**Лабораторна робота №7**

Об’єктно-орієнтована декомпозиція

**Мета:** Використання об'єктно-орієнтованого підходу для розробки об'єкта предметної (прикладної) галузі.

1 ВИМОГИ

* Використовуючи об'єктно-орієнтований аналіз, реалізувати класи для представлення сутностей відповідно прикладної задачі - domain-об'єктів.
* Забезпечити та продемонструвати коректне введення та відображення кирилиці.
* Продемонструвати можливість управління масивом domain-об'єктів.

1.1 Розробник

* Абдулаєв Ібрагім Заурбекович
* Група КІТ-119В
* Варіант 1

2 ОПИС ПРОГРАМИ

2.1 Опис класів

* **ua.khpi.oop.abdulaev07.Challenger**



Рис. 2.1 – Опис класу **Challenger**

* **ua.khpi.oop.zhukov08.RecruitingAgency**

****

Рис. 2.2 – Опис класу **RecruitingAgency**

2.2 Тест програми

**Challenger.java**

package ua.khpi.oop.abdulaev07;  
  
import java.util.Arrays;  
import java.util.HashMap;  
import java.util.Map;  
  
public class Challenger {  
 Map<String, Integer> experience;  
 String education;  
 String dateOfDismissal;  
 String[][] requirementsForWork;  
  
 public Challenger(Map<String, Integer> experience, String education, String dateOfDismissal, String[][] requirementsForWork) {  
 this.experience = experience;  
 this.education = education;  
 this.dateOfDismissal = dateOfDismissal;  
 this.requirementsForWork = requirementsForWork;  
 }  
  
 public Challenger() {  
 this.experience = new HashMap<>();  
 this.education = "None";  
 this.dateOfDismissal = "None";  
 this.requirementsForWork = new String[][] {};  
 }  
  
 public Map<String, Integer> getExperience() {  
 return experience;  
 }  
  
 public String getEducation() {  
 return education;  
 }  
  
 public String getDateOfDismissal() {  
 return dateOfDismissal;  
 }  
  
 public String[][] getRequirementsForWork() {  
 return requirementsForWork;  
 }  
  
 public void setExperience(Map<String, Integer> experience) {  
 this.experience = experience;  
 }  
  
 public void setEducation(String education) {  
 this.education = education;  
 }  
  
 public void setDateOfDismissal(String dateOfDismissal) {  
 this.dateOfDismissal = dateOfDismissal;  
 }  
  
 public void setRequirementsForWork(String[][] requirementsForWork) {  
 this.requirementsForWork = requirementsForWork;  
 }  
  
 @Override  
 public String toString() {  
 return "Challenger[" +  
 "experience=" + experience +  
 ", education='" + education + '\'' +  
 ", dateOfDismissal='" + dateOfDismissal + '\'' +  
 ", requirementsForWork=" + Arrays.*toString*(requirementsForWork) +  
 ']';  
 }  
}

**RecruitingAgency.java**

package ua.khpi.oop.abdulaev07;  
  
import java.beans.XMLDecoder;  
import java.beans.XMLEncoder;  
import java.io.FileInputStream;  
import java.io.FileOutputStream;  
import java.util.ArrayList;  
import java.util.Iterator;  
  
public class RecruitingAgency {  
 ArrayList<Challenger> list;  
  
 public RecruitingAgency() {  
 list = new ArrayList<>();  
 }  
  
 public void add(Challenger challenger) {  
 list.add(challenger);  
 }  
  
 public Challenger remove(int index) {  
 return list.remove(index);  
 }  
  
 public Challenger get(int index) {  
 return list.get(index);  
 }  
  
 public Iterator<Challenger> iterator() {  
 return list.iterator();  
 }  
  
 public void print() {  
 for (int i = 0; i < size(); i++) {  
 System.*out*.printf("%d. %s", i + 1, get(i).toString());  
 }  
 }  
  
 public int size() {  
 return list.size();  
 }  
  
 public void read(String filePath) {  
 try {  
 XMLDecoder decoder = new XMLDecoder(new FileInputStream(filePath));  
 list = new ArrayList<>((ArrayList<Challenger>) decoder.readObject());  
  
 decoder.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 public void write(String filePath) {  
 try {  
 XMLEncoder encoder = new XMLEncoder(new FileOutputStream(filePath));  
 encoder.writeObject(filePath);  
  
 encoder.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

**Main.java**

package ua.khpi.oop.abdulaev07;  
  
import java.util.HashMap;  
  
public class Main {  
 public Main() {  
 }  
  
 public static void main(String[] args) {  
 RecruitingAgency recruitingAgency = new RecruitingAgency();  
 System.*out*.println("1) Додати новий елемент до масиву ->");  
 recruitingAgency.add(  
 new Challenger(  
 new HashMap<>() {{this.put("job1", 1);}},  
 "Вища",  
 "None",  
 new String[][] {  
 {"req1", "req2", "req3"}  
 }  
 )  
 );  
 recruitingAgency.print();  
 int index = 10;  
 System.*out*.println("\n2) Отримати неіснуючий елемент масиву ->");  
 if (index < 0 || index > recruitingAgency.size()) {  
 System.*out*.printf("Елемент за індексом %d не знайдено\n", index);  
 }  
  
 System.*out*.println("3) Видалити елемент з масиву ->");  
 recruitingAgency.remove(0);  
 recruitingAgency.print();  
 }  
}

3 ВИСНОВКИ

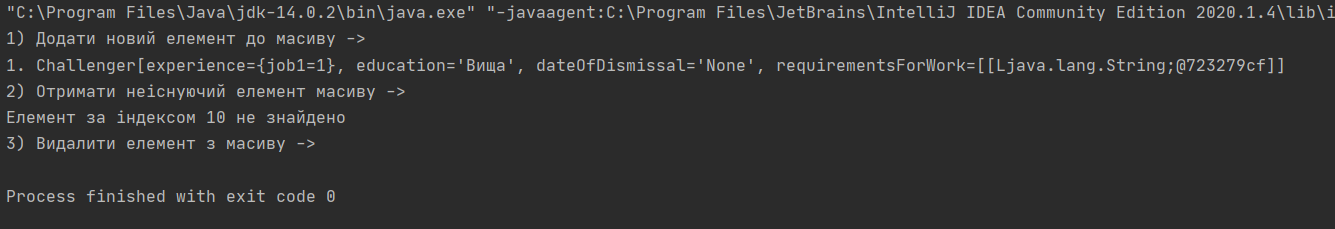


Рис. 3.1 – Результат роботи програми

На лабораторній роботі навчилися використовувати об'єктно-орієнтований підхід для розробки об'єкта предметної (прикладної) галузі.