**Звіт**

**Лабораторна работа 11. Регулярні вирази. Перевірка даних**

**Мета роботи**: Ознайомлення з принципами використання регулярних виразів для перевірки рядка на відповідність шаблону.

**ВИМОГИ**

Продемонструвати ефективне (оптимальне) використання регулярних виразів для перевірки коректності (валідації) даних, що вводяться, перед записом в domain-об'єкти відповідно до призначення кожного поля для заповнення розробленого контейнера:

* при зчитуванні даних з текстового файла в автоматичному режимі;
* при введенні даних користувачем в діалоговому режимі.
  1. **Розробник**: Момот Роман Євгенійович, КІТ119а, варіант №14.

1. **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. Також є клас Node, який виконує роль покажчика на елемент і клас MyContainer, який містить покажчик на головний елемент та методи обробки масиву елементів.
   3. **Важливі фрагменти програми:**

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

Pattern pattern;

Matcher matcher;

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

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(28,4,2002), 120, "ул. Революции",

people, "Pest party ever");

arr.add(event);

people = new ArrayList<String>();

people.add("Roman");

people.add("Dmitriy");

event = new Event(new GregorianCalendar(15,10,2020), 30, "пр. Тракторостроителей",

people, "So boring");

arr.add(event);

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

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( );

}

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;

}

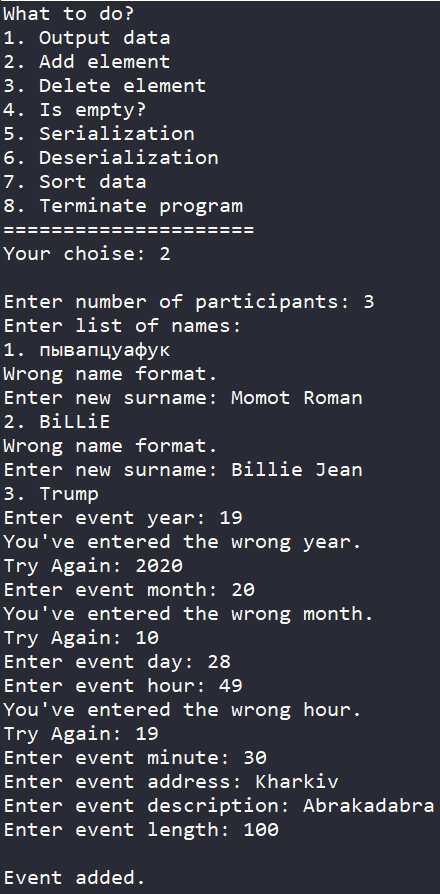
}while(!ready);

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 регулярними виразами.

Програма протестована, виконується без помилок.