Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Кафедра информационных технологий и систем

**Проектирование лексического анализатора**

Лабораторная работа

по дисциплине:

«Теория языков программирования и методы трансляции»

Разработал:

Студент группы 8091

Кудряшов И. С. \_\_\_\_\_\_

«\_\_».\_\_\_\_\_.2020г.

Проверил:

Макаров В. А. \_\_\_\_\_\_

«\_\_».\_\_\_\_\_.2020г.

**Великий Новгород**

**2020**

# Постановка задачи

## Формулировка

Разработать программу, решающую задачу лексического разбора исходного текста программы.

Результат работы лексического анализатора выводятся в файл в виде таблицы распознанных лексем. Таблица должна следующие поля содержать: номер лексемы и атрибуты лексемы (имя лексемы, класс лексемы (подкласс))

## Представление входных данных

Грамматика языка была составлена путём добавления ко 2 варианту из документа “Варианты грамматик 2020” некоторых ключевых слов.

Алфавит:**begin, end, true, false, if, then, while, for, do, until, not, or, and, eqv, implies, step, real, integer, print, ‘:=’, ‘,’, ‘;’, ‘<’, ‘<=’, ‘=’, ‘>=’, ’>’, ‘~=’**

Классы лексем:

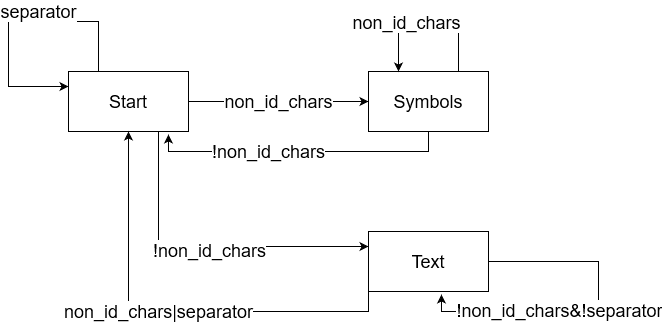
1. Ключевые слова языка (**begin, end, true, false, if, then, while, for, do, until, not, or, and, eqv, implies, step, real, integer, print**);
2. Знаки операций **(‘<’, ‘<=’, ‘=’, ‘>=’, ’>’, ‘~=’)** ;
3. Разделители **(‘,’, ‘;’);**
4. Знаки операции присваивания **(‘:=’)**;
5. Идентификаторы (могут состоять из всех символов за исключением, использованных в пунктах 2, 3, 4 и пробела со знаком перевода строки);
6. Целочисленная константа;
7. Константа с плавающей точкой;

## Описание выбранного метода решения задачи лексического анализа

В качестве основного метода лексического анализа был выбран конечный автомат, но для обеспечения возможности изменения алфавита без перекомпиляции программы конечный автомат используется ограниченно, для хранения лексем используются std::map и std::set, конечный результат сохраняется в std::vector.

# Описание реализации

Программа определяет 3 различных набора символов- символы разделители (пробел и перенос строки, всегда означают конец лексемы и не могут быть частью любой лексемы), специальные символы (использованные в знаках операций, разделителях или знаках присвоения) и все остальные. Появление каждого типа символов переключает состояние автомата, в соответствующее этому классу. Для разделителей это стартовое состояние. Для специальных символов символьное состояние. Для остальных символов текстовое состояние. При переходе из состояния, не являющегося стартовым, в любое другое последовательность символов, получившаяся за время нахождения в этом состоянии, ищется в списках лексем. По окончанию разбора файла для всех получившихся переменных выполняется проверка не являются ли они числами. Если являются их класс изменяется на числовой.



# Обработка ошибок

Если лексема не найдена, то для состояния символов лексема записывается как ошибочная, для состояния текста как переменная.

После обнаружения ошибочной лексемы сканирование файла продолжается.

# Методика тестирования

Для проверки работы анализатора, разработан следующий тестовый файл

Он небольшой так как было непонятно какую из версий языка реализовывать и отсутствовали математические операции в описании грамматики в файле “Варианты грамматик 2020”

begin

real a,b,c,d;

integer f;

f := 0;

a := 0;

for b := 0.1 step 0.23 until 3.541 do

begin

print b;

end;

end;

# Выводы

Мной была собрана грамматика, на основе представленной в файле “Варианты грамматик 2020” с добавлением трёх ключевых слов. Также мной был разработан лексический анализатор для этой грамматики, способный различать лексемы и числовые константы.