|  |  |
| --- | --- |
| Институт (факультет) | Институт Информационных Технологий |
| Кафедра | Математического и Программного Обеспечения ЭВМ |

КУРСОВАЯ РАБОТА

|  |
| --- |
| по дисциплине Теория автоматов и формальных языков |

|  |  |
| --- | --- |
| на тему | Конструирование моделей лексического и синтаксического анализа |

|  |
| --- |
| Выполнил студент группы |
| 1ПИб-01-21оп |
| направление подготовки (специальности) |
| 09.03.04, Программная инженерия |
| *шифр, наименование* |
| Мороз Никита Романович |
| *фамилия, имя, отчество* |

|  |
| --- |
| Руководитель |
| **Ганичева Оксана Георгиевна** |
| *фамилия, имя, отчество* |
| доцент |
| *должность* |

|  |
| --- |
| Дата представления работы |
| «\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_ г. |
|  |
| Заключение о допуске к защите |
|  |
|  |
|  |
|  |
| Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| количество баллов |
| Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

# Аннотация

Данный документ содержит описание курсовой работы по дисциплине «Теория автоматов и формальных языков» на тему «Конструирование моделей лексического и синтаксического анализа».

Документ содержит следующие приложения:

* Приложение 1: Техническое задание.
* Приложение 2: Текст программы.
* Лист 1: Граф конечного автомата.

Оглавление

[Аннотация 2](#_Toc470480731)

[Введение 4](#_Toc470480732)

[1. Лексический анализ 5](#_Toc470480733)

[1.1 Введение 5](#_Toc470480734)

[1.2 Конечный автомат 6](#_Toc470480735)

[1.3 Регулярная грамматика 12](#_Toc470480736)

[Регулярная грамматика может быть задана набором правил как левая или правая регулярная грамматика. 12](#_Toc470480737)

[1.4 Программная реализация лексического анализатора 18](#_Toc470480738)

[1.4.1 Изучение и описание предметной области 18](#_Toc470480739)

[1.4.2 Постановка задачи 18](#_Toc470480740)

[1.4.3 Логическое проектирование 18](#_Toc470480741)

[1.4.5 Проектирование интерфейса 18](#_Toc470480742)

[1.4.6 Тестирование 19](#_Toc470480743)

[2. Синтаксический анализ 21](#_Toc470480744)

[2.1 Пример инструкции языка программирования 21](#_Toc470480745)

[2.2 Контекстно-свободная грамматика 21](#_Toc470480746)

[2.3 Форма Бэкуса-Наура 23](#_Toc470480747)

[2.4 Синтаксические диаграммы Вирта 24](#_Toc470480748)

# Введение

Целью курсовой работы является разработка лексического и синтаксического анализаторов для заданных синтаксических конструкций языка С++.

* Компилятор — программа или техническое средство, выполняющее компиляцию.
* Компиляция — трансляция программы, составленной на исходном языке высокого уровня, в эквивалентную программу на низкоуровневом языке, близком машинному коду
* Формальная грамматика — способ описания формального языка, то есть выделения некоторого подмножества из множества всех слов некоторого конечного алфавита.
* Теория автоматов — это раздел теории управляющих систем, изучающий математические модели преобразователей дискретной информации, называемые автоматами.
* Конечный автомат — абстрактный автомат, число возможных внутренних состояний которого конечно. Математическим аппаратом для описания языка программирования является теория формальных языков и грамматик.

На начальных этапах компиляции необходимо провести лексический и синтаксический анализы:

* Лексический анализ — процесс аналитического разбора входной последовательности символов на последовательность лексем.
* Синтаксический анализ — процесс сопоставления линейной последовательности лексем формального языка с его формальной грамматикой и вывод дерева разбора.

# 1. Лексический анализ

1.1 Введение

* Лексема – это структурная единица языка, которая состоит из элементарных символов языка и не содержит в своем составе других структурных единиц языка.

Задача лексического анализа состоит в чтении символов и выдачи последовательности лексем, которые в дальнейшем обрабатывается синтаксическим анализатором. Входными данными для лексического анализатора является текст программы, а на выходе генерируется в виде таблиц последовательностей дескрипторов хранящих в себе лексемы.

* Дескриптор — лексическая единица выраженная информативным словом или кодом

Основные функции лексического анализа:

* Удаление пробелов, отбивок, переносов строки, символов табуляции из исходного текста программы;
* Удаление комментариев из текста исходной программы;
* выделение лексем типов: констант (символьных и числовых), знаков операций, идентификаторов, ключевых слов и разделителей.

Для распознавания регулярного языка необходимо реализовать конечный автомат.

1.2 Конечный автомат

Конечный автомат — модель, содержащая конечное число состояний определенной модели какого-либо явления. Используется для представления и управления потоком выполнения команд.

Конечный автомат может быть представлен как:

A = (V, Q, R, q0, I), где:

* V – входной алфавит;
* Q – конечное множество состояний;
* R – множество заключительных состояний (R ⊂ Q);
* q0 – начальное состояние (q0 ⊂ Q);
* I – набор команд;

Конечный автомат:

(Граф на Листе 1)

A = (V, Q, R, q0, I)

V = {a..z**,** A..Z, 0..9, **+**, -, **\***, **/**, **=**, **:**, **;**, **,**, **\_**, **<**, **>**, [ , ],  **(**, **)**, {, }, |, #, &, !}

Q = {q0, q1, .., q117}

R = {q1, q2, q4, q6, q12, q16, q19, q25, q29, q33, q39, q43, q44, q49, q53, q55, q58, q61, q64, q67, q70, q73, q74, q75, q76, q77, q78, q79, q80 q81, q82, q83, q84, q85, q86, q87, q88, q89, q90, q91, q92, q93, q94, q102, q103, q104, q105, q106, q107, q108, q109, q110, q111, q112, q113, q114, q115}

I:

(q0, i) → q3

(q0, d) → q7

(q0, e) → q13

(q0, s) → q20

(q0, l) → q26

(q0, v) → q30

(q0, r) → q34

(q0, t) → q40

(q0, f) → q45

(q0, c) → q56

(q0, >) → q74

(q0, <) → q77

(q0, +) → q80

(q0, -) → q83

(q0, \*) → q86

(q0, /) → q88

(q0, =) → q91

(q0, !) → q93

(q0, #) → q95

(q0, &) → q103

(q0, |) → q105

(q0, ;) → q107

(q0, :) → q108

(q0, ,) → q109

(q0, {) → q110

(q0, }) → q111

(q0, [) → q112

(q0, ]) → q113

(q0, () → q114

(q0, )) → q115

(q0, a) → q117

(q0, b) → q117

(q0, g) → q117

(q0, h) → q117

(q0, j) → q117

(q0, k) → q117

(q0, m) → q117

(q0, n) → q117

(q0, o) → q117

(q0, p) → q117

(q0, q) → q117

(q0, u) → q117

(q0, w) → q117

(q0, x) → q117

(q0, y) → q117

(q0, z) → q117

(q0, A) → q117

...

(q0, Z) → q117

(q0, \_) → q117

(q0, 0) → q116

…

(q0, 9) → q116

(q3, f) → q4

(q3, n) → q5

(q5, t) → q6

(q7, o) → q8

(q8, u) → q9

(q9, b) → q10

(q10, l) → q11

(q11, e) → q12

(q13, l) → q14

(q14, s) → q15

(q15, e) → q16

(q13, n) → q17

(q17, d) → q18

(q18, l) → q19

(q20, t) → q21

(q21, r) → q22

(q22, i) → q23

(q23, n) → q24

(q24, g) → q25

(q26, o) → q27

(q27, n) → q28

(q28, g) → q29

(q30, o) → q31

(q31, i) → q32

(q32, d) → q33

(q34, e) → q35

(q35, t) → q36

(q36, u) → q37

(q37, r) → q38

(q38, n) → q39

(q40, r) → q41

(q41, u) → q42

(q42, e) → q43

(q41, y) → q44

(q45, l) → q46

(q46, o) → q47

(q47, a) → q48

(q48, t) → q49

(q45, a) → q50

(q50, l) → q51

(q51, s) → q52

(q52, e) → q53

(q45, o) → q54

(q54, r) → q55

(q56, i) → q56

(q57, n) → q57

(q56, o) → q59

(q59, u) → q60

(q60, t) → q61

(q59, n) → q62

(q62, s) → q63

(q63, t) → q64

(q56, h) → q65

(q65, a) → q66

(q66, r) → q67

(q56, a) → q68

(q68, s) → q69

(q69, e) → q70

(q68, t) → q71

(q71, c) → q72

(q72, h) → q73

(q74, >) → q75

(q74, =) → q76

(q77, <) → q78

(q77, =) → q79

(q80, +) → q81

(q80, =) → q82

(q83, -) → q84

(q83, =) → q85

(q86, /) → q87

(q88, /) → q89

(q88, \*) → q90

(q91, =) → q92

(q93, =) → q94

(q95, i) → q96

(q96, n) → q97

(q97, c) → q98

(q98, l) → q99

(q99, u) → q100

(q100, d) → q101

(q101, e) → q102

(q3, a) → q117

…

(q3, e) → q117

…

(q3, m) → q117

(q3, o) → q117

…

(q3, g) → q117

…

(q3, z) → q117

(q3, A) → q117

…

(q3, Z) → q117

(q3, 0) → q117

…

(q3, 9) → q117

(q3, \_) → q117

(q4, a) → q117

…

(q4, z) → q117

(q4, A) → q117

…

(q4, Z) → q117

(q4, 0) → q117

…

(q4, 9) → q117

(q4, \_) → q117

(q5, a) → q117

…

(q5, s) → q117

(q5, u) → q117

…

(q5, z) → q117

(q5, A) → q117

…

(q5, Z) → q117

(q5, 0) → q117

…

(q5, 9) → q117

(q5, \_) → q117

(q6, a) → q117

…

(q6, z) → q117

(q6, A) → q117

…

(q6, Z) → q117

(q6, 0) → q117

…

(q6, 9) → q117

(q6, \_) → q117

(q7, a) → q117

…

(q7, n) → q117

(q7, p) → q117

…

(q7, z) → q117

(q7, A) → q117

…

(q7, Z) → q117

(q7, 0) → q117

…

(q7, 9) → q117

(q7, \_) → q117

(q8, a) → q117

…

(q8, t) → q117

(q8, v) → q117

…

(q8, z) → q117

(q8, A) → q117

…

(q8, Z) → q117

(q8, 0) → q117

…

(q8, 9) → q117

(q8, \_) → q117

(q9, a) → q117

(q9, c) → q117

(q9, g) → q117

…

(q9, z) → q117

(q9, A) → q117

…

(q9, Z) → q117

(q9, 0) → q117

…

(q9, 9) → q117

(q9, \_) → q117

(q10, a) → q117

…

(q10, k) → q117

(q10, m) → q117

…

(q10, z) → q117

(q10, A) → q117

…

(q10, Z) → q117

(q10, 0) → q117

…

(q10, 9) → q117

(q10, \_) → q117

(q11, a) → q117

…

(q11, d) → q117

(q11, f) → q117

…

(q11, z) → q117

(q11, A) → q117

…

(q11, Z) → q117

(q11, 0) → q117

…

(q11, 9) → q117

(q11, \_) → q117

(q12, a) → q117

…

(q12, z) → q117

(q12, A) → q117

…

(q12, Z) → q117

(q12, 0) → q117

…

(q12, 9) → q117

(q12, \_) → q117

(q13, a) → q117

…

(q13, k) → q117

(q13, m) → q117

(q13, o) → q117

…

(q13, z) → q117

(q13, A) → q117

…

(q13, Z) → q117

(q13, 0) → q117

…

(q13, 9) → q117

(q13, \_) → q117

(q14, a) → q117

…

(q14, r) → q117

(q14, t) → q117

…

(q14, z) → q117

(q14, A) → q117

…

(q14, Z) → q117

(q14, 0) → q117

…

(q14, 9) → q117

(q14, \_) → q117

(q15, a) → q117

…

(q15, d) → q117

(q15, f) → q117

…

(q15, z) → q117

(q15, A) → q117

…

(q15, Z) → q117

(q15, 0) → q117

…

(q15, 9) → q117

(q15, \_) → q117

(q16, a) → q117

…

(q16, z) → q117

(q16, A) → q117

…

(q16, Z) → q117

(q16, 0) → q117

…

(q16, 9) → q117

(q16, \_) → q117

(q17, a) → q117

…

(q17, c) → q117

(q17, e) → q117

…

(q17, z) → q117

(q17, A) → q117

…

(q17, Z) → q117

(q17, 0) → q117

…

(q17, 9) → q117

(q17, \_) → q117

(q18, a) → q117

…

(q18, k) → q117

(q18, m) → q117

…

(q18, z) → q117

(q18, A) → q117

…

(q18, Z) → q117

(q18, 0) → q117

…

(q18, 9) → q117

(q18, \_) → q117

(q19, a) → q117

…

(q19, z) → q117

(q19, A) → q117

…

(q19, Z) → q117

(q19, 0) → q117

…

(q19, 9) → q117

(q19, \_) → q117

(q20, a) → q117

…

(q20, s) → q117

(q20, u) → q117

…

(q20, z) → q117

(q20, A) → q117

…

(q20, Z) → q117

(q20, 0) → q117

…

(q20, 9) → q117

(q20, \_) → q117

(q21, a) → q117

…

(q21, q) → q117

(q21, s) → q117

…

(q21, z) → q117

(q21, A) → q117

…

(q21, Z) → q117

(q21, 0) → q117

…

(q21, 9) → q117

(q21, \_) → q117

(q22, a) → q117

…

(q22, h) → q117

(q22, j) → q117

…

(q22, z) → q117

(q22, A) → q117

…

(q22, Z) → q117

(q22, 0) → q117

…

(q22, 9) → q117

(q22, \_) → q117

(q23, a) → q117

…

(q23, m) → q117

(q23, o) → q117

…

(q23, z) → q117

(q23, A) → q117

…

(q23, Z) → q117

(q23, 0) → q117

…

(q23, 9) → q117

(q23, \_) → q117

(q24, a) → q117

…

(q24, f) → q117

(q24, h) → q117

…

(q24, z) → q117

(q24, A) → q117

…

(q24, Z) → q117

(q24, 0) → q117

…

(q24, 9) → q117

(q24, \_) → q117

(q25, a) → q117

…

(q25, z) → q117

(q25, A) → q117

…

(q25, Z) → q117

(q25, 0) → q117

…

(q25, 9) → q117

(q25, \_) → q117

(q26, a) → q117

…

(q26, n) → q117

(q26, p) → q117

…

(q26, z) → q117

(q26, A) → q117

…

(q26, Z) → q117

(q26, 0) → q117

…

(q26, 9) → q117

(q26, \_) → q117

(q27, a) → q117

…

(q27, m) → q117

(q27, o) → q117

…

(q27, z) → q117

(q27, A) → q117

…

(q27, Z) → q117

(q27, 0) → q117

…

(q27, 9) → q117

(q27, \_) → q117

(q28, a) → q117

…

(q28, f) → q117

(q28, h) → q117

…

(q28, z) → q117

(q28, A) → q117

…

(q28, Z) → q117

(q28, 0) → q117

…

(q28, 9) → q117

(q28, \_) → q117

(q29, a) → q117

…

(q29, z) → q117

(q29, A) → q117

…

(q29, Z) → q117

(q29, 0) → q117

…

(q29, 9) → q117

(q29, \_) → q117

(q30, a) → q117

…

(q30, n) → q117

(q30, p) → q117

…

(q30, z) → q117

(q30, A) → q117

…

(q30, Z) → q117

(q30, 0) → q117

…

(q30, 9) → q117

(q30, \_) → q117

(q31, a) → q117

…

(q31, h) → q117

(q31, j) → q117

…

(q31, z) → q117

(q31, A) → q117

…

(q31, Z) → q117

(q31, 0) → q117

…

(q31, 9) → q117

(q31, \_) → q117

(q32, a) → q117

…

(q32, c) → q117

(q32, e) → q117

…

(q32, z) → q117

(q32, A) → q117

…

(q32, Z) → q117

(q32, 0) → q117

…

(q32, 9) → q117

(q32, \_) → q117

(q33, a) → q117

…

(q33, z) → q117

(q33, A) → q117

…

(q33, Z) → q117

(q33, 0) → q117

…

(q33, 9) → q117

(q33, \_) → q117

(q34, a) → q117

…

(q34, d) → q117

(q34, f) → q117

…

(q34, z) → q117

(q34, A) → q117

…

(q34, Z) → q117

(q34, 0) → q117

…

(q34, 9) → q117

(q34, \_) → q117

(q35, a) → q117

…

(q35, s) → q117

(q35, u) → q117

…

(q35, z) → q117

(q35, A) → q117

…

(q35, Z) → q117

(q35, 0) → q117

…

(q35, 9) → q117

(q35, \_) → q117

(q36, a) → q117

…

(q36, t) → q117

(q36, v) → q117

…

(q36, z) → q117

(q36, A) → q117

…

(q36, Z) → q117

(q36, 0) → q117

…

(q36, 9) → q117

(q36, \_) → q117

(q37, a) → q117

…

(q37, q) → q117

(q37, s) → q117

…

(q37, z) → q117

(q37, A) → q117

…

(q37, Z) → q117

(q37, 0) → q117

…

(q37, 9) → q117

(q37, \_) → q117

(q38, a) → q117

…

(q38, m) → q117

(q38, o) → q117

…

(q38, z) → q117

(q38, A) → q117

…

(q38, Z) → q117

(q38, 0) → q117

…

(q38, 9) → q117

(q38, \_) → q117

(q39, a) → q117

…

(q39, z) → q117

(q39, A) → q117

…

(q39, Z) → q117

(q39, 0) → q117

…

(q39, 9) → q117

(q39, \_) → q117

(q40, a) → q117

…

(q40, q) → q117

(q40, s) → q117

…

(q40, z) → q117

(q40, A) → q117

…

(q40, Z) → q117

(q40, 0) → q117

…

(q40, 9) → q117

(q40, \_) → q117

(q41, a) → q117

…

(q41, t) → q117

(q41, v) → q117

…

(q41, x) → q117

(q41, z) → q117

(q41, A) → q117

…

(q41, Z) → q117

(q41, 0) → q117

…

(q41, 9) → q117

(q41, \_) → q117

(q42, a) → q117

…

(q42, d) → q117

(q42, f) → q117

…

(q42, z) → q117

(q42, A) → q117

…

(q42, Z) → q117

(q42, 0) → q117

…

(q42, 9) → q117

(q42, \_) → q117

(q43, a) → q117

…

(q43, z) → q117

(q43, A) → q117

…

(q43, Z) → q117

(q43, 0) → q117

…

(q43, 9) → q117

(q43, \_) → q117

(q44, a) → q117

…

(q44, z) → q117

(q44, A) →q117

…

(q44, Z) → q117

(q44, 0) → q117

…

(q44, 9) → q117

(q44, \_) → q117

(q45, b) → q117

…

(q45, k) → q117

(q45, m) → q117

(q45, n) → q117

…

(q45, p) → q117

(q45, A) → q117

…

(q45, Z) → q117

(q45, 0) → q117

…

(q45, 9) → q117

(q45, \_) → q117

(q46, a) → q117

…

(q46, n) → q117

(q46, p) → q117

…

(q46, z) → q117

(q46, A) → q117

…

(q46, Z) → q117

(q46, 0) → q117

…

(q46, 9) → q117

(q46, \_) → q117

(q47, b) → q117

…

(q47, z) → q117

(q47, A) → q117

…

(q47, Z) → q117

(q47, 0) → q117

…

(q47, 9) → q117

(q47, \_) → q117

(q48, a) → q117

…

(q48, s) → q117

(q48, u) → q117

…

(q48, z) → q117

(q48, A) → q117

…

(q48, Z) → q117

(q48, 0) → q117

…

(q48, 9) → q117

(q48, \_) → q117

(q49, a) → q117

…

(q49, z) → q117

(q49, A) → q117

…

(q49, Z) → q117

(q49, 0) → q117

…

(q49, 9) → q117

(q49, \_) → q117

(q50, a) → q117

…

(q50, k) → q117

(q50, m) → q117

…

(q50, z) → q117

(q50, A) → q117

…

(q50, Z) → q117

(q50, 0) → q117

…

(q50, 9) → q117

(q50, \_) → q117

(q51, a) → q117

…

(q51, r) → q117

(q51, t) → q117

…

(q51, z) → q117

(q51, A) → q117

…

(q51, Z) → q117

(q51, 0) → q117

…

(q51, 9) → q117

(q51, \_) → q117

(q52, a) → q117

…

(q52, d) → q117

(q52, f) → q117

…

(q52, z) → q117

(q52, A) → q117

…

(q52, Z) → q117

(q52, 0) → q117

…

(q52, 9) → q117

(q52, \_) → q117

(q53, a) → q117

…

(q53, z) → q117

(q53, A) → q117

…

(q53, Z) → q117

(q53, 0) → q117

…

(q53, 9) → q117

(q53, \_) → q117

(q54, a) → q117

…

(q54, q) → q117

(q54, s) → q117

…

(q54, z) → q117

(q54, A) → q117

…

(q54, Z) → q117

(q54, 0) → q117

…

(q54, 9) → q117

(q54, \_) → q117

(q55, a) → q117

…

(q55, z) → q117

(q55, A) → q117

…

(q55, Z) → q117

(q55, 0) → q117

…

(q55, 9) → q117

(q55, \_) → q117

(q56, b) → q117

…

(q56, g) → q117

(q56, j) → q117

…

(q56, n) → q117

(q56, p) → q117

…

(q56, z) → q117

(q56, A) → q117

…

(q56, Z) → q117

(q56, 0) → q117

…

(q56, 9) → q117

(q56, \_) → q117

(q57, a) → q117

…

(q57, m) → q117

(q57, o) → q117

…

(q57, z) → q117

(q57, A) → q117

…

(q57, Z) → q117

(q57, 0) → q117

…

(q57, 9) → q117

(q57, \_) → q117

(q58, a) → q117

…

(q58, z) → q117

(q58, A) → q117

…

(q58, Z) → q117

(q58, 0) → q117

…

(q58, 9) → q117

(q58, \_) → q117

(q59, a) → q117

…

(q59, m) → q117

(q59, o) → q117

…

(q59, t) → q117

(q59, v) → q117

…

(q59, z) → q117

(q59, A) → q117

…

(q59, Z) → q117

(q59, 0) → q117

…

(q59, 9) → q117

(q59, \_) → q117

(q60, a) → q117

…

(q60, s) → q117

(q60, u) → q117

…

(q60, z) → q117

(q60, A) → q117

…

(q60, Z) → q117

(q60, 0) → q117

…

(q60, 9) → q117

(q60, \_) → q117

(q61, a) → q117

…

(q61, z) → q117

(q61, A) → q117

…

(q61, Z) → q2

(q61, 0) → q117

…

(q61, 9) → q117

(q61, \_) → q117

(q62, a) → q117

…

(q62, r) → q117

(q62, t) → q117

…

(q62, z) → q117

(q62, A) → q117

…

(q62, Z) → q117

(q62, 0) → q117

…

(q62, 9) → q117

(q62, \_) → q117

(q63, a) → q117

…

(q63, s) → q117

(q63, u) → q117

…

(q63, z) → q117

(q63, A) → q117

…

(q63, Z) → q117

(q63, 0) → q117

…

(q63, 9) → q117

(q63, \_) → q117

(q64, a) → q117

…

(q64, z) → q117

(q64, A) → q117

…

(q64, Z) → q117

(q64, 0) → q117

…

(q64, 9) → q117

(q64, \_) → q117

(q65, b) → q117

…

(q65, z) → q117

(q65, A) → q117

…

(q65, Z) → q117

(q65, 0) → q117

…

(q65, 9) → q117

(q65, \_) → q117

(q66, a) → q117

…

(q66, q) → q117

(q66, s) → q117

…

(q66, z) → q117

(q66, A) → q117

…

(q66, Z) → q117

(q66, 0) → q117

…

(q66, 9) → q117

(q66, \_) → q117

(q67, a) → q117

…

(q67, z) → q117

(q67, A) → q117

…

(q67, Z) → q117

(q67, 0) → q117

…

(q67, 9) → q117

(q67, \_) → q117

(q68, a) → q117

…

(q68, r) → q117

(q68, t) → q117

…

(q68, z) → q117

(q68, A) → q117

…

(q68, Z) → q117

(q68, 0) → q117

…

(q68, 9) → q117

(q68, \_) → q117

(q69, a) → q117

…

(q69, d) → q117

(q69, f) → q117

…

(q69, z) → q117

(q69, A) → q117

…

(q69, Z) → q117

(q69, 0) → q117

…

(q69, 9) → q117

(q69, \_) → q117

(q70, a) → q117

…

(q70, z) → q117

(q70, A) → q117

…

(q70, Z) → q117

(q70, 0) → q117

…

(q70, 9) → q117

(q70, \_) → q117

(q71, a) → q117

(q71, b) → q117

(q71, d) → q117

…

(q71, z) → q117

(q71, A) → q117

…

(q71, Z) → q117

(q71, 0) → q117

…

(q71, 9) → q117

(q71, \_) → q117

(q72, a) → q117

…

(q72, g) → q117

(q72, i) → q117

…

(q72, z) → q117

(q72, A) → q117

…

(q72, Z) → q117

(q72, 0) → q117

…

(q72, 9) → q117

(q72, \_) → q117

(q73, a) → q117

…

(q73, z) → q117

(q73, A) → q117

…

(q73, Z) → q117

(q73, 0) → q117

…

(q73, 9) → q117

(q73, \_) → q117

(q117, a) → q117

…

(q117, z) → q117

(q117, A) → q117

…

(q117, Z) → q117

(q117, 0) → q117

…

(q117, 9) → q117

(q117, \_) → q117

(q117, a) → q2

…

(q117, z) → q2

(q117, A) → q2

…

(q117, Z) → q2

(q117, 0) → q2

…

(q117, 9) → q2

(q117, \_) → q2

(q116, a) → q1

…

(q116, z) → q1

(q116, A) → q1

…

(q116, Z) → q1

(q116, 0) → q2

…

(q116, 9) → q2

(q116, \_) → q1

1.3 Регулярная грамматика

Регулярная грамматика может быть задана набором правил как левая или правая регулярная грамматика.

правая регулярная грамматика — все правила могут быть в одной из следующих форм:

A → a A → aB A → ε

левая регулярная грамматика - все правила могут быть в одной из следующих форм:

A → a A → Ba A → ε

Где A, B – нетерминальные символы; a – терминальный символ.

ε - пустая строка, (строка длины 0).

По конечному автомату была построена соответствующая ему регулярная грамматика:

G = (N, T, S, P)

N = {Q} = {Q0, Q1, , Q125}

T = {V} = {a..z, A..Z, 0..9, **+**, -, **\***, **/**, **<**, **>**, **=**, **:**, **;**, **,**, **\_**, **(**, **)**, **[**, ] ,{, }, &, |, !}

S = {q0}= {Q0}

P:

Q0 → aQ1

Q0 → bQ1

Q0 → cQ30

Q0 → dQ1

Q0 → eQ1

Q0 → fQ14

Q0 → gQ1

Q0 → hQ1

Q0 → iQ27

Q0 → gQ1

Q0 → “Q47

Q0 → {Q48

Q0 → ‘Q49

Q0 → kQ1

Q0 → lQ1

Q0 → mQ1

Q0 → nQ1

Q0 → oQ1

Q0 → pQ1

Q0 → qQ1

Q0 → rQ1

Q0 → sQ5

Q0 → tQ19

Q0 → uQ42

Q0 → vQ1

Q0 → wQ1

Q0 → xQ1

Q0 → yQ1

Q0 → zQ1

Q0 → /Q53

Q1 → vrQ1

Q1 → brdrQ2

Q53 → >Q61

Q53 → <Q62

Q53 → =Q64

Q48 → }Q1

Q26 → )Q0

Q5 → tQ6

Q6 → rQ7

Q7 → uQ8

Q8 → cQ9

Q9 → tQ10

Q14 → aQ14

Q14 → aQ6

Q15 → lQ11

Q16 → sQ8

Q17 → eQ18

Q18 → vrQ1

Q19 → rQ20

Q20 → uQ21

Q21 → eQ22

Q27 → nQ28

Q28 → tQ29

Q23 → oQ39

Q39 → oQ40

Q42 → nQ43

Q43 → iQ44

Q44 → oQ45

Q45 → nQ46

Q47 → “Qx

Q48 → }Qx

Q49 → ‘Qx

# 1.4 Программная реализация лексического анализатора

1.4.1 Изучение и описание предметной области

Распознавание лексем осуществляется лексическим анализатором. Входными данными является текст программы на языке С++, а выходными последовательность дескрипторов в виде таблиц лексем.

1.4.2 Постановка задачи

Необходимо моделировать работу лексического анализатора средствами языка программирования C++. Программа должна в виде таблиц представлять распознанные лексемы, определять ошибки, неизвестные лексемы и указывать на номер строки, а также формировать дескрипторный код и псевдокод .

1.4.6 Тестирование

Результаты тестирования описаны в таблице 1.2

*Таблица 1.2*

Тестирование

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Тестируемый модуль или подпрограмма | Входные данные | Кто проводил тестирование | Ожидаемый результат | Результаты тестирования |
| void CheckID(String ID) | buffer | Мороз Н.Р. | Проверка таблицы на пустоту, проверка лексемы в буфере на универсальность, в случае положительного результата занесение значения в таблицу. | Успех |
| bool isLetter(char b){ | allcode[i] |  | Возвращает 1 если является символом латинского алфавита | Успех |
| IsDigit(char) | allcode[i] |  | Возвращает 1 в случае если аргумент десятичная цифра | Успех |
| bool endOf(char b) | allcode[i] |  | Возвращает 1 в случае если аргумент является границей лексемы | Неудача |

Заключение

В процессе выполнения курсовой работы была смоделирована работа компилятора посредством реализации грамматика: Синтаксического и лексического анализа. Данная модель была реализована на языке программирования С++. Программа также оснащена визуальным интерфейсом для удобства пользования и наглядности полученных результатов.

# Список литературы

1. Ганичева, О. Г.Теория языков программирования и методы трансляции: Учеб. пособие. – Череповец: ГОУ ВПО ЧГУ, 2011. – 186 с.
2. Карпов Ю. Г. Теория автоматов. – СПб.: Питер, 2002. – 224 с.

# Приложение 1

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

Кафедра математического и программного обеспечения ЭВМ

Теория автоматов и формальных языков

УТВЕРЖДАЮ

Зав. кафедрой МПО ЭВМ

д. т.н., профессор \_\_\_\_\_\_\_\_ Ершов Е.В.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

КОНСТРУИРОВАНИЕ МОДЕЛЕЙ ЛЕКСИЧЕСКОГО И

СИНТАКСИЧЕСКОГО АНАЛИЗА

Техническое задание на курсовую работу

Листов 5

Руководитель: Ганичева О. Г.

Исполнитель:

студент 1ПИб-01-12оп

Мороз Н.Р.

2016 г.

# ВВЕДЕНИЕ

Задача курсовой работы состоит в создании программы моделирующей компилятор, распознавание лексем посредством синтаксического и лексического анализатора и их дальнейшей обработке.

Лексический анализатор — часть компилятора предназначенная для чтения и выделения лексем в тексте исходного кода программы. На вход анализатора подается текст программы, после обработки на выход получаем лексемы входного языка в виде таблиц.

Дескриптор – единица языка информационно-поисковой системы, соответствующая определенному ключевому или базовому понятию.

|  |  |
| --- | --- |
| Задание (инструкция) | Распознаватель |
| Структуры (struct) и объединения (union) | нисходящий |

# ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Основанием для разработки служит задание на курсовую работу.

1. Разработайте грамматику для моделирования работы компилятора. Для синтаксического анализа – КС-грамматику, для лексического анализа – регулярную грамматику.
2. Постройте лексический анализатор, который решает следующие задачи: выделяет из текста входной программы все лексемы, входящие в заданную языковую конструкцию, удаляет лишние пробелы и комментарии из входной строки. Построение лексического анализатора выполнить на основе конечного автомата. По конечному автомату построить соответствующую ему регулярную грамматику.
3. Постройте КС-грамматику, описывающую синтаксис инструкции языка программирования.
4. Докажите возможность вывода заданной синтаксической конструкции с помощью грамматики из п.1.3. Результатом работы синтаксического анализатора должно быть дерево вывода и соответствующее ему порождение рассматриваемой цепочки.
5. Путем эквивалентных преобразований получите приведенную грамматику.

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

Программа может быть полезна для людей занимающихся программированием на языке С++, студентов, преподавателей.

# ТРЕБОВАНИЯ К ПРОГРАММЕ

4.1 Требования к функциональным характеристикам

1. В задании I заданную грамматику представить в виде: правил КС-грамматики, в форме БНФ, на языке синтаксических диаграмм и регулярной грамматики.
2. Работа лексического анализатора должна моделироваться конечным автоматом, граф переходов которого должен быть представлен в пояснительной записке.
3. Выход лексического анализатора – дескрипторный код и псевдокод, таблица классов лексем, сообщения об ошибках: тип ошибки и в какой строке.
4. В ходе работы ЛА должны удаляться лишние пробелы и комментарии

4.2 Требования к надёжности

Необходимо предусмотреть удаление пробелов и комментариев. Реализовать вывод сообщений об ошибках: лексема и номер строки.

4.3 Минимальные требования к составу и параметрам технических средств

* Процессор с тактовой частотой 1,6 ГГц или большей;
* 1 ГБ ОЗУ (1,5 ГБ при выполнении в виртуальной машине);
* 5 ГБ доступного пространства на жестком диске;
* Жесткий диск 5400 об/мин;
* Видеоадаптер с поддержкой DirectX9 и разрешением экрана 1024x768 или выше;
* Наличие монитора, мышки, клавиатуры.

4.4 Требования к информационной и программной совместимости

Операционная система: Windows 7

Язык программирования: С++

# ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ

Программная документация должна содержать расчетно-пояснительную записку с приложениями:

Приложение 1: Техническое задание

Приложение 3: Текст программы

Приложение 4: Руководство пользователя

# СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

*Таблица 1.1*

Стадии и этапы разработки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа разработки ПО | Сроки разработки | Результат выполнения | Отметка о выполнении |
| 1 | 2 | 3 | 4 |
| Изучение темы курсовой работы и поставленных в ней задач | 3.11.2016 | Изучил тему, цель курсовой работы. | Выполнено |
| Написание технического задания | 6.11.2016 | Изучил методическое пособие по написанию технического задания. | Выполнено |
| Создание программы проекта | 8.12.2016 | Написание кода программы. | Выполнено |
| Написание и сдача курсовой работы | 22.12.2016 | Написание курсовой работы, оформление технического задания и РПЗ защита. | Выполнено |

# 

# ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ

*Таблица 1.2*

Порядок контроля и приёмки

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование контрольного этапа выполнения курсовой работы | Сроки контроля | Результат выполнения | Отметка о приёмке результата контрольного этапа |
| Техническое задание | 4.11.2016 | Выполнено |  |
| Сдача курсовой работы | 22.12.2016 | Выполнено |  |

# Приложение 2

# Текст программы

|  |
| --- |
|  |

# Приложение 3

# Руководство пользователя

Интерфейс лексического анализатора рис п3.1

|  |
| --- |
|  |

Рис. п3.1«Внешний вид лексического анализатора»