1. Тема роботи

Об’єктно-орієнтована декомпозиція. Основи введення/виведення Java SE.

1. ВИМОГИ

1.1 Розробник

Подоба Арсен Мирославович
КН-108
Варіант 1

1.2 Загальне завдання

Вимоги:
1. Використовуючи об’єктно-орієнтований аналіз, реалізувати класи для представлення сутностей відповідно списку прикладних задач - domain-об’єктів (Прикладні задачі. Список №2. 20 варіантів)
2. Забезпечити та продемонструвати коректне введення та відображення кирилиці.
3. Продемонструвати можливість управління масивом domain-об’єктів.
4. Забезпечити можливість збереження і відновлення масива об’єктів рішення завдання з Прикладні задачі. Список №2. 20 варіантів.
5. Забороняється використання стандартного протокола серіалізації .
6. Продемонструвати використання моделі Long Term Persistence .
7. Забезпечити діалог з користувачем у вигляді текстового меню.
8. 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

Декомпозиція для розділення завдання між класами, композиція, інкапсуляція, серіалізація, десеріалізація

2.2 Важливі фрагменти програми

Обробка даних користувача

```
import java.io.Serializable;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.Scanner;

public class CadreAgency implements Serializable
{
    private ArrayList<Person> people;
    private Scanner scanner = new Scanner(System.in);

    CadreAgency()
    {
        people = new ArrayList<>();
        people.add(new Person(
                new Work(15,"Programmer"),
                "Computer science",
                LocalDate.of(2015, 12, 11)));

        people.add(new Person(
                        new Work(4,"Office manager"),
                        "Economical science",
                new RequireToFutureWork("Director of
financial part", null, 0),
                LocalDate.of(2018, 4, 8)));
    }

    public void showPerson()
    {
        if(!people.isEmpty())
        {
            for (Person x : people)
                System.out.println(x+"\n");
        }
        else
            System.out.println("Cadre Agency hasn't
employee");
    }

    public boolean removePerson(int registrationNumber)
    {
        for(int i = 0; i < people.size(); i++)
        {
            if (people.get(i).getRegistrotionNumber()
== registrationNumber)
```

```
                {
                people.remove(i);
                return true;
            }
        }
        return false;
    }

    public void addPerson()
    {
        System.out.println("Enter person's specialty:
");
        String specialty = scanner.nextLine();
        System.out.println("Enter amount of years while
person was working: ");
        int years = scanner.nextInt();
        Work job = new Work(years, specialty);

        System.out.println("Enter person's Education");
        scanner.nextLine();
        String education = scanner.nextLine();

        System.out.println("Enter data of person's
release: Year Month Day");
        int year = scanner.nextInt();
        int month = scanner.nextInt();
        int day = scanner.nextInt();

        System.out.println("Does person has any
require? Press y - yes, n - no.");
        scanner.nextLine();
        String choice = scanner.nextLine();

        if (choice.equals("n"))
        {
            people.add(new Person(job, education,
LocalDate.of(year, month, day)));
            System.out.println("Person has been
added");
        }
        else if (choice.equals("y"))
        {
            String specialtyWanted;
            String conditionOfwork;
```

```
            int minSalary;

            System.out.println("Enter specialty, on
which person wants. You can skip this point");
            specialtyWanted = scanner.nextLine();

            System.out.println("Enter special
condition. You can skip this point");
            conditionOfwork = scanner.nextLine();

            System.out.println("Enter minimal salary.
You can skip this point");
            minSalary = scanner.nextInt();

            people.add(new Person(job, education, new
RequireToFutureWork(specialtyWanted, conditionOfwork,
minSalary), LocalDate.of(year, month, day)));

            System.out.println("Person has been
added");
        }
        else
            System.out.println("Enter your
choice....");
    }

    public ArrayList<Person> getPeople() { return
people; }

}
```

ВИСНОВКИ

В ході лабораторної роботи, я навчився використовувати об’єктно-орієнтованого підхід для розробки об’єкта предметної (прикладної) галузі. Оволодів навичками управління введенням/виведенням даних з використанням класів Java SE.