МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

САНКТ**-**ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**,** МЕХАНИКИ И ОПТИКИ

Кафедра Информатики и прикладной математики

Лабораторная работа №1 (8 вариант)

По предмету «Верификация моделей программ»

Выполнил:

студент гр. № Р4115

Назукин Д. Е.

Проверил :

Кореньков Ю. Д.

Санкт-Петербург

2017

**Цели**

Освоение верификации синтаксических конструкций текстовых представлений данных.

**Задачи**

Реализовать построение по исходному файлу с текстом синтаксического дерева с узлами, соответствующими правилам варианта, задающего язык для анализа. Вывести полученное дерево в файл.

**Описание работы**

Абстрактное синтаксическое дерево (АСД) — в [информатике](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) [конечное](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE) помеченное ориентированное [дерево](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)), в котором внутренние вершины сопоставлены (помечены) с [операторами](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) языка программирования, а листья — с соответствующими [операндами](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D0%BD%D0%B4). Таким образом, листья являются пустыми операторами и представляют только переменные и константы.

Синтаксические деревья используются в [парсерах](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7" \o "Синтаксический анализ) для [промежуточного представления](https://ru.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D0%BC%D0%B5%D0%B6%D1%83%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) программы между [деревом разбора](https://ru.wikipedia.org/w/index.php?title=%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D1%80%D0%B0%D0%B7%D0%B1%D0%BE%D1%80%D0%B0&action=edit&redlink=1) (конкретным синтаксическим деревом) и [структурой данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), которая затем используется в качестве внутреннего представления в [компиляторе](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) или [интерпретаторе](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80) [компьютерной программы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0) для оптимизации и генерации кода. Возможные варианты подобных структур описываются абстрактным синтаксисом.

**Аспекты реализации**

Для реализации построения абстрактного синтаксического дерева была использована библиотека ANTLR4 для языка Java.

ANTLR — [генератор](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80_%D0%BA%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D0%BE%D0%B2) [нисходящих](https://ru.wikipedia.org/wiki/%D0%9D%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D1%8F%D1%89%D0%B8%D0%B9_%D1%81%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7) [анализаторов](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%80%D1%81%D0%B5%D1%80) для формальных языков. ANTLR преобразует [контекстно-свободную грамматику](https://ru.wikipedia.org/wiki/LL-%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) в виде [РБНФ](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%88%D0%B8%D1%80%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D1%84%D0%BE%D1%80%D0%BC%D0%B0_%D0%91%D1%8D%D0%BA%D1%83%D1%81%D0%B0_%E2%80%94_%D0%9D%D0%B0%D1%83%D1%80%D0%B0) в программу на [C++](https://ru.wikipedia.org/wiki/C%2B%2B), [Java](https://ru.wikipedia.org/wiki/Java" \o "Java), [C#](https://ru.wikipedia.org/wiki/C_Sharp), [Python](https://ru.wikipedia.org/wiki/Python" \o "Python), [Ruby](https://ru.wikipedia.org/wiki/Ruby" \o "Ruby). Используется для разработки [компиляторов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80%D1%8B), [интерпретаторов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B0%D1%82%D0%BE%D1%80) и [трансляторов](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D1%81%D0%BB%D1%8F%D1%82%D0%BE%D1%80).

Ниже приведена часть грамматики:



После описания грамматики библиотека сгенерирует необходимые классы для парсинга текста и построения AST дерева.



**Результаты**

Вход программы файл с текстом:

function sto()  
 sto = 0;  
 do  
 sto = sto+1;  
 loop while (sto < 100)  
end function

Полученное дерево:

Source  
 SourceItem  
 FuncDef  
 FuncSignature  
 Identifier: sto  
 ListArgDef  
 ExpressionStatement  
 AssignExpr  
 PlaceExpr  
 Identifier: sto  
 LiteralExpr: 0  
 DoStatement  
 Type: while  
 ExpressionStatement  
 AssignExpr  
 PlaceExpr  
 Identifier: sto  
 BinaryExpr  
 PlaceExpr  
 Identifier: sto  
 BinOp: +  
 LiteralExpr: 1  
 BracesExpr  
 BinaryExpr  
 PlaceExpr  
 Identifier: sto  
 BinOp: <  
 LiteralExpr: 100

**Вывод**

В ходе работы реализовано построение по исходному файлу с текстом синтаксического дерева с узлами, соответствующими правилам варианта, задающего язык для анализа. Выведено полученное дерево в файл.