("\nMenu:");
                    System.out.println("1. Show clients");
                    System.out.println("2. Add client");
                    System.out.println("3. Delete client");
```

```
System.out.println("4. Change information");
System.out.println("5. Clear list");
System.out.println("6. Serialize data");
System.out.println("7. Deserialize data");
System.out.println("8. Exit");
System.out.println("Enter your option:");
int option = inInt.nextInt();
System.out.println();
switch (option)
{
case 1:
	list.printAll();
	break;
case 2:
	System.out.println("Enter gender:");
	String gender = inStr.nextLine();
	System.out.println("Enter registration date:");
	String date = inStr.nextLine();
	System.out.println("Enter information about yourself: Name, age, height, eye colour, hobby.");
	info = new InfoAboutYourself(inStr.nextLine(), inInt.nextInt(), inInt.nextInt(), inStr.nextLine(), inStr.nextLine());
	System.out.println("Enter partner requirements: Gender, min age, max age.");
	requirements = new PartnerRequirements(inStr.nextLine(), inInt.nextInt(), inInt.nextInt());
	list.add(new Client(gender, ID++, date, info, requirements));
	System.out.println();
	list.printAll();
	break;
case 3:
	System.out.println("Enter client's ID to delete him:");
	int delete = inInt.nextInt();
	int sizeBeforeDeleting = list.getSize();
	for(int i = 0; i < list.getSize(); i++)
		if(list.array[i].getId() == delete)
		{
			list.remove(i);
			System.out.println();
			list.printAll();
			break;
		}
	if(sizeBeforeDeleting == list.getSize())
		System.out.println("There is no client with that ID.");
	break;
case 4:
	System.out.println("Enter client's ID to change his information:");
	int id = inInt.nextInt();
	int index = 0;
	for(index = 0; index < list.getSize(); index++)
		if(list.array[index].getId() == id)
			break;
	if(index == list.getSize())
	{
		System.out.println("There is no client with that ID.");
		break;
	}
	boolean endCheck2 = true;
	int option2 = 0;
	while(endCheck2)
	{
		System.out.println();
		list.print(index);
```

```
                                        System.out.println("\nWhich information you want to
change?");
                                        System.out.println("1. Gender");
                                        System.out.println("2. ID");
                                        System.out.println("3. Registration date");
                                        System.out.println("4. Information about yourself");
                                        System.out.println("5. Partner requirements");
                                        System.out.println("6. End of change");
                                        System.out.println("Enter option:");
                                        option2 = inInt.nextInt();
                                        System.out.println();
                                        switch(option2)
                                        {
                                        case 1:
                                                System.out.println("Enter new gender:");
                                                list.array[index].setClientGender(inStr.nextLine());
                                                break;
                                        case 2:
                                                System.out.println("Enter new ID:");
                                                list.array[index].setId(inInt.nextInt());
                                                break;
                                        case 3:
                                                System.out.println("Enter new registration date:");
                                                list.array[index].setDate(inStr.nextLine());
                                                break;
                                        case 4:
                                                System.out.println("Information about yourself:");
                                                System.out.println("1. Name");
                                                System.out.println("2. Age");
                                                System.out.println("3. Height");
                                                System.out.println("4. Eye colour");
                                                System.out.println("5. Hobby");
                                                System.out.println("Enter option:");
                                                int option3 = inInt.nextInt();
                                                System.out.println();
                                                switch(option3)
                                                {
                                                case 1:
                                                        System.out.println("Enter new name:");

        list.array[index].getInformation().setName(inStr.nextLine());
                                                        break;
                                                case 2:
                                                        System.out.println("Enter new age:");

        list.array[index].getInformation().setAge(inInt.nextInt());
                                                        break;
                                                case 3:
                                                        System.out.println("Enter new height:");

        list.array[index].getInformation().setHeight(inInt.nextInt());
                                                        break;
                                                case 4:
                                                        System.out.println("Enter new eye colour:");

        list.array[index].getInformation().setEyeColour(inStr.nextLine());
                                                        break;
                                                case 5:
                                                        System.out.println("Enter new hobby:");

        list.array[index].getInformation().setClientHobby(inStr.nextLine());
                                                        break;
                                                }
```

```
                                        break;
                                case 5:
                                        System.out.println("Partner requirements:");
                                        System.out.println("1. Gender");
                                        System.out.println("2. Min age");
                                        System.out.println("3. Max age");
                                        System.out.println("Enter option:");
                                        option3 = inInt.nextInt();
                                        System.out.println();
                                        switch(option3)
                                        {
                                        case 1:
                                                System.out.println("Enter new gender:");

list.array[index].getRequirements().setPartnerGender(inStr.nextLine());
                                                break;
                                        case 2:
                                                System.out.println("Enter new min age:");

list.array[index].getRequirements().setMinAge(inInt.nextInt());
                                                break;
                                        case 3:
                                                System.out.println("Enter new max age:");

list.array[index].getRequirements().setMaxAge(inInt.nextInt());
                                                break;
                                        }
                                        break;
                                case 6:
                                        endCheck2 = false;
                                        break;
                                default:
                                        System.out.println("Wrong command.");
                                        break;
                                }
                        }
                        break;
                case 5:
                        list.clear();
                        System.out.println("List cleared.");
                        break;
                case 6:
                        String absolutePath = new File("").getAbsolutePath();
                        File folder = new File(absolutePath);
                        File[] listFiles = folder.listFiles();
                        String filename;
                        String currentDir = absolutePath;
                        String highestDir = folder.getName();
                        endCheck2 = true;
                        boolean leave = false;
                        index = 0;
                        option2 = 0;
                        System.out.print("Enter XML filename:");
                        filename = inStr.nextLine();
                        if (filename.indexOf(".xml") == -1)
                                filename += ".xml";
                        while(endCheck2)
                        {
                                index = 0;
                                System.out.println("\nCurrent path: " + currentDir);
                                System.out.println("XML file name: " + filename);
                                System.out.println("\nFiles and directories in this path:");
                                for (index = 0; index < listFiles.length; index++)
```

```
                                        System.out.println(index + 1 + ". " +
listFiles[index].toString().substring(currentDir.length()+1));
                                System.out.println();
                                System.out.println("Serialization menu:");
                                System.out.println("1. Write XML file in current directory");
                                System.out.println("2. Move up one level");
                                System.out.println("3. Enter the folder");
                                System.out.println("4. End of serialization");
                                System.out.print("Enter option:");
                                option2 = inInt.nextInt();
                                System.out.println();
                                switch(option2)
                                {
                                case 1:
                                        endCheck2 = false;
                                        break;
                                case 2:
                                        if(folder.getName().equals(highestDir))
                                        {
                                                System.out.print("This is the highest
directory.");
                                                break;
                                        }
                                        currentDir = currentDir.substring(0,
currentDir.indexOf(folder.getName())-1);
                                        folder = new File(currentDir);
                                        listFiles = folder.listFiles();
                                        break;
                                case 3:
                                        boolean option3 = true;
                                        while(option3)
                                        {
                                                System.out.print("Choose the number of
folder:");
                                                index = inInt.nextInt();
                                                if(!listFiles[index-1].isDirectory() || index < 1
|| index > listFiles.length)
                                                        System.out.println("Error, that's not
a folder.");
                                                else
                                                {
                                                        currentDir = listFiles[index-
1].toString();
                                                        System.out.println("New current
directory:" + currentDir);
                                                        folder = new File(currentDir);
                                                        listFiles = folder.listFiles();
                                                        option3 = false;
                                                }
                                        }
                                        break;
                                case 4:
                                        System.out.println("End of serialization");
                                        leave = true;
                                        endCheck2 = false;
                                        break;
                                default:
                                        System.out.println("Wrong command.");
                                        break;
                                }
                        }
                        if(leave == true)
                                break;
```

```
                                absolutePath = currentDir;
                                folder = new File(absolutePath);
                                File file = new File(folder,filename);
                                try
                                {
                                        XMLEncoder encoder = new XMLEncoder(new
BufferedOutputStream(new FileOutputStream(file)));
                                        encoder.writeObject(list.array);
                                        encoder.close();
                                }
                                catch (Exception e)
                                {
                                        System.out.println(e);
                                        break;
                                }
                                System.out.println("File was written in this directory: " + absolutePath);
                                System.out.println("Serialization complete.");
                                break;
                        case 7:
                                absolutePath = new File("").getAbsolutePath();
                                folder = new File(absolutePath);
                                listFiles = folder.listFiles();
                                currentDir = absolutePath;
                                highestDir = folder.getName();
                                leave = false;
                                endCheck2 = true;
                                index = 0;
                                option2 = 0;
                                while(endCheck2)
                                {
                                        index = 0;
                                        System.out.println("Current path: " + currentDir);
                                        System.out.println("Files and directories in this path:");
                                        for (index = 0; index < listFiles.length; index++) {
                                                System.out.println(index + 1 + ". " +
listFiles[index].toString().substring(currentDir.length()+1));
                                        }
                                        System.out.println();
                                        System.out.println("Deserialization menu:");
                                        System.out.println("1. Read XML file in current directory");
                                        System.out.println("2. Move up one level");
                                        System.out.println("3. Enter the folder");
                                        System.out.println("4. End of deserialization");
                                        System.out.print("Enter option:");
                                        option2 = inInt.nextInt();
                                        System.out.println();
                                        switch(option2)
                                        {
                                        case 1:
                                                System.out.print("Enter ID of the file:");
                                                index = inInt.nextInt();
                                                if(listFiles[index-1].getName().indexOf(".xml") == -1
|| listFiles[index-1].isDirectory())
                                                {
                                                        System.out.println("Error, that's not a .XML
file.");
                                                        break;
                                                }
                                                endCheck2 = false;
                                                break;
                                        case 2:
                                                if(folder.getName().equals(highestDir))
                                                {
```

```
                                        System.out.println("This is the highest directory.");
                                        break;
                                }
                                currentDir = currentDir.substring(0, currentDir.indexOf(folder.getName())-1);
                                folder = new File(currentDir);
                                listFiles = folder.listFiles();
                                break;
                        case 3:
                                boolean option3 = true;
                                while(option3)
                                {
                                        System.out.print("Choose the number of folder:");
                                        index = inInt.nextInt();
                                        if(!listFiles[index-1].isDirectory() || index < 1 || index > listFiles.length)
                                                System.out.println("Error, that's not a folder.");
                                        else
                                        {
                                                currentDir = listFiles[index-1].toString();
                                                System.out.println("New current directory: " + currentDir);
                                                folder = new File(currentDir);
                                                listFiles = folder.listFiles();
                                                option3 = false;
                                        }
                                }
                                break;
                        case 4:
                                System.out.println("End of deserialization");
                                leave = true;
                                endCheck2 = false;
                                break;
                        default:
                                System.out.println("Wrong command.");
                                break;
                        }
                }
                if(leave == true)
                        break;
                absolutePath = currentDir + "\\" + listFiles[index-1].getName();
                file = new File(absolutePath);
                try
                {
                        XMLDecoder decoder = new XMLDecoder(new BufferedInputStream(new FileInputStream(file)));
                        list.array = (Client[])decoder.readObject();
                        decoder.close();
                        list.setSize(list.array.length);
                }
                catch (Exception e)
                {
                        System.out.println(e);
                        break;
                }
                System.out.println("File was read from this directory: " + listFiles[index-1]);
                System.out.println("Deserialization complete.");
                break;
```

```
                    case 8:
                        endCheck = false;
                        inInt.close();
                        inStr.close();
                        break;
                    default:
                        System.out.println("Wrong command\n");
                        break;
                    }
                }
                System.out.println("End");
            }
}
```

3 Варіанти використання

У результаті виконання лабораторної роботи було розроблено меню, яке дозволяє користувачу:

1. Вивести усі елементи у консоль (1 команда);
2. Додати елемент у список (2 команда);
3. Видалити елемент зі списку (3 команда);
4. Змінити інформацію в елементі (4 команда);
5. Очистити список (5 команда);
6. Серіалізувати поточний список у файл (6 команда);
7. Десеріалізувати дані з файлу у список (7 команда);

4 Результати роботи програми

```
Menu:
1. Show clients
2. Add client
3. Delete client
4. Change information
5. Clear list
6. Serialize data
7. Deserialize data
8. Exit
Enter your option:
1

ID - 1
Registration date - 01.05.2018
Gender - Male

Information about yourself:
Name - Yehor
Age - 18
Height - 185
Eye colour - Blue
Hobby - Video games

Partner requirements:
Gender - Female
Min age - 18
Max age - 25
--------------------------------
ID - 2
Registration date - 12.12.2020
Gender - Female

Information about yourself:
Name - Kate
Age - 17
Height - 170
Eye colour - Green
Hobby - Art, singing

Partner requirements:
Gender - Male
Min age - 18
Max age - 25
--------------------------------
```

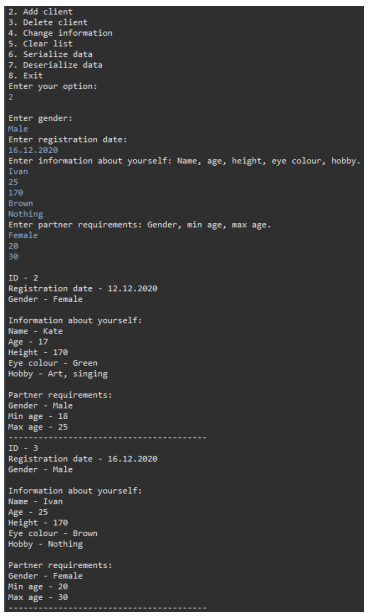
а)

```
Menu:
1. Show clients
2. Add client
3. Delete client
4. Change information
5. Clear list
6. Serialize data
7. Deserialize data
8. Exit
Enter your option:
6

Enter XML filename:123

Current path: D:\Eclipse_Workspace\zanochkyn-yehor
XML file name: 123.xml

Files and directories in this path:
1. .classpath
2. .project
3. 123.xml
4. bin
5. doc
6. Serial.ser
7. src
8. ua.khpi.oop.adbullin03.jar
9. ua.khpi.oop.zanochkyn03.jar

Serialization menu:
1. Write XML file in current directory
2. Move up one level
3. Enter the folder
4. End of serialization
Enter option:1

File was written in this directory: D:\Eclipse_Workspace\zanochkyn-yehor
Serialization complete.
```

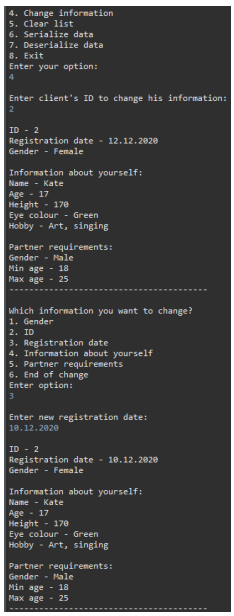
б)

```
Menu:
1. Show clients
2. Add client
3. Delete client
4. Change information
5. Clear list
6. Serialize data
7. Deserialize data
8. Exit
Enter your option:
3

Enter client's ID to delete him:
1

ID - 2
Registration date - 12.12.2020
Gender - Female

Information about yourself:
Name - Kate
Age - 17
Height - 170
Eye colour - Green
Hobby - Art, singing

Partner requirements:
Gender - Male
Min age - 18
Max age - 25
--------------------------------
```

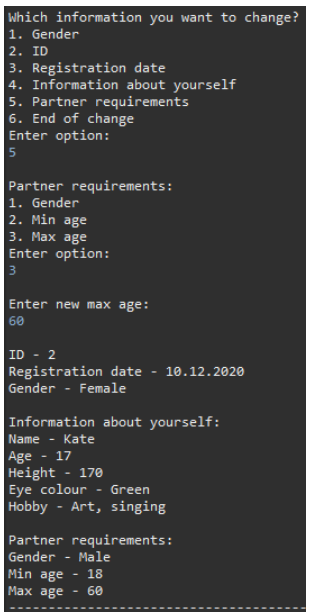
в)



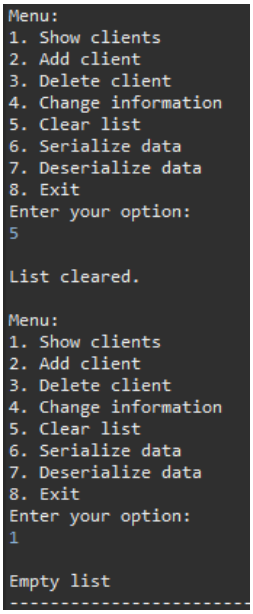
г)



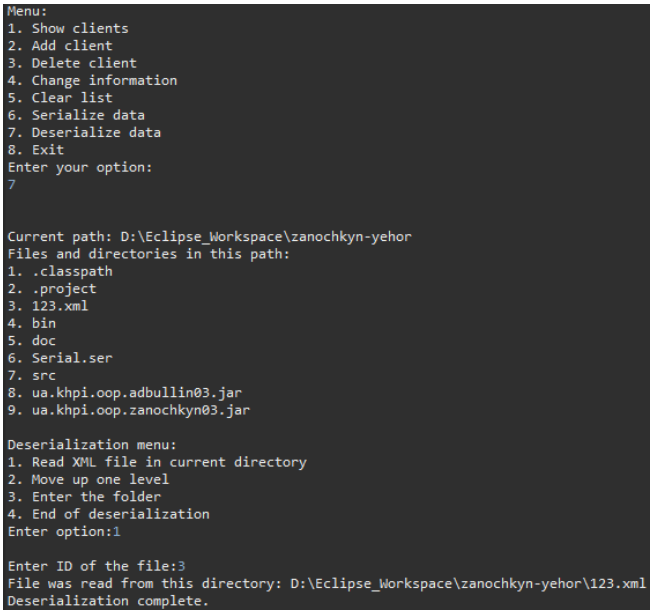
г)



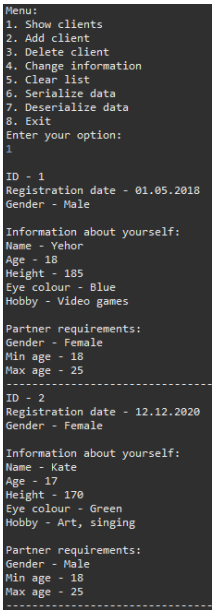
д)



е)



є)



ж)

Рисунок 8.1 – Результат роботи програми у середовищі Eclipse

Висновок

Під час виконання лабораторної роботи було набуто навичок роботи з основами введення/виведення у середовищі Eclipse IDE.