**Звіт**

**Лабораторна работа 7. Об'єктно-орієнтована декомпозиція**

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

**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.

14. Планувальник

Захід: дата, час початку і тривалість; місце проведення; опис; учасники (кількість не обмежена).

* 1. **Розробник**: Момот Роман Євгенійович, КІТ119-а, варіант №14.

1. **ОПИС ПРОГРАМИ**
   1. **Засоби ООП**: клас, метод класу, поле класу.
   2. **Ієрархія та структура класів:** один публічний клас Main та публічний клас Event, у полях якого є час початку події, тривалість, адреса події, кількість людей, гетери, сетери, конструктор класу та метод виведення даних класу.
   3. **Важливі фрагменти програми:**

public class Event {

private GregorianCalendar startTime; //время начала

private int duration; //длительность

private String address; //место проведения

private String[] people; //имена участников

public GregorianCalendar getStartTime() {

return startTime;

}

public void setStartTime(GregorianCalendar startTime) {

this.startTime = startTime;

}

public int getDuration() {

return duration;

}

public void setDuration(int duration) {

this.duration = duration;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String[] getPeople() {

return people;

}

public void setPeople(String[] people) {

this.people = people;

}

public Event(GregorianCalendar date, int length, String address, String[] people)

{

startTime = date;

duration = length;

this.address = address;

this.people = people;

}

public void outputData()

{

System.out.println("Event start time: " + startTime.getTime());

System.out.println("Duration of the event (in minutes): " + duration);

System.out.println("Event address: " + address);

System.out.println("List of participants: ");

for (int i = 0; i < people.length; i++) {

System.out.println(i+1 + ". " + people[i]);

}

}

}

public class Main {

public static void main(String[] args) {

String[] listOfPeople1 = {"Дмитрий Иванов", "Александр Гекторов", "Иван Романов"};

String[] listOfPeople2 = {"Махатма Ганди", "Иисак Ньютон", "Джордж Буш Младший"};

GregorianCalendar date1 = new GregorianCalendar(2017, 5, 28);

date1.set(Calendar.HOUR\_OF\_DAY, 18);

date1.set(Calendar.MINUTE, 0);

date1.set(Calendar.SECOND, 10);

Event event1 = new Event(date1,180,"Проспект Льва Ландау 87",listOfPeople1);

date1 = new GregorianCalendar(2002, 1, 1);

date1.set(Calendar.HOUR\_OF\_DAY, 9);

date1.set(Calendar.MINUTE, 30);

date1.set(Calendar.SECOND, 00);

Event event2 = new Event(date1,45,"Площадь Конституции",listOfPeople2);

LinkedList<Event> eventList = new LinkedList<Event>();

eventList.add(event1);

eventList.add(event2);

for (Event event : eventList) {

event.outputData();

System.out.println();

}

}

}

**Висновки**

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з об’єктно-орієнтованою декомпозицією.

Програма протестована, виконується без помилок.