# **МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ»**



##### **Факультет управления и информатики в технологических системах**

###### **Кафедра Информационной безопасности**

**Направление подготовки**

**(специальность) 10.05.03 Информационная безопасность автоматизированных систем**

**Отчет**

**по практике по технологиям и методам программирования**

наименование (вид) практики

Выполнил студент гр. УБ-42

Крылов Никита Романович

###### (Ф.И.О.)

###### \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись)*

###### Проверили:

Маслов А.А.

*(Ф.И.О.)*

*(оценка) (подпись)*

*(дата)*

Воронеж - 2025

**Задание 2.**

Написать регулярное выражение, определяющее является ли данная строка GUID с или без   
скобок. Где GUID это строчка, состоящая из 8, 4, 4, 4, 12   
шестнадцатеричных цифр разделенных тире.   
Пример правильных выражений: e02fd0e4-00fd-090A-ca30-0d00a0038ba0.   
Пример неправильных выражений: e02fd0e400fd090Aca300d00a0038ba0.

Код программы.  
package *K*urs\_2.*P*r2;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.regex.Pattern;

public class Var2 {

private static final Pattern **GUID\_PATTERN** = Pattern.compile(

"^(?:\\{[0-9a-fA-F]{8}(?:-[0-9a-fA-F]{4}){3}-[0-9a-fA-F]{12}\\}|[0-9a-fA-F]{8}(?:-[0-9a-fA-F]{4}){3}-[0-9a-fA-F]{12})$"

);

public static boolean isGuid(String s) {

if (s == null) return false;

return **GUID\_PATTERN**.matcher(s).matches();

}

public static void main(String[] args) throws Exception {

BufferedReader reader = new BufferedReader(new InputStreamReader(System.**in**));

String line;

boolean anyInput = false;

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

anyInput = true;

boolean ok = isGuid(line.trim());

System.**out**.println(ok);

}

if (!anyInput) {

*// Пример использования:*

String[] examples = new String[] {

"{550e8400-e29b-41d4-a716-446655440000}", *// true*

"550e8400-e29b-41d4-a716-446655440000", *// true*

"{550e8400-e29b-41d4-a716-44665544000}", *// false (11 в конце)*

"550e8400e29b-41d4-a716-446655440000", *// false (нет первой -)*

"{550e8400-e29b-41d4-a716-446655440000" *// false (нет закрывающей скобки)*

};

for (String s : examples) {

System.**out**.printf("%s -> %s%n", s, isGuid(s));

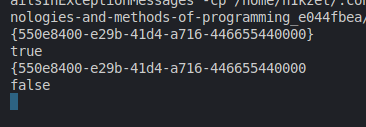
}

}

}

}

Вывод программы:



Вывод

Сегодня я изучил и научился решать задачу валидации строк по сложному шаблону с использованием регулярных выражений в Java. Данная программа проверяет, соответствует ли входная строка формату GUID (Globally Unique Identifier), который широко используется в программировании для генерации уникальных идентификаторов.

Изучив и реализовав это задание, я осознал мощь и эффективность регулярных выражений для обработки текстовой информации и проверки сложных форматов данных. Ключевым элементом программы является константа GUID\_PATTERN, в которой задается шаблон для проверки GUID. Этот шаблон использует синтаксис регулярных выражений для точного описания двух допустимых форматов: с фигурными скобками и без них.

Я научился работать с классом Pattern из пакета java.util.regex, который позволяет компилировать регулярные выражения для многократного использования. Метод Pattern.compile() создает объектPattern, а метод matcher().matches() проверяет полное совпадение всей строки с заданным шаблоном. Это важно, так как мы проверяем не наличие GUID в тексте, а то, что вся строка представляет собой корректный GUID.

Особенностью реализации является использование неточной группы (?:...) для объединения двух вариантов формата без создания capturing group, что улучшает производительность. Шаблон четко определяет структуру GUID:

* 8 шестнадцатеричных символов [0-9a-fA-F]{8}
* три группы по 4 шестнадцатеричных символа, разделенных дефисами (?:-[0-9a-fA-F]{4}){3}
* финальную группу из 12 шестнадцатеричных символов -[0-9a-fA-F]{12}
* опциональные фигурные скобки вокруг всего идентификатора

Я также освоил работу с потоком ввода через BufferedReader и InputStreamReader, что позволяет программе interactively обрабатывать ввод пользователя в реальном времени. Программа читает строки из стандартного ввода до тех пор, пока не получит null (что обычно означает конец ввода или закрытие потока). Если пользователь не предоставил никаких данных, программа демонстрирует работу на заранее подготовленных примерах, что очень удобно для тестирования.

Важной частью обучения стало понимание необходимости обработки крайних случаев: проверка на null, использование trim() для удаления лишних пробелов, и корректная обработка обеих форм записи GUID. Реализованный метод isGuid() является надежным и может быть легко интегрирован в larger projects для валидации идентификаторов.