**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования  
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ имени Н.Э.БАУМАНА  
(национальный исследовательский университет)»**

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

**Лабораторная работа № 3**

“Лексический анализатор на основе регулярных выражений”

по дисциплине «Конструирование компиляторов»

Вариант 17

Работу выполнил

студент группы ИУ9-62Б

Сербин Денис

Москва, 2022

# **1. Цель работы**

# Целью данной работы является приобретение навыка разработки простейших лексических анализаторов, работающих на основе поиска в тексте по образцу, заданному регулярным выражением.

# **2. Исходные данные**

# Стандартная библиотека любого современного языка программирования содержит средства для поиска в тексте образцов, заданных регулярными выражениями. При этом используется расширенный синтаксис записи регулярных выражений, позволяющий по сути выйти за рамки регулярных языков. Механизм поиска по таким регулярным выражениям годится для написания простейших лексических анализаторов. Однако, для этого механизма характерна нелинейная зависимость времени работы от длины распознаваемой лексемы, поэтому в промышленных компиляторах он не используется.

# В качестве языка реализации в данной лабораторной работе выберем язык Java, стандартная библиотека которого содержит пакет java.util.regex, в котором располагаются классы Pattern и Matcher, предназначенные для поиска по регулярным выражениям. Документацияпо этому пакету находится по адресу:

# http://docs.oracle.com/javase/7/docs/api/java/util/regex/package-summary.html. Вводную статью по синтаксису регулярных выражений можно прочитать здесь: <http://www.quizful.net/post/Java-RegExp>.

# Идея лексического анализа на основе поиска по регулярным выраженням состоит в использовании групп, представляющих собой фрагменты регулярных выражений, заключённые в круглые скобки, значения которых запоминаются при сопоставлении текста с образцом. На пример, на листинге 1 показано, как с использованием групп отличить идентификаторы от числовых литералов.

# **3. Задание**

В лабораторной работе необходимо реализовать на языке Java две первые фазы стадии анализа: чтение входного потока и лексический анализ. Чтение входного потока должно осуществляться из файла (в UTF-8), при этом лексический анализатор должен вычислять текущие координаты в обрабатываемом тексте. В результате работы программы в стандартный поток вывода должны выдаваться описания распознанных лексем в формате

Тег (координаты): значение

Например,

IDENT (1, 2): count

ASSIGN. (1, 8): :=

NUMBER (1, 11): 100

Лексемы во входном файле могут разделяться пробельными символами (пробел, горизонтальная табуляция, маркеры окончания строки), а могут быть записаны слитно (если это не приводит к противоречиям). Идентификаторы и числовые литералы не могут содержать внутри себя пробельных сим волов, если в задании явно указано иного (варианты 3. 5 и 34). Комментарии, строковые и символьные литералы могут содержать внутри себя пробельные символы. Входной файл может содержать ошибки. при обнаружении которых лексический анализатор должен выдавать сообщение с указанием координаты:

syntax error (10,2)

После обнаружения ошибки лексический анализатор должен восстанавливаться по следующей схеме: из входного потока пропускаются все подряд идущие символы до нахождения следующей лексемы.

Лексический анализатор должен иметь программный интерфейс для взаимодействия с парсером. Рекомендуется реализовывать его как итератор с методом nextToken () для императивных языков или функцию, возвращающую список лексем, для функциональных языков.

В регулярных выражениях рекомендуется использовать классы символов Unicode для обо значения букв, чисел и других подобных множеств. Многие движки регулярных выражений для задания классов используют синтаксис \р{класс}. Вместо нумерованных групп рекомендуется использовать именованные (?<имя>regexр), при использовании нумерованных групп ненумеруемые обозначаются как (?:regexp). Варианты языков для лексического анализа приведены в таблицах 1, 2, 3, 4 и 5.

**4. Индивидуальный вариант**

# Идентификаторы: последовательности буквенных символов Unicode и цифр, начинающиеся с буквы, не чувствительны к регистру. Целочисленные константы: десятичные последовательности десятичных цифр, шестнадцатеричные последовательности шестнадцатиричных цифр, начинающиеся на «&Н». Ключевые слова «PRINT», «GOTO», «GOSUB» без учёта регистра.

# **5. Результат выполнения**

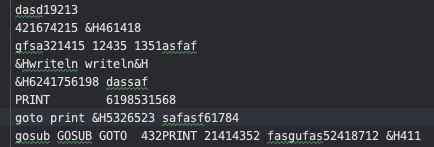


Рисунок 1. Текст входного потока

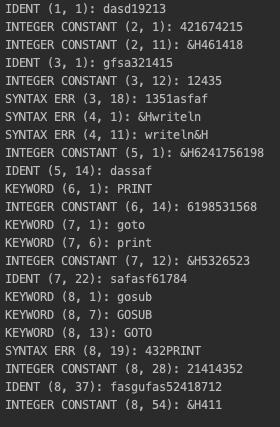


Рисунок 2. Лексический анализ

# **6. Вывод**

В рамках данной лабораторной работы были реализованы две первые фазы стадии анализа: чтение входного потока и лексический анализ.