# УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники Направление подготовки 09.03.04 Программная инженерия Дисциплина «Системное программное обеспечение»

Лабораторная работа №1

## Студент

*Лещуков А. А. P4114*

## Преподаватель

*Кореньков Ю. Д.*

Санкт-Петербург, 2024 г.

# Цель

Использовать средство синтаксического анализа по выбору, реализовать модуль для разбора текста в соответствии с языком по варианту. Реализовать построение по исходному файлу с текстом синтаксического дерева с узлами, соответствующими элементам синтаксической модели языка. Вывести полученное дерево в файл в формате, поддерживающем просмотр графического представления.

# Задачи

1. Изучить выбранное средство синтаксического анализа.
2. Изучить синтаксис разбираемого по варианту языка и записать спецификацию для средства синтаксического анализа, включающую следующие конструкции:
3. Подпрограммы со списком аргументов и возвращаемым значением
4. Операции контроля потока управления – простые ветвления if-else и циклы или аналоги
5. Целочисленные, строковые и односимвольные литералы
6. Выражения численной, битовой и логической арифметики
7. Выражения над одномерными массивами
8. Выражения вызова функции
9. Реализовать модуль, использующий средство синтаксического анализа для разбора языка по варианту
10. Реализовать тестовую программу для демонстрации работоспособности созданного модуля

# Описание работы

1. Я выбрал в качестве средства синтаксического анализа ANTRL3. Для начала я написал правила грамматики своего языка, используя ANTLR3. Файл, содержащий данную спецификацию, передаётся инструменту ANTLR3, после запуска которого образуются .c и .h файлы, которые используются в дальнейшем.
2. Разработка модуля ast, который на вход принимает поток ввода и на выходе возвращает абстрактное синтаксическое дерево. Также ast модуль предоставляет функционал для создания DOT файла.
3. Тестовое приложение на входе получает наименования файлов: с исходным кодом языка и файл для демонстрации абстрактного синтаксического дерева. Обращается к ast модулю для получения абстрактного синтаксического дерева.

# Аспекты реализации

В качестве дополнительной обработки пришлось перестраивать синтаксическое дерево разбора в местах вызова функций и доступа к массиву.

Суть в том, что для их успешного парсинга в ANTLR3 возможно только право-рекурсивное дерево (реализация), а это не тот результат, который необходим. Поэтому я реализовал дополнительный функционал, который переворачивает дерево (вызовы функций и доступ к массиву) с право-рекурсивного вида к лево- рекурсивнуму.

# Результаты

1. Получение исходных файлов в соответствии с спецификацией языка



1. Исходный код, который рассматривается:

// check function signature

def func\_sign(a of int, b of string, c of long array[10], d of bool array[5]array[5]array[5]) of byte

x = 2;

end

// check if statements

def if\_stat()

if (a > 3) then

b = "ehy";

if c < 5 then

d = 0b1100;

else

d = 0xFFFF;

end

// check while statements

def while\_stat()

while x > 2

y = y + 1;

end

end

// check repeat statements

def repeat\_stat()

{

x = 5 \* x;

if x > 50 then break;

} until x < 100;

y = 2 + y;

until y < 50;

i = 0;

{

j = 0;

{

j = j + 1;

} while j < i;

} while i < 10;

end

// check expressions

def foo\_expr()

a = (b + c) \* d;

a = b + c \* d;

a = b \* c + d;

a = b > c && d < e;

a = -b;

a = !c;

a = b && c = d;

a = x(arg);

a = x(2)("hello", 3 + 3);

y = a[1];

y = a[1][1];

c = x[1] + a[2];

a = x[1, 3..5];

z = x()[0];

w = b[0][0]()()[0]();

end

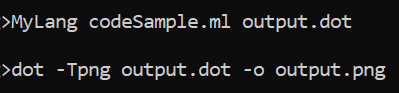
def check\_logic\_and\_bit()

x = a | ~b & c ^ d;

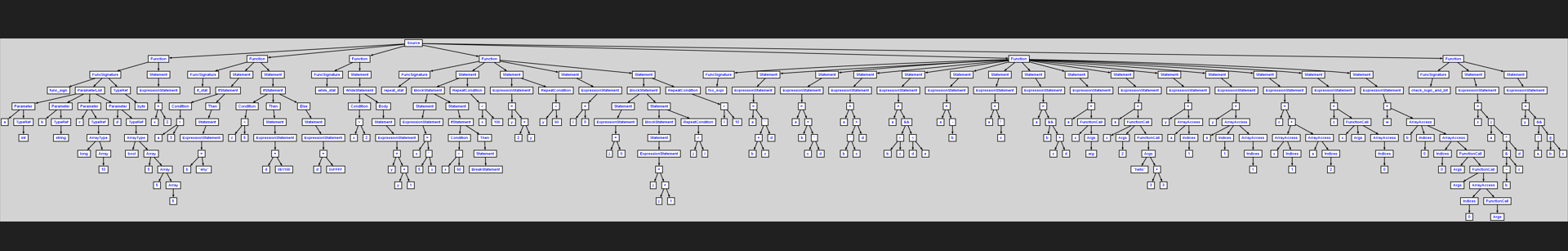
y = !a && b || c;

end

Запуск тестового модуля, который принимает файл с исходным кодом. Кодирование получившегося выходного файла в png формат.



1. Абстрактное синтаксическое дерево, которое мы теперь можем посмотреть (https://github.com/chetter14/my-language/blob/master/output.png):



# Выводы

В результате лабораторной работы я изучил средство синтаксического анализа ANTLR3. С его помощью написал спецификацию языка. Разработал модуль для получения абстрактного синтаксического дерева исходного кода в файле, который получаем на входе. На выходе имеется возможность просмотреть абстрактное синтаксическое дерево, которое было получено. Также можно увидеть, что абстрактное синтаксическое дерево обрабатывает все конструкции, которые требовались заданием, и отображает их в иерархическом порядке.