

## 12. Регулярні вирази. Обробка тексту

- **Мета:** Ознайомлення з принципами використання регулярних виразів для обробки тексту.

### 1 ВИМОГИ

#### 1.1 Розробник

Інформація про розробника:

- Шарма Олександр Раджнішович
- НТУ “ХПІ” КІТ-1196
- Варіант 2 (25)

#### 1.2 Загальне завдання

1. Використовуючи програми рішень попередніх задач, продемонструвати ефективне (оптимальне) використання регулярних виразів при вирішенні [прикладної задачі](#).
2. Передбачити можливість незначної зміни умов пошуку.
3. Продемонструвати розроблену функціональність в діалоговому та автоматичному режимах.

#### 1.3 Задача

[Кадрове агентство](#). Знайти всі вакансії, де потрібні викладачі (педагоги, вчителі) зі стажем не менше 10 років, які знають англійську мову та володіють автомобілем.

### 2 ОПИС ПРОГРАМИ

#### 2.1 Засоби ООП

Композиція, інкапсуляція.

```
classDiagram
    class Agency {
        <<Java Class>>
        +String firmName
        +String position
        +String circs
        +int salary
        +boolean key
        +Agency()
        +getFirmName() String
        +getPosition() String
        +getCircs() String
        +getSalary() int
        +getKey() boolean
        +setFirmName(String) void
        +setPosition(String) void
        +setCircs(String) void
        +setSalary(int) void
        +setKey(boolean) void
        +getReqs() Requirments
        +setReqs(Requirments) void
        +isNeedRequirments(int, String) void
        +print() void
        +writeExternal(ObjectOutput) void
        +readExternal(ObjectInput) void
    }

    class Requirments {
        <<Java Class>>
        +long serialVersionUID
        +Integer yexp
        +String education
        +Requirments()
        +getYexp() Integer
        +setYexp(int) void
        +getEducation() String
        +setEducation(String) void
        +print() void
        +writeExternal(ObjectOutput) void
        +readExternal(ObjectInput) void
    }

    class Helper {
        <<Java Class>>
        +Helper()
        +lets go(int) void
        +print main menu() void
        +showAgencies(List<Agency>) void
        +enterFirmName() String
        +enterPosition() String
        +enterCircs() String
        +enterSalary() int
        +enterKey() boolean
        +enterExperience() int
        +enterEducation() String
        +get_data(List<Agency>) void
        +directory() String
        +writeFileName() String
        +validateText(String) boolean
        +validateInt(String) boolean
        +matchByCondition(List<Agency>) void
    }

    class List_T {
        <<Java Class>>
        +List<T>
        +_size: int
        +List()
        +getSize() int
        +add(T) void
        +add(T[]) void
        +remove(T) void
        +toArray() Agency[]
        +toString(List<Agency>) String
        +getHead() Node<T>
        +getTail() Node<T>
        +iterator() Iterator<T>
        +compareFirmNames(T, T) boolean
        +compareEducation(T, T) boolean
        +compareCircs(T, T) boolean
        +sortList() void
    }

    class ListIterator_T {
        <<Java Class>>
        +ListIterator(List<T>)
        +hasNext() boolean
        +next()
        +remove() void
    }

    class Node_T {
        <<Java Class>>
        +Node<T>
        +data: T
        +Node(T, Node<T>)
        +setData(T) void
        +setNext(Node<T>) void
        +getData()
        +getNext() Node<T>
    }

    class Main {
        <<Java Class>>
        +Main()
        +main(String[]) void
    }

    class Serialization {
        <<Java Class>>
        +Serialization()
        +LongTermPersistenceWrite(List<Agency>, int) int
        +LongTermPersistenceRead(int, int) List<Agency>
        +SerializationA(List<Agency>, String) void
        +DeserializationA(String) List<Agency>
    }

    Agency "0..1" -- "0..1" Requirments : -reqs
    Agency --> List_T : +lets go
    List_T --> ListIterator_T : +iterator
    List_T --> Node_T : +getHead
    List_T --> Node_T : +getTail
    Node_T --> Node_T : ~next
    Main --> Helper : +main
    Helper --> List_T : +showAgencies
    Helper --> List_T : +get_data
    Helper --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T : +enterKey
    List_T --> List_T : +enterExperience
    List_T --> List_T : +enterEducation
    List_T --> List_T : +validateText
    List_T --> List_T : +validateInt
    List_T --> List_T : +directory
    List_T --> List_T : +writeFileName
    List_T --> List_T : +lets go
    List_T --> List_T : +print main menu
    List_T --> List_T : +showAgencies
    List_T --> List_T : +get_data
    List_T --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T : +enterKey
    List_T --> List_T : +enterExperience
    List_T --> List_T : +enterEducation
    List_T --> List_T : +validateText
    List_T --> List_T : +validateInt
    List_T --> List_T : +directory
    List_T --> List_T : +writeFileName
    List_T --> List_T : +lets go
    List_T --> List_T : +print main menu
    List_T --> List_T : +showAgencies
    List_T --> List_T : +get_data
    List_T --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T : +enterKey
    List_T --> List_T : +enterExperience
    List_T --> List_T : +enterEducation
    List_T --> List_T : +validateText
    List_T --> List_T : +validateInt
    List_T --> List_T : +directory
    List_T --> List_T : +writeFileName
    List_T --> List_T : +lets go
    List_T --> List_T : +print main menu
    List_T --> List_T : +showAgencies
    List_T --> List_T : +get_data
    List_T --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T : +enterKey
    List_T --> List_T : +enterExperience
    List_T --> List_T : +enterEducation
    List_T --> List_T : +validateText
    List_T --> List_T : +validateInt
    List_T --> List_T : +directory
    List_T --> List_T : +writeFileName
    List_T --> List_T : +lets go
    List_T --> List_T : +print main menu
    List_T --> List_T : +showAgencies
    List_T --> List_T : +get_data
    List_T --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T : +enterKey
    List_T --> List_T : +enterExperience
    List_T --> List_T : +enterEducation
    List_T --> List_T : +validateText
    List_T --> List_T : +validateInt
    List_T --> List_T : +directory
    List_T --> List_T : +writeFileName
    List_T --> List_T : +lets go
    List_T --> List_T : +print main menu
    List_T --> List_T : +showAgencies
    List_T --> List_T : +get_data
    List_T --> List_T : +matchByCondition
    List_T --> List_T : +sortList
    List_T --> List_T : +compareFirmNames
    List_T --> List_T : +compareEducation
    List_T --> List_T : +compareCircs
    List_T --> List_T : +toArray
    List_T --> List_T : +remove
    List_T --> List_T : +add
    List_T --> List_T : +getSize
    List_T --> List_T : +toString
    List_T --> List_T : +setFirmName
    List_T --> List_T : +setPosition
    List_T --> List_T : +setCircs
    List_T --> List_T : +setSalary
    List_T --> List_T : +setKey
    List_T --> List_T : +getReqs
    List_T --> List_T : +setReqs
    List_T --> List_T : +isNeedRequirments
    List_T --> List_T : +print
    List_T --> List_T : +writeExternal
    List_T --> List_T : +readExternal
    List_T --> List_T : +enterFirmName
    List_T --> List_T : +enterPosition
    List_T --> List_T : +enterCircs
    List_T --> List_T : +enterSalary
    List_T --> List_T :
```

## 2.3 Важливі фрагменти програми

```
private static void matchByCondition(List<Agency> agencies) {
    Pattern p = Pattern.compile("^([0,]{0,}[1-9]{2,})");
    for (Agency a : agencies) {
        Matcher m = null;
        if(a.getKey()) {m = p.matcher(a.getReqs().getYexp().toString());}
        else { return;}
        if(m.matches() == true) {
            Pattern n = Pattern.compile("^(teacher|trainer)");
            m = n.matcher(a.getPosition());
            if(m.matches() == true) {
                Pattern z = Pattern.compile(".*(English skills | English | Knowledge of English | level of English)\\w*");
                m = z.matcher(a.getReqs().getEducation());
                a.print();
            }
        }
    }
}
```

Рисунок 2 – валідація даних

### 3 ВАРІАНТИ ВИКОРИСТАННЯ

Програма створена для роботи з прикладною задачею. Для коректної роботи були реалізовані методи введення та отримання даних, також дані приховані від користувача, щоб не порушувати суттєвість об'єкту.

```
10.Exit
9
Информация об этой фирме:
Название фирмы : sadsadasd
Должность : teacher
Условия работы: asdsa
Зарплата : 3232
Дополнительные требования:
Опыт работы : 22
Образование : English
1.Enter data
2.Show current data
3.Remove tail
4.Save data
```

Рисунок 3 – результати валідації

## **ВИСНОВКИ**

В даній лабораторній роботі було розроблено методи валідації даних за допомогою регулярних виразів. Набуто навичок об'єктно-орієнтованого підходу та було розширено параметризацію.