МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное государственное учреждение высшего образования

Санкт-петербургский национальный исследовательский институт информационных технологий, механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Лабораторная работа №2**

**По дисциплине «Методы трансляции»**

**Ручное построение нисходящих синтаксических анализаторов**

**Вариант 2**

**Выполнил студент группы №M33091:**

*Золотарева Анастасия Кирилловна*

**Проверил:**

*Станкевич Андрей Сергеевич*

**САНКТ-ПЕТЕРБУРГ**

**2021**

Введение

Цель работы

Цель данной лабораторной работы - научиться разрабатывать грамматики для заданных неформально языков с учетом семантического смысла и приоритета операторов, разрабатывать лексические анализаторы, разрабатывать вручную нисходящие синтаксические анализаторы, разрабатывать подробные тестовые наборы для анализаторов.

Форма отчетности: программа и текстовый отчет. Программа может быть написана на любом языке программирования общего назначения (рекомендуются Си++ и Java).

Задание варианта

Регулярные выражения с операциями конкатенации (простая последовательная запись строк), выбора (вертикальная черта), замыкания Клини. Приоритет операций стандартный. Скобки могут использоваться для изменения приоритета.

Для обозначения базовых языков используются маленькие буквы латинского алфавита. Используйте один терминал для всех символов.

Пример: **((abc\*b|a)\*ab(aa|b\*)b)\***.

Разработка грамматики

Разработайте контекстно-свободную грамматику для языка, описанного в условии вашего варианта. Сначала разработайте грамматику, исходя из структуры языка, чтобы она максимально близко соответствовала интуитивным представлениям о построении слов из языка. Затем, при необходимости, устраните левую рекурсию и/или правое ветвление.

В отчете приведите исходную и преобразованную грамматику, опишите смысл всех нетерминалов.

Грамматика

// из описания правил

S->SS

S->S|S

S->S\*S

S-> (S)

S->eps || char

Грамматику следовало бы исправить, задействовать больше нетерминалов:

S -> V S'

S' -> | V S'

S' -> eps

V -> K V'

V' -> K V'

V' -> \* V'

V' -> eps

K -> C K'

K' -> eps

C -> ( S )

C -> char

|  |  |
| --- | --- |
| **Нетерминал** | **Описание** |
| S | Правильное регулярное выражение |
| S’ | Продолжение правильного рег. выражения |
| V | Вариативная последовательность |
| V’ | Продолжение вариативной последовательности |
| K | Строка с возможным замыканием |
| K’ | Продолжение строки с возможным замыканием |
| C | Строка из букв латинского алфавита |

Построение лексического анализатора

Лексический анализатор должен получать на вход строку и выдавать последовательность терминалов (токенов). Пробелы и переводы строк должны игнорироваться.

Токены

|  |  |
| --- | --- |
| **Терминал** | **Токен** |
| ( | LPAREN |
| ) | RPAREN |
| | | VARIANT |
| \* | CONC |
| [a-z] | CHARACTER |
| $ | END |

**Код смотреть в файле lexicalAnalyser.cpp.**

Построение синтаксического анализатора

Постройте множества FIRST и FOLLOW для нетерминалов вашей грамматики. Затем напишите синтаксический анализатор с использованием рекурсивного спуска.

FIRST & FOLLOW

|  |  |  |
| --- | --- | --- |
| **Нетерминал** | **FIRST** | **FOLLOW** |
| S | (, char | ), $ |
| S’ | |, eps | ), $ |
| V | (, char, | ), |, $ |
| V’ | (, \*, char, eps | ), |, $ |
| K | (, char | ), \*, |, $ |
| K’ | eps | ), \*, |, (, char, ), $ |
| C | (, char | ), \*, |, (, char, ), $ |

Analyser

**Смотреть файл lexSyn.cpp.**

Подготовка набора тестов

Подготовьте набор исчерпывающих тестов для вашей программы. По возможности добейтесь, чтобы все возможные ошибки в грамматике или в лексическом/синтаксическом анализаторе ими покрывались. Про каждый тест дайте короткое описание.

Тесты (см. файлы отмеченные, как <…>Test):

Тесты лексического анализатора:

|  |  |
| --- | --- |
| **Тест** | **Описание** |
| a | Тесты на учет простых правильных последовательностей нетерминалов строки. |
| ab |
| abc |
| a|b |
| a\* |
| a\*|b |
| (a)\*|b |
| A | Тесты на учет простых неправильных последовательностей нетерминалов строки. |
| aA |
| 10 |
| a? |
| (a )\* |b | Тесты на учет знаков разделения, табуляции и переноса стоки |
| (a\n)\*\t|b\n |
| ((abc\*b|a)\*ab(aa|b\*)b)\* | Простой случайный тест |

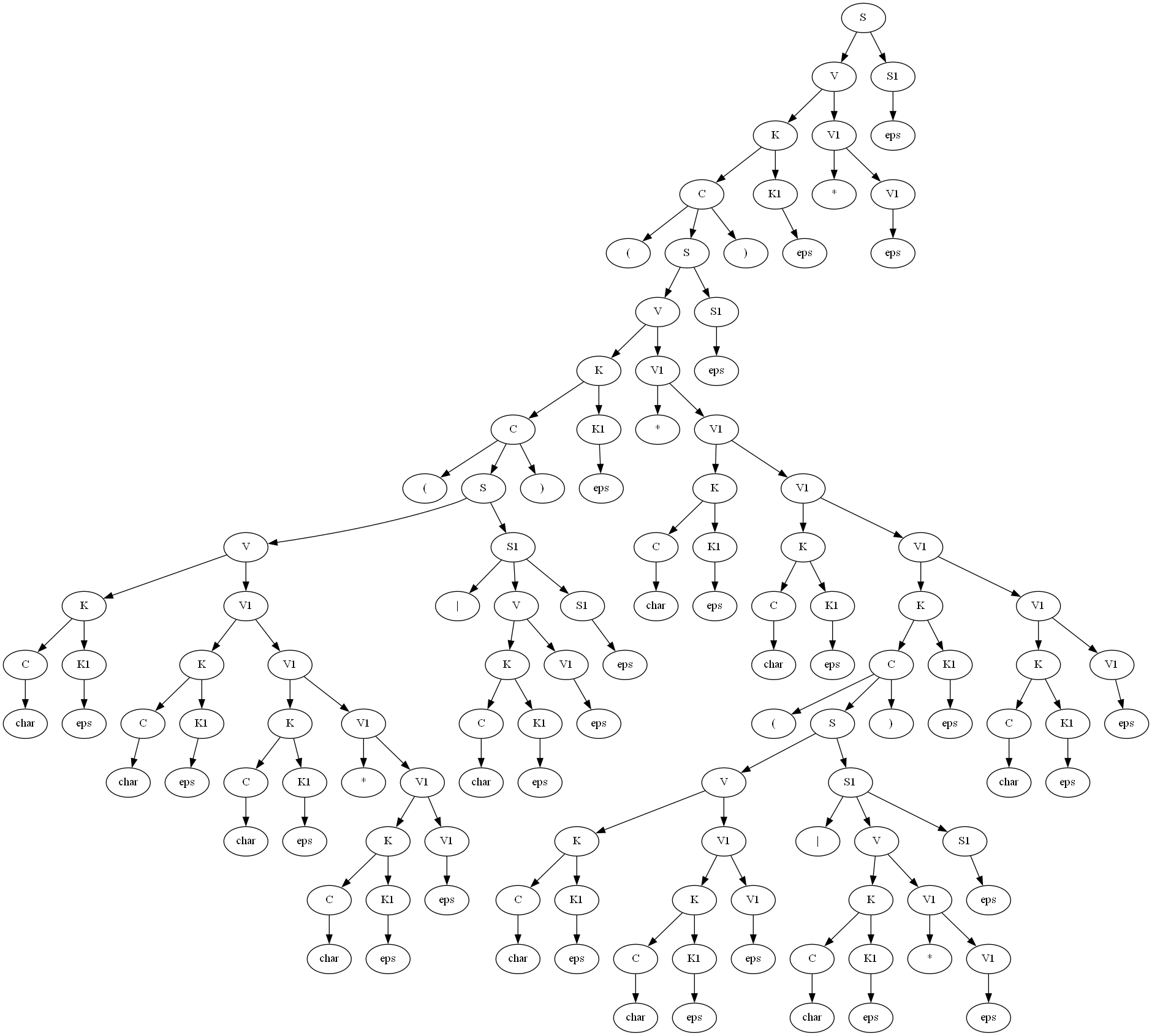
Тестирование синтаксического анализатора:

|  |  |
| --- | --- |
| **Тест** | **Описание** |
| a | Тест на правило C -> char |
| a\* | Тест на правила V -> K V’, V’ -> \* V’ |
| a|b | Тест на правило S|S |
| abc|b | SS|S |
| abc\*b | SS\*S |
| abc\*|b | SS\*|S |
| (a) | (S) |
| ((a)) | S-> (S) -> ((S)) |
| (abc) | (S) -> (SS) |
| (abc)\* | (S) -> (S)\* |
| ((abc\*b|a)\*ab(aa|b\*)b)\* | Случайный тест |
| \* | Нет символьных терминалов перед \* |
| a| | После | не следует правильная последовательность |
| ()|b | Пустые скобки |
| |b | Перед | нет правильного регулярного выражения |
| ()| | Пустые скобки и после | нет правильного регулярного выражения |
| abc\*| | После | не следует правильная последовательность |
| \*| | Перед \* нет правильной скобочной последовательности – отсутствие символов |
| | | С | строка не может начинаться |
| (a)) | Лишняя закрывающая скобка |
| )( | Только скобки (неправильно расставлены |
| ((a) | Нет закрывающей скобки |
| ( | После скобки нет правильной последовательности, скобка не закрыта |

Визуализация дерева разбора

Для изучения результата разработайте систему визуализации дерева разбора. Рекомендуется использовать систему GraphViz (<https://graphviz.org/>).

Визуализации для теста ((abc\*b|a)\*ab(aa|b\*)b)\*:



**Все визуализации можно посмотреть в папке imgs.**