ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

ИМЕНИ Н. Э. БАУМАНА

(МГТУ ИМ. Н. Э. БАУМАНА)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»*

*КАФЕДРА «ТЕОРЕТИЧЕСКАЯ ИНФОРМАТИКА*

*И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ»*

ОТЧЕТ ПО ПРЕДДИПЛОМНОЙ ПРАКТИКЕ

***«СУПЕРКОМПИЛЯЦИЯ LL(1)-ГРАММАТИК»***

Студент группы ИУ9-72 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ **Головань С. М.**

Руководитель дипломного проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ **Коновалов А. В.**

Москва, 2018

**СОДЕРЖАНИЕ**

[Введение 3](#_Toc507451576)

[1. Разработка интерпретатора Python. 4](#_Toc507451577)

[1.1. Общие требования к интерпретатору 4](#_Toc507451578)

[1.2. Реализация интерпретатора 5](#_Toc507451579)

[2. Разработка UI 9](#_Toc507451580)

[2.1. Общие требования к пользовательскому интерфейсу 9](#_Toc507451581)

[2.2. Реализация пользовательского интерфейса 10](#_Toc507451582)

[3. Разработка генератора заданий 16](#_Toc507451583)

[4. Тестирование 18](#_Toc507451584)

[Заключение 19](#_Toc507451585)

[Список литературы 20](#_Toc507451586)

# ВВЕДЕНИЕ

Каждый день огромное количество людей, работающих с вычеслительными средствами явно или неявно сталкиваются с компиляторами и интерпретаторами – при просмотре web-страницы в браузере, запуске эмуляторов, приложений, при написании ПО и т.д.

Именно поэтому тестирование компиляторов занимает важное место при их разработке. Ожидается, что в процессе компиляции логика программы не будет искажена, а компилятор выдаст корректный результат при любых допустимых входных данных, возможно, ошибочных с точки зрения входного языка.

В данной работе рассматривается метод генерации тестов для синтаксического анализатора, основанный на идее суперкомпиляции, предложенной советским и американским ученым В. Ф. Турчиным еще в 1986 году в статье The concept of a supercompiler.

В частности, в качестве входного языка часто рассматриваются LL(1)-грамматики, преимуществом которых является возможность построения таблицы предсказывающего анализа для выполнения разбора входной цепочки распознавателем – автоматом с магазинной памятью.

Целью данной работы является практическое изучение и применение методов суперкомпиляции (вложения, обобщения) для построения набора тестовых цепочек парсера LL(1)-грамматик. Программное средство, реализующее данные методы, называется суперкомпилятором предметной области (domain specific supercompiler).

# ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ.

## LL(1)-грамматика

Известно, что LL(1)-грамматики, являясь подмножеством КС-грамматик, могут быть распознаны автоматом с магазинной памятью – предсказывающим анализатором, работа которого основывается на построении таблицы предсказывающего анализа.

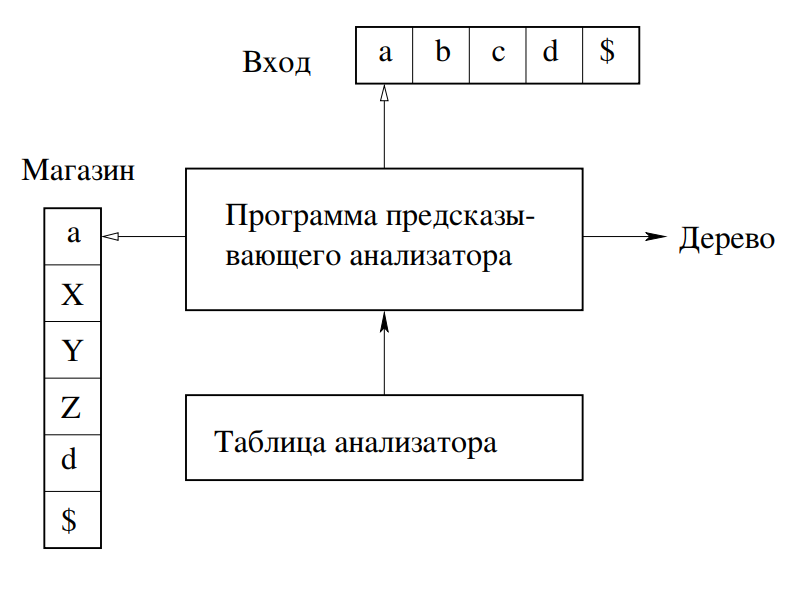


Рисунок 1. Структура предсказывающего анализатора

**Контекстно свободная грамматика** – кортеж , где:

– множество нетерминальных символов

– множество терминальных символов

– набор правил вывода: , *где* ,

– стартовое правило вывода (аксиома)

Класс LL(1)-грамматик, а также таблицу предсказывающего анализа удобно определять на основе множеств FIRST и FOLLOW, определяемых для любой КС-грамматики:

- множество терминалов, с которых начинаются цепочки, выводимые из .

- множество терминалов таких, что существует вывод вида**:**

.

Грамматика является LL(1), если для раскрытия любого из её правил достаточно знать только текущий символ входного потока.

Иными словами, **-грамматика**, если для любого правила

выполняется:

2. *Если то*

Обычно, грамматика задается в общей форме, в которой возможно наличие левой рекурсии, недостижимых символов, отсутствие левой факторизации и т.д. Таким образом, необходимо предварительно преобразовать исходную грамматику к LL(1)-виду.

## Суперкомпиляция

Суперкомпиляция –

## Понятие тестирования

# РАЗРАБОТКА

# ТЕСТИРОВАНИЕ

# ЗАКЛЮЧЕНИЕ

# СПИСОК ЛИТЕРАТУРЫ