**Звіт**

**Лабораторна работа 15. Колекції в Java**

**Мета роботи**:

* Ознайомлення з бібліотекою колекцій Java SE.
* Використання колекцій для розміщення об'єктів розроблених класів.

**ВИМОГИ**

1. Розробити консольну програму для реалізації завдання обробки даних згідно прикладної області.
2. Для розміщення та обробки даних використовувати контейнери (колекції) і алгоритми з Java Collections Framework.
3. Забезпечити обробку колекції об'єктів: додавання, видалення, пошук, сортування згідно розділу Прикладні задачі л.р. №10.
4. Передбачити можливість довготривалого зберігання даних: 1) за допомогою стандартної серіалізації; 2) не використовуючи протокол серіалізації.
5. Продемонструвати розроблену функціональність в діалоговому та автоматичному режимах за результатом обробки параметрів командного рядка.
   1. **Розробник**: Момот Роман Євгенійович, КІТ119а, варіант №14.
6. **ОПИС ПРОГРАМИ**
   1. **Засоби ООП**: клас, метод класу, поле класу.
   2. **Ієрархія та структура класів:** один публічний клас **Main**, публічний клас **Event**, у полях якого є час початку події, тривалість, адреса події, імена людей, опис події, гетери, сетери, конструктор класу та метод виведення даних класу. Клас **MyThread**, який виконує роль потока.
   3. **Важливі фрагменти програми:**

public class Main {

public static void main(String[] args) {

ArrayList<Event> arr = new ArrayList<Event>();

for(String str: args)

{

if(str.equals("-a") || str.equals("-auto")) {

arr = auto(arr);

return;

}

else if(str.equals("-d") || str.equals("-dialog")) {

arr = menu(arr);

return;

}

}

arr = menu(arr);

}

private static ArrayList<Event> auto(ArrayList<Event> arr) {

Pattern pattern;

Matcher matcher;

System.out.println("\nSize of container: " + arr.size());

System.out.println("Adding elements...");

ArrayList<String> people = new ArrayList<String>();

people.add("John");

people.add("Bill");

people.add("Івасик");

Event event = new Event(new GregorianCalendar(2019,4,28), 120, "Харьков, ул. Заозёрская 39",

people, "Лучшая тусовка");

arr.add(event);

people = new ArrayList<String>();

people.add("Roman");

people.add("Dmitriy");

event = new Event(new GregorianCalendar(2005,12,15), 30, "Харьков, пр. Тракторостроителей",

people, "Скучно");

arr.add(event);

System.out.println("Size of container: " + arr.size());

System.out.println("\nOutputing data with toArray:");

Object[] tempArr = arr.toArray();

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

System.out.println(i+1 + ")");

((Event)tempArr[i]).outputData();

System.out.println( );

}

System.out.println("Is container empty?");

System.out.println(arr.isEmpty());

System.out.println("\nReading data from file...");

try(BufferedReader reader = new BufferedReader(new InputStreamReader(new FileInputStream("data.txt"), "UTF-8"))){

int i = 0;

String str;

String[] data;

String[] patterns = {"^(?!^0)\\d{4}$", "^([1-9])|([1][0-2])$", "^([1-9])|([12][0-9])|([3][01])$",

"^([0-9])|([1][0-9])|([2][0-4])$", "^([0-9])|([1-5][0-9])|([6][0])$", "^(?!^0)\\d{1,9}$",

"^(([A-Z][a-z]+)|([A-Z][a-z]\*)([\\s][A-Z][a-z]\*))$"};

while((str = reader.readLine()) != null) {

data = str.split("\\s\*(;)\\s\*");

for(i = 2; i < 9; i++) {

pattern = Pattern.compile(patterns[i-2]);

matcher = pattern.matcher(data[i]);

if(!matcher.matches()) {

System.out.println("Wrong data in line. Moving to next line.");

i = 10;

}

}

if(i == 11) {

continue;

}

people.clear();

pattern = Pattern.compile(patterns[6]);

i--;

for (; i < data.length; i++) {

matcher = pattern.matcher(data[i]);

if(!matcher.matches()) {

System.out.println("Wrong name " + data[i] + " in line. It wont be added.");

}

else {

people.add(data[i]);

}

}

arr.add(new Event(new GregorianCalendar(Integer.parseInt(data[2]),Integer.parseInt(data[3]),Integer.parseInt(data[4]),Integer.parseInt(data[5]),Integer.parseInt(data[6]),0), Integer.parseInt(data[7]),data[0],people,data[1]));

}

}

catch(IOException ex) {

System.out.println(ex.getMessage());

}

System.out.println("\nOutputing data with toArray:");

tempArr = arr.toArray();

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

System.out.println(i+1 + ")");

((Event)tempArr[i]).outputData();

System.out.println( );

}

Pattern pattYear = Pattern.compile("^(2019)|(2018)|(2020)$");

Pattern pattCity = Pattern.compile("Харьков(.\*)");

Pattern pattDuration = Pattern.compile("^([2][5-9]+)|([3-9][0-9]+)|([1-9][0-9]{2,})$");

Matcher matcher1, matcher2, matcher3;

System.out.println("Outputting array with regex...\n");

for(var i : arr) {

matcher1 = pattYear.matcher(Integer.toString(i.getStartTime().get(Calendar.YEAR)));

matcher2 = pattCity.matcher(i.getAddress());

matcher3 = pattDuration.matcher(Integer.toString(i.getDuration()));

System.out.println( );

if(matcher1.matches() && matcher2.matches() && matcher3.matches()) {

i.outputData();

}

}

return arr;

}

private static ArrayList<Event> menu(ArrayList<Event> arr) {

Scanner scan = new Scanner(System.in);

boolean stop = false;

int choise, choise2;

ArrayList<String> people = new ArrayList<String>();

people.add("John");

people.add("Bill");

people.add("Івасик");

Event evToCompare = new Event(new GregorianCalendar(2002,3,28), 120, "Харьков, ул. Революции 67",

people, "Pest party ever");

arr.add(evToCompare);

do {

System.out.println("What to do?");

System.out.println("1. Output data");

System.out.println("2. Add element");

System.out.println("3. Delete element");

System.out.println("4. Is empty?");

System.out.println("5. Serialization");

System.out.println("6. Deserialization");

System.out.println("7. Sort data");

System.out.println("8. Find the number of people (Multithreading)");

System.out.println("9. Terminate program");

System.out.println("=====================");

System.out.print("Your choise: ");

choise = scan.nextInt();

switch(choise) {

case 1:

System.out.println("\nChoose the output method");

System.out.println("1. Using foreach");

System.out.println("2. Using toArray");

System.out.println("3. Find element by criteria");

System.out.println("4. Return");

System.out.println("================");

System.out.print("Your choise: ");

choise2 = scan.nextInt();

System.out.println( );

switch(choise2) {

case 1:

if(arr.size() > 0){

for(var i : arr) {

i.outputData();

System.out.println( );

}

}

else {

System.out.println("Array is empty.\n");

}

break;

case 2:

if(arr.size() > 0) {

Object[] tempArr = arr.toArray();

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

System.out.println(i+1 + ")");

((Event)tempArr[i]).outputData();

System.out.println();

}

}

else {

System.out.println("Array is empty.\n");

}

break;

case 3:

if(arr.size() == 0) {

System.out.println("Array is empty.\n");

break;

}

Pattern pattYear;

Pattern pattCity;

Pattern pattDuration = Pattern.compile("^([2][5-9]+)|([3-9][0-9]+)|([1-9][0-9]{2,})$");

Matcher matcher1, matcher2, matcher3;

String regex = "^(?)|(?)|(?)$";

System.out.println("Task: Знайти всі конференції, що пройшли\n"

+ "-за останні три роки "

+ "\n-в Харкові та області "

+ "\n-з тривалістю не менше доби.");

System.out.println("Передбачити можливість незначної зміни умов пошуку.");

System.out.print("\nEnter the year: ");

int year = scan.nextInt();

for (int i = 0; i < 3; i++) {

regex = regex.substring(0,regex.indexOf('?')) + Integer.toString(year - i) + regex.substring(regex.indexOf('?') + 1, regex.length());

}

pattYear = Pattern.compile(regex);

System.out.print("Enter the city: ");

scan.nextLine();

String city = scan.nextLine();

city = city.concat("(.\*)");

pattCity = Pattern.compile(city);

for(var i : arr) {

matcher1 = pattYear.matcher(Integer.toString(i.getStartTime().get(Calendar.YEAR)));

matcher2 = pattCity.matcher(i.getAddress());

matcher3 = pattDuration.matcher(Integer.toString(i.getDuration()));

System.out.println( );

if(matcher1.matches() && matcher2.matches() && matcher3.matches()) {

i.outputData();

}

}

break;

case 4:

System.out.println("Returning\n");

break;

default:

System.out.println("You've entered the wrong number");

break;

}

break;

case 2:

Event newEvent = inputNewEvent();

arr.add(newEvent);

break;

case 3:

if(arr.size() > 0) {

System.out.print("\nEnter the index of element: ");

choise = scan.nextInt();

try {

arr.remove(choise-1);

}

catch(IndexOutOfBoundsException ex) {

System.out.println("Error. Wrong id.\n");

break;

}

System.out.println("Element was deleted.\n");

} else {

System.out.println("\nArray is empty.\n");

}

break;

case 4:

if(arr.isEmpty()) {

System.out.println("\nArray is empty.\n");

} else {

System.out.println("\nArray isn't empty.\n");

}

break;

case 5:

System.out.println("\nChoose the method");

System.out.println("1. Standard serialization");

System.out.println("2. XML serialization");

System.out.println("3. Return");

System.out.println("=========================");

System.out.print("Your choise: ");

choise2 = scan.nextInt();

switch(choise2) {

case 1:

scan.nextLine();

System.out.print("\nEnter the name of file: ");

String filename = scan.nextLine();

if (filename.indexOf(".ser") == -1) {

filename += ".ser";

}

try(ObjectOutputStream oos = new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream(filename)))){

oos.writeObject(arr);

System.out.println("Serialization successful.");

}catch(Exception ex){

System.out.println(ex.getMessage());

ex.printStackTrace();

}

break;

case 2:

scan.nextLine();

System.out.print("\nEnter the name of file: ");

filename = scan.nextLine();

if (filename.indexOf(".xml") == -1) {

filename += ".xml";

}

try(XMLEncoder encoder = new XMLEncoder(new BufferedOutputStream(new FileOutputStream(filename)))){

encoder.writeObject(arr);

System.out.println("Serialization successful.");

}

catch(Exception ex){

System.out.println(ex.getMessage());

}

break;

case 3:

break;

default:

System.out.println("You've entered the wrong command.");

break;

}

break;

case 6:

System.out.println("\nChoose the method");

System.out.println("1. Standard deserialization");

System.out.println("2. XML deserialization");

System.out.println("3. Return");

System.out.println("=========================");

System.out.print("Your choise: ");

choise2 = scan.nextInt();

switch(choise2) {

case 1:

scan.nextLine();

System.out.print("\nEnter the name of file: ");

String filename = scan.nextLine();

if (filename.indexOf(".ser") == -1) {

filename += ".ser";

}

try(ObjectInputStream oos = new ObjectInputStream(new BufferedInputStream(new FileInputStream(filename)))){

arr.clear();

arr = (ArrayList<Event>) oos.readObject();

System.out.println("\nSerialization successful.");

}catch(Exception ex){

System.out.println(ex.getMessage());

}

break;

case 2:

scan.nextLine();

System.out.print("\nEnter the name of file: ");

filename = scan.nextLine();

if (filename.indexOf(".xml") == -1) {

filename += ".xml";

}

try(XMLDecoder decoder = new XMLDecoder(new BufferedInputStream(new FileInputStream(filename)))){

arr.clear();

arr = (ArrayList<Event>) decoder.readObject();

System.out.println("Serialization successful.\n");

}catch(IOException ex){

System.out.println( );

}

break;

case 3:

break;

default:

System.out.println("You've entered the wrong command.");

break;

}

break;

case 7:

if(arr.size() != 0) {

System.out.println("\nChoose sorting field:");

System.out.println("1. Sort by event date");

System.out.println("2. Sort by event length");

System.out.println("3. Sort by number of people");

System.out.println("4. Return");

System.out.println("===========================");

System.out.print("Your choise: ");

choise2 = scan.nextInt();

switch(choise2) {

case 1:

arr.sort(new EventDateComparator());

System.out.println("\nData sorted\n");

break;

case 2:

arr.sort(new EventLengthComparator());

System.out.println("\nData sorted\n");

break;

case 3:

arr.sort(new EventPeopleNumberComparator());

System.out.println("\nData sorted\n");

break;

case 4:

System.out.println("\nReturning\n");

break;

default:

System.out.println("\nYou have entered the wrong number.\n");

break;

}

}

else {

System.out.println("\nArray is empty.\n");

}

break;

case 8:

final int ARR\_SIZE = 5000000;

final int NUBER\_OF\_THREADS;

final int NUMBER\_OF\_ITERATIONS;

int time = -1;

int numberOfPeople;

int choise3;

long time1, time2;

ArrayList<Event> newArr = new ArrayList<Event>();

ArrayList<String> newPeople = new ArrayList<String>();

System.out.println("\nCreating new array...");

System.out.println("Adding new elements...");

for(int i = 0; i < ARR\_SIZE; i++) {

numberOfPeople = (int) (2 + Math.random() \* 10);

for(int j = 0; j < numberOfPeople; j++) {

newPeople.add("j");

}

newEvent = new Event(new GregorianCalendar(), i, Integer.toString(i), newPeople, Integer.toString(i));

newArr.add(newEvent);

newPeople = new ArrayList<String>();

}

System.out.println("\nWhat do you want?");

System.out.println("1. Parallel calculations");

System.out.println("2. Serial calculations");

System.out.println("=====================");

System.out.print("What do you want: ");

choise3 = scan.nextInt();

System.out.println( );

if(choise3 != 1 && choise3 != 2) {

System.out.println("You have entered the wrong command");

break;

}

// System.out.print("Want to set a maximum lead time? ");

// if((choise3 = scan.nextBoolean()) == true) {

// System.out.print("Enter the time in milliseconds: ");

// time = scan.nextInt();

// }

if(choise3 == 1) {

NUBER\_OF\_THREADS = 3;

NUMBER\_OF\_ITERATIONS = 1;

}

else {

NUBER\_OF\_THREADS = 1;

NUMBER\_OF\_ITERATIONS = 3;

}

MyThread[] threads = new MyThread[NUBER\_OF\_THREADS];

//System.out.println(newArr.size());

try {

for(int i = 0; i < NUBER\_OF\_THREADS; i++) {

threads[i] = new MyThread(newArr, "Parallel thread " + (i+1), NUMBER\_OF\_ITERATIONS);

threads[i].thread.start();

}

// if(time > 0) {

// Thread.currentThread().sleep(time);

//

// for(int i = 0; i < NUBER\_OF\_THREADS; i++) {

// threads[i].disable();

// }

// }

time1 = System.currentTimeMillis();

for(int i = 0; i < NUBER\_OF\_THREADS; i++) {

threads[i].thread.join();

}

time2 = System.currentTimeMillis();

//System.out.println(time1 + "\t" + time2);

System.out.println("Time result: " + (double)(time2 - time1)/1000 + " seconds");

}

catch(InterruptedException ex) {

System.out.println("Thread has been interrupted.");

}

//newArr.clear();

System.out.println( );

break;

case 9:

System.out.println("\nTerminating the program.");

stop = true;

break;

default:

System.out.println("You have entered the wrong number.");

break;

}

}while(!stop);

scan.close();

return arr;

}

private static Event inputNewEvent(){

Pattern pattSurname = Pattern.compile("^(([A-Z][a-z]+)|([A-Z][a-z]\*)([\\s][A-Z][a-z]\*))$");

Pattern pattYear = Pattern.compile("^(?!^0)\\d{4}$");

Pattern pattMonth = Pattern.compile("^([1-9])|([1][0-2])$");

Pattern pattDay = Pattern.compile("^([1-9])|([12][0-9])|([3][01])$");

Pattern pattTime = Pattern.compile("^(?!^0)\\d{1,9}$");

Pattern pattHour = Pattern.compile("^([0-9])|([1][0-9])|([2][0-4])$");

Pattern pattMinute = Pattern.compile("^([0-9])|([1-5][0-9])|([6][0])$");

Matcher matcher;

Scanner scan = new Scanner(System.in);

int value;

ArrayList<String> list = new ArrayList<String>();

String temp;

GregorianCalendar date = new GregorianCalendar();

boolean ready = false;

do {

System.out.print("\nEnter number of participants: ");

value = scan.nextInt();

if(value < 1) {

System.out.println("Error. Wrong list size.\n");

}

ready = true;

}while(!ready);

System.out.println("Enter list of names:");

scan.nextLine();

ready = false;

for (int i = 0; i < value; i++) {

System.out.print(i+1 + ". ");

temp = scan.nextLine();

do {

matcher = pattSurname.matcher(temp);

if(!matcher.matches()) {

System.out.print("Wrong name format.\nEnter new surname: ");

temp = scan.nextLine();

}

ready = true;

}while(!ready);

list.add(temp);

}

System.out.print("Enter event year: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattYear.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong year.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

date.set(Calendar.YEAR, value);

System.out.print("Enter event month: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattMonth.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong month.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

date.set(Calendar.MONTH, value-1);

System.out.print("Enter event day: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattDay.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong day.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

date.set(Calendar.DAY\_OF\_MONTH, value);

System.out.print("Enter event hour: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattHour.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong hour.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

date.set(Calendar.HOUR\_OF\_DAY, value);

System.out.print("Enter event minute: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattMinute.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong minute.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

date.set(Calendar.MINUTE, value);

date.set(Calendar.SECOND, 0);

System.out.print("Enter event address: ");

scan.nextLine();

temp = scan.nextLine();

System.out.print("Enter event description: ");

String description = scan.nextLine();

System.out.print("Enter event length: ");

value = scan.nextInt();

ready = false;

do {

matcher = pattTime.matcher(Integer.toString(value));

if(!matcher.matches()) {

System.out.print("You've entered the wrong event length.\nTry Again: ");

value = scan.nextInt();

}

else {

ready = true;

}

}while(!ready);

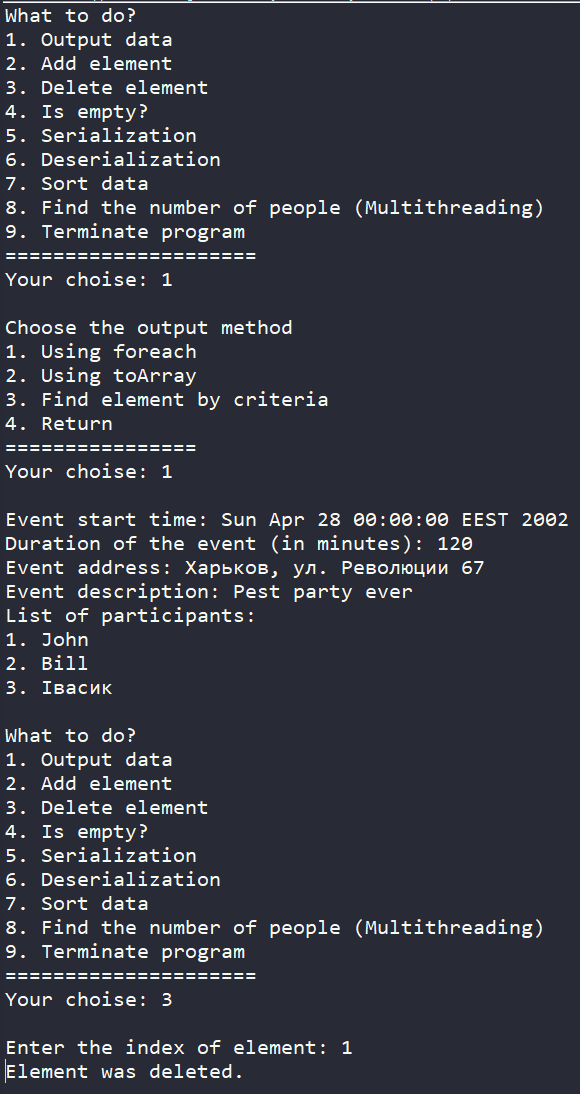
System.out.println("\nEvent added.\n");

Event newEvent = new Event(date,value,temp,list,description);

return newEvent;

}

}

**Результат роботи програми**

**Висновки**

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з колекціями.

Програма протестована, виконується без помилок.