МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №14

з дисципліни
«Об’єктно-орієнтоване програмування»

Виконав:
студент групи КН-107
Шиманський П.С.
Прийняв:
Старший викладач
Гасько Р.Т.

Львів – 2018 р.

Результат роботи програми:

1)

```
package week14;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.Collections;
import java.util.LinkedList;
public class lab14 implements Serializable {
   private static  LinkedList<TradePoint> list =  new LinkedList<>();
   private static int find(TradePoint value){
      return list.indexOf(value);
   }
   private static void remove(TradePoint value){
      list.remove(value);
   }
   private static void add(TradePoint value){
      list.add(value);
   }
   private static void sort(LinkedList<TradePoint> list){
      Collections.sort(list, (object1, object2) ->
object1.name.getName().compareTo(object2.name.getName()));
   }
   public static void main(String[] args) throws IOException {
      TradePoint tradePoint = new TradePoint( "Bob", "Adress",123,
"Programmer", "Monday", "20:00");
      TradePoint secondPoint = new TradePoint( "Andrew",
"MainStreet",880553535, "Trader", "Sunday", "18:00");
      TradePoint thirdPoint = new TradePoint( "Joe",
"SalmanelaStreet",742625725, "Cook", "Tuesday", "17:00");
      TradePoint forthPoint = new TradePoint( "Anton", "RynokStreet",450236234,
"Manger", "Wednesday", "9:00");
      add(tradePoint);
      add(secondPoint);
      add(thirdPoint);
      add(forthPoint);
      for (Object i: list)
        System.out.println(i);
      sort(list);
      System.out.println();
      for (Object i: list)
         System.out.println(i);
      remove(thirdPoint);
      for (Object i: list)
```

```
            System.out.println(i);
        System.out.println("Find index of tradepoint " + find(tradePoint));
        FileOutputStream fos = new FileOutputStream("temp.out");
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(list);
        oos.flush();
        oos.close();
    }
}
```

2)

```
package week14;

import java.io.Serializable;
import java.util.*;
class Name implements Serializable{
    public String name;
    Name(String name){
        this.name = name;
    }
    public void  setName(String name){
        this.name = name;
    }
    public String getName(){
        return name;
    }
}
class Adress implements Serializable{
    public String adress;
     Adress(String adress){
         this.adress = adress;
    }
    public void setAdress(String adress){
        this.adress = adress;
    }
    public String getAdress(){
        return adress;
    }
}
class Number implements Serializable{
    public ArrayList<Integer> numbers;
    Number(int number){
        this.numbers = new ArrayList<>();
        numbers.add(number);
    }
```

```
    public void addNumbers(int newNumber){
        numbers.add(newNumber);
    }
    public ArrayList getNumbers(){
        return numbers;
    }
    @Override
    public String toString(){
        return numbers.toString();
    }
}
class Specialization implements Serializable{
    public String specialization;
    Specialization(String specialization){
        this.specialization = specialization;
    }
    public void setSpecialization(String specialization){
        this.specialization = specialization;
    }
    public String getSpecialization(){
        return specialization;
    }
}
class Schedule implements Serializable{
    public HashMap<String,String> schedule;
    Schedule(String day, String time){
        this.schedule = new HashMap<>();
        schedule.put(day,time);
    }
    public void addNewWorkingDay(String day, String time){
        schedule.put(day,time);
    }
    public void setSchedule(String day, String time){
        schedule.replace(day,time);
    }
    public void ShowSchedule(){
        for(Map.Entry<String, String> x : schedule.entrySet()){
            System.out.println(x.getKey() + " " + x.getValue());
        }
    }
    public  HashMap<String, String> getSchedule() {
        return schedule;
    }
    @Override
    public String toString(){
```

```
        return schedule.toString();
    }
}

class TradePoint implements Serializable {
    Name name;
    Adress adress;
    Specialization specialization;
    Number number;
    Schedule schedule;
    public TradePoint( String name1, String adress1, int number1, String
specialization1, String day, String time){
        this.name = new Name(name1);
        this.adress = new Adress(adress1);
        this.specialization = new Specialization(specialization1);
        this.number = new Number(number1);
        this.schedule = new Schedule(day,time);
    }

    public String ShowAllData(){
        String str = name.getName() + " " + adress.getAdress() + " " +
specialization.getSpecialization() + " " + number + " " + schedule;
        return str;
    }
    @Override
    public String toString(){
        return ShowAllData();
    }
}
public class Main {
    public static void main(String args[]) {
        TradePoint tradePoint = new TradePoint( "Name", "Adress",123,
"Programmer", "Monday", "20:00");
        TradePoint secondPoint = new TradePoint( "Andrew",
"MainStreet",880553535, "Trader", "Sunday", "18:00");
        System.out.println(tradePoint);
        System.out.println(secondPoint);
    }
}
```

Результат роботи програми:

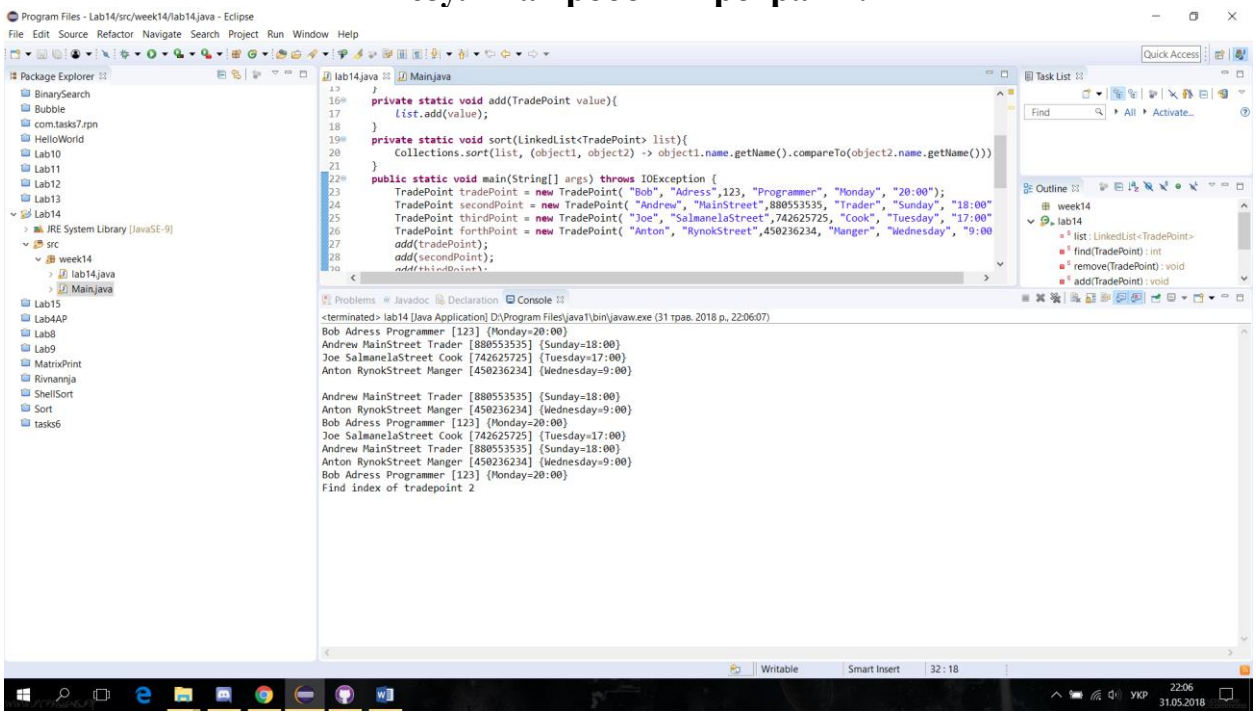

Рис.1 Результат роботи програми