МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ НЕФТЕГАЗОВЫЙ УНИВЕРСИТЕТ»

Институт геологии и нефтегазодобычи

Кафедра кибернетических систем

ТЕОРИЯ КОМПИЛЯЦИИ. ЛЕКСИЧЕСКИЙ АНАЛИЗАТОР.

Методические указания к лабораторной работе по теме «Теория компиляции. Лексический анализатор» по дисциплине «Системное программное обеспечение»

Составитель У. В. Лаптева, старший преподаватель

Тюмень
ТюмГНГУ
2016

Теория компиляции. Лексический анализатор: методические указания к лабораторной работе по теме «Теория компиляции. Лексический анализатор» по дисциплине «Системное программное обеспечение» для студентов направления 27.03.04 «Управление в технических системах» дистанционной формы обучения / сост. У. В. Лаптева; Тюменский государственный нефтегазовый университет.– Тюмень: Издательский центр БИК, ТюмГНГУ, 2016.– 25 с.

Методические указания рассмотрены и рекомендованы к изданию на заседании кафедры кибернетических систем «04» марта 2016 года, протокол №9.

Аннотация

Методические указания к лабораторным работам по теме «Теория компиляции. Лексический анализатор» по дисциплине «Системное программное обеспечение» для студентов направления 27.03.04 «Управление в технических системах» очной и заочной формы обучения. Данная дисциплина изучается в одном семестре.

В методических указаниях приведена краткая теория трансляции и компиляции исходного текста программ. Разобран пример лексического анализа исходного текста программы и формирование дескрипторного текста.

Содержание

1. Цель и задачи дисциплины 4
2. Учебно-тематический план дисциплины «Системное программное обеспечение» 5
3. Указания по изучению разделов дисциплины 7
4. Указания по планированию времени, отведенного на изучение дисциплины 9
5. Указания по работе с литературой, конспектами лекций и учебно-методическими изданиями 10
6. Рекомендации по подготовке к лабораторным и практическим занятиям 11
7. Рекомендации по выполнению контрольных работ 14
8. Рекомендации по организации самостоятельной работы студентов 16
9. Указания по самоконтролю и подготовке к контрольному тестированию 20
10. Рекомендации по подготовке к экзамену 21
11. Список литературы 22

1. Введение

Методические указания по выполнению лабораторной работы на тему «Теория компиляции. Лексический анализатор» предназначены для студентов направления 27.03.04 «Управление в технических системах» очной и заочной формы обучения и содержат основные теоретические положения трансляции программ, указания по выполнению первого этапа трансляции исходного текста программы – лексического анализа.

После выполнения данной лабораторной работы студент должен:

1) знать основные этапы работы транслятора, алгоритм работы лексического анализатора исходного текста программы, структуру дескрипторного текста.
2) уметь демонстрировать работу лексического анализатора на примере.

2. Содержание лабораторной работы

2.1. Цель работы

Получение дескрипторного текста как результата выполнения этапа лексического анализа исходного теста программы.

2.2. Основные теоретические положения

Компилятор – это программа, которая считывает текст программы, написанной на одном языке – исходном, и транслирует (переводит) его в эквивалентный текст на другом языке – целевом (рисунок 1). Одним из важных моментов трансляции является сообщение пользователю о наличии ошибок в исходной программе.

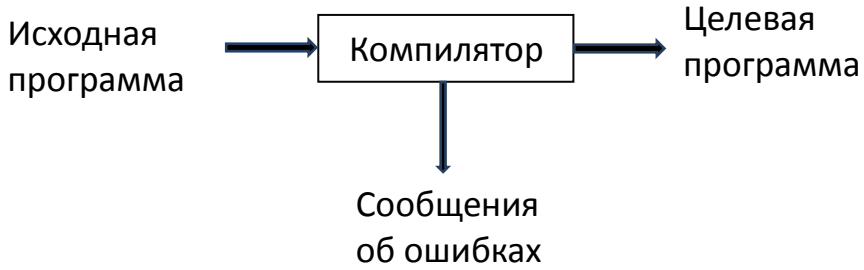

Рисунок 1. Компилятор

Используются тысячи исходных языков, от традиционных (Fortran, Pascal, …) до специализированных.

Целевые языки также разнообразны – это могут быть другие языки программирования, различные машинные языки (от языков микропроцессоров до суперкомпьютеров).

Компиляторы классифицируются как однопроходные, многопроходные, исполняющие, отлаживающие, оптимизирующие – в зависимости от

предназначения, принципов и технологий их создания. Несмотря на такое разнообразие, основные задачи, выполняемые различными компиляторами одни и те же.

Первые компиляторы появились в начале 1950-х гг., сегодня нельзя однозначно сказать, когда появился первый компилятор, т.к. в те годы проводилось множество экспериментов и разработок различными независимыми группами. В основном целью этих разработок было преобразование в машинный код арифметических формул.

Модель анализа – синтеза компиляции.

Компиляция состоит из двух частей: анализа и синтеза. Анализ – это разбиение исходной программы на составные части и создание её промежуточного представления. Синтез – конструирование требуемой целевой программы из промежуточного представления.

В процессе анализа определяются и записываются в иерархическую древовидную структуру операции, заданные исходной программой. Часто используется специальный вид дерева, называемого синтаксическим (или деревом синтаксического разбора), в котором каждый узел представляет операцию, а его дочерние узлы – аргументы операции. Пример синтаксического дерева разбора инструкции присвоения position:=initial+rate*60 показан на рисунке 2.

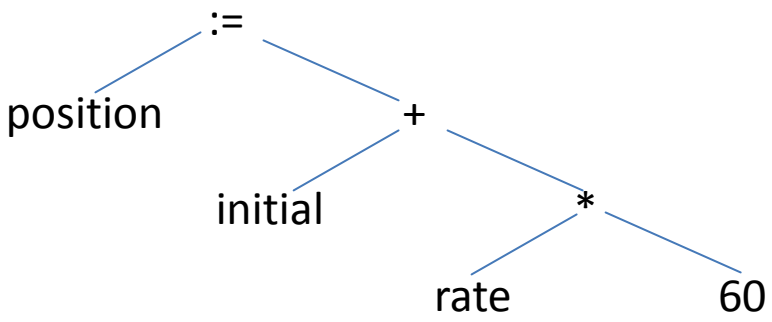

Рисунок 2. Синтаксическое дерево разбора для оператора position:=initial+rate*60

Многие программные инструменты, работающие с исходными программами, сначала выполняют определённый вид анализа. Примеры таких инструментов:

1. *Структурный редактор* – анализирует текст программы и помещает в исходную программу соответствующую иерархическую структуру (если пользователь введёт while, редактор добавит соответствующее ему ключевое слово do и предложит ввести условное выражение между ними);
2. *Программы форматированного вывода на печать* – анализирует программу и распечатывает таким образом, чтобы её структура была максимально ясной. Например, комментарии могут быть выделены

специальным шрифтом, а операторы – выведены с отступами, указывающими уровень вложенности в иерархической структуре операторов;

3. *Статические проверяющие программы* – анализируют программы и пытаются найти потенциальные ошибки без запуска программы. Например, статическая проверяющая программа может определить невыполнение какой-то части исходной программы или использование некоторой переменной до её объявления;
4. *Интерпретаторы.* Вместо создания целевой программы в результате трансляции интерпретатор выполняет операции, указанные в исходной программе. Некоторые языки «очень высокого уровня», типа APL, обычно интерпретируются, поскольку имеется множество атрибутов данных, таких как размер или тип массива, которые не могут быть определены в процессе компиляции.

Контекст компилятора.

При создании целевой программы, кроме компилятора, можт потребоваться и ряд других программ. Исходная программа может быть разделена на модули, хранимые в отдельных файлах.задача сбора исходной программы иногда поручается отдельной программе – препроцессору, который может также раскрывать в тексте исходной программы сокращения, называемые макросами.

На рисунке 3 показана типовая схема компиляции. Целевая программа, создаваемая компилятором, может потребовать дополнитльной обработки перед запуском. Компилятор, представленный на рисунке 3, создаёт ассемблерный код, который переводится ассемблером в машинный код, а затем связывается («линкуется») совместно с некоторыми библиотечными программами в реально запускаемый на машине код.

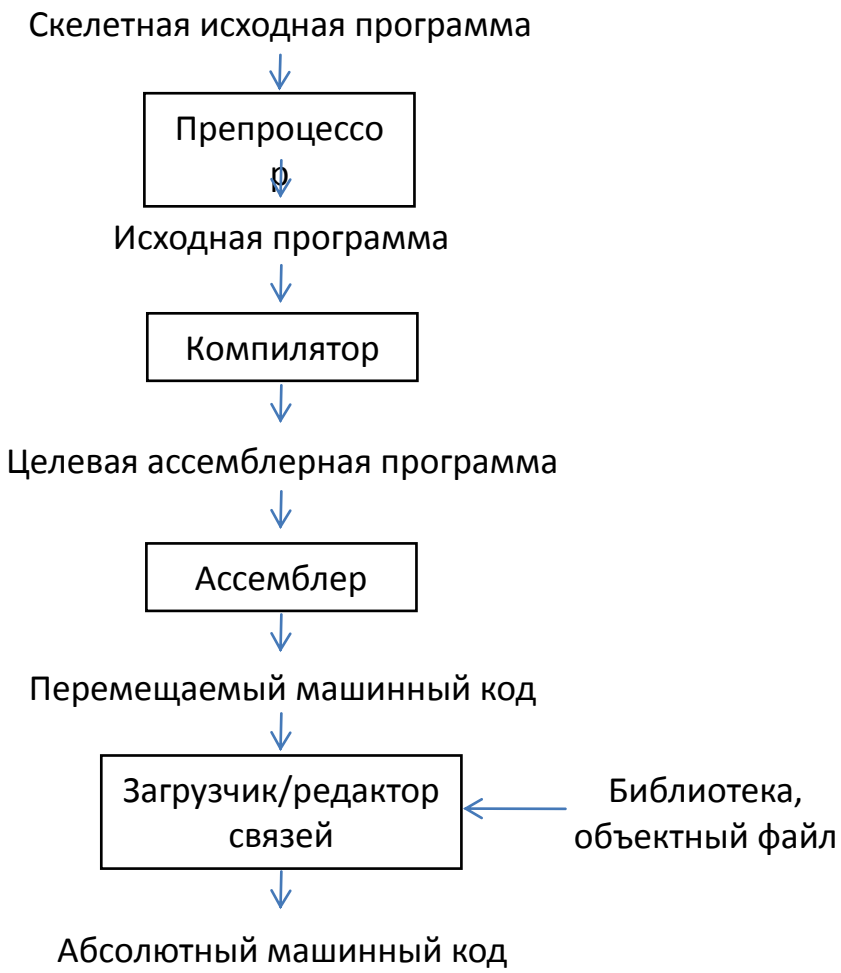

Рисунок 3. Схема обработки исходной программы

Анализ исходной программы.

При компиляции анализ состоит из трёх фаз:

1. *Линейный анализ*, при котором поток символов исходной программы считывается слева направо и группируется в токены (token), представляющие собой последовательность символов с определённым совокупным значением.
2. *Иерархический анализ (синтаксический анализ)*, при котором символы или токены иерархически группируются во вложенные конструкции с совокупным значением.
3. *Семантический анализ*, позволяющий проверить, насколько корректно совместное размещение компонентов программы.

В компиляторах линейный анализ называется *лексическим*. Например, при лексическом анализе символы в инструкции присвоения

position := initial + rate * 60

будут сгруппированы в следующие токены:

1. Идентификатор position

2. Символ присвоения :=
3. Идентификатор initial
4. Знак сложения
5. Идентификатор rate
6. Знак умножения
7. Число 60

Пробелы, разделяющие символы этих токенов, при лексическом анализе обычно отбрасываются.

Фазы компилятора.

Концептуально компилятор работает пофазно, причем в процессе прохождения исходной программой каждой фазы происходит её преобразование из одного представления в другое. Типовое разбиение компилятора на фазы показано на рисунке 4. На практике некоторые фазы могут быть сгруппированы вместе и промежуточные представления программы внутри таких групп могут явно не строиться.

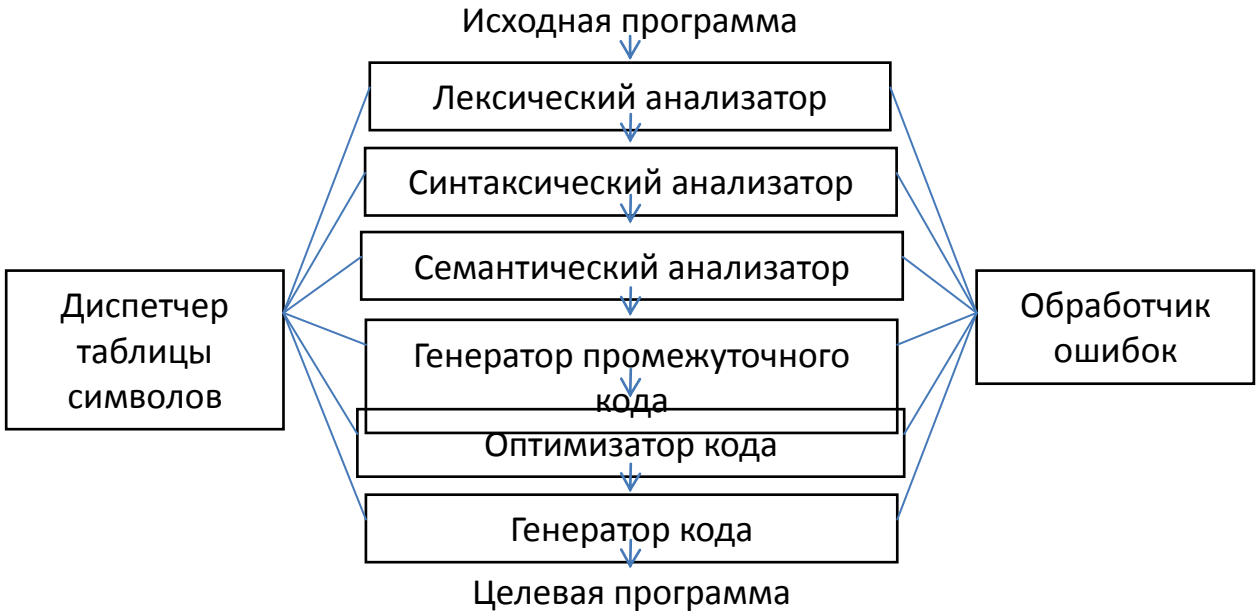

Рисунок 4. Фазы компилятора

В процессе трансляции изменяется внутреннее представление исходной программы. На рисунке 5 показано внутреннее представление инструкции position := initial + rate * 60 после каждой фазы.

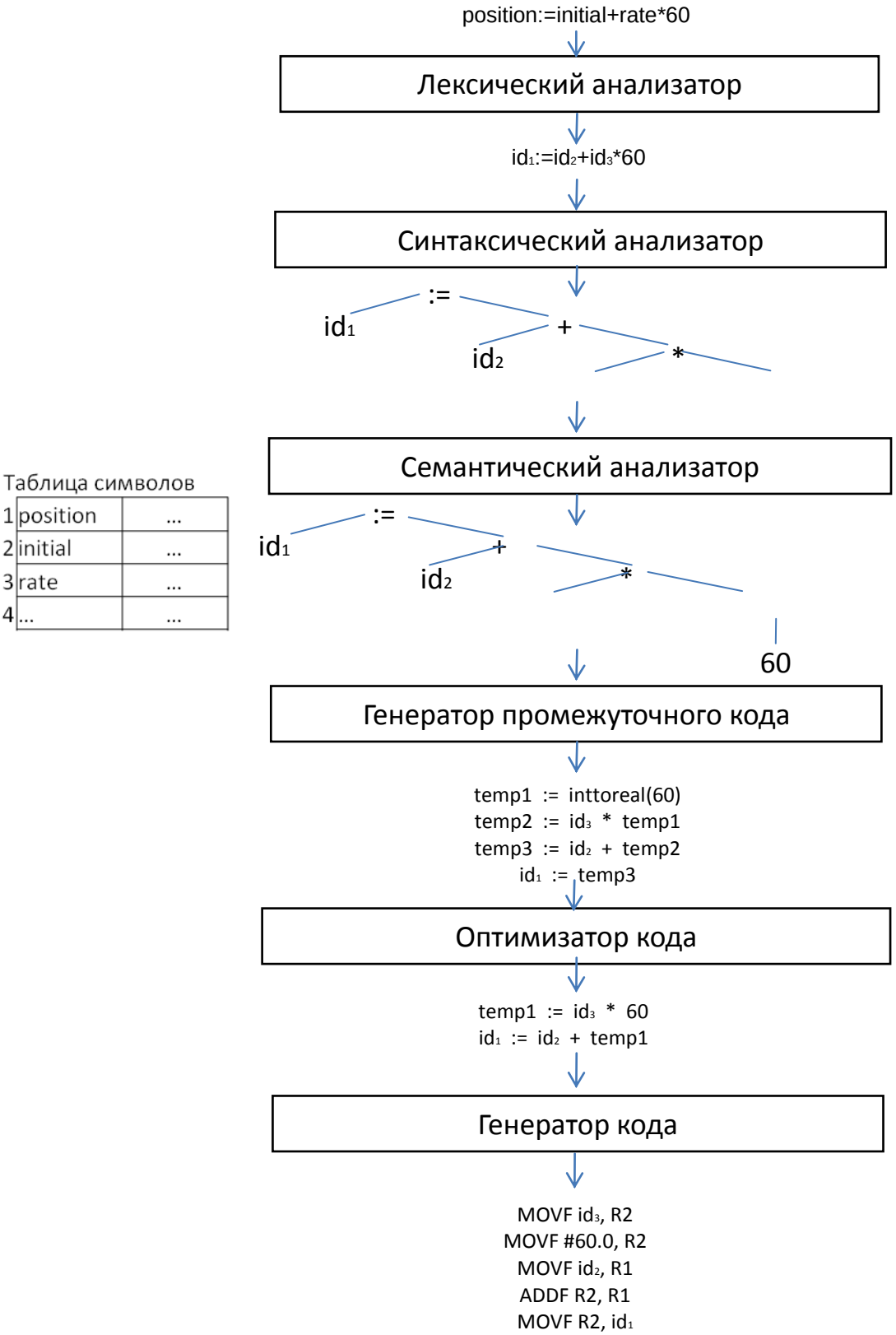

Рисунок 5. Трансляция инструкции

Лексический анализ

Этап лексической обработки текста позволяет сократить общее время компиляции программы. Это достигается за счет того, что исходная программа,

представленная на входе компилятора в виде непрерывной последовательности символов, на этапе лексической обработки преобразуется к некоторому стандартному виду, что облегчает дальнейший анализ.

Лексический анализ важен для процесса компиляции по нескольким причинам:

a) замена в программе идентификаторов, констант, ограничителей и служебных слов лексемами делает представление программы более удобным для дальнейшей обработки;

b) лексический анализ уменьшает длину программы, устраняя из ее исходного представления несущественные пробелы и комментарии;

c) если будет изменена кодировка в исходном представлении программы, то это отразится только на лексическом анализаторе.

Задачи и функционирование блока лексического анализа

Лексический анализ – процесс предварительной обработки исходной программы, на котором основные лексические единицы программы – *лексемы*: ключевые (служебные) слова, идентификаторы, метки, константы приводятся к единому формату и заменяются условными кодами или ссылками на соответствующие таблицы, а комментарии исключаются из текста программы.

Входами лексического анализа являются поток образов лексем – дескрипторов и таблицы, в которых хранятся значения выделенных в программе лексем.

Дескриптор – это пара вида: (<тип лексемы>,<указатель>),

где <тип лексемы> - это числовой код класса лексемы;

<указатель> - это либо начальный адрес области основной памяти, в которой хранится адрес этой лексемы, либо число, адресующее элемент таблицы, в которой хранится значение этой лексемы.

Количество классов лексем в языках программирования может быть различным. Наиболее распространённые классы:

- Идентификаторы;
- Служебные (ключевые) слова;
- Разделители;
- Константы.

Могут вводиться и другие классы. В общем случае все выделяемые классы являются либо конечными (ключевые слова, разделители и др.) – классы фиксированных для данного языка программирования слов, либо бесконечными или очень большими (идентификаторы, константы и др.) – классы переменных для данного языка программирования слов.

С этих позиций коды образов лексем (дескрипторов) из конечных классов всегда одни и те же в различных программах для данного компилятора. Коды образов лексем из бесконечных классов различны для разных программ и формируются каждый раз на этапе лексического анализа.

Содержимое таблиц, в особенности таблицы идентификаторов, в дальнейшем пополняется на этапе семантического анализа исходной программы и используется на этапе генерации объектной программы.

Работа лексического анализатора заключается в следующем: первоначально в тексте исходной программы лексический анализатор выделяет последовательность символов, которая по его предположению должна быть словом в программе, т.е. лексемой. Может выделяться не вся последовательность, а только один символ, который считается началом лексемы. Это наиболее ответственная часть работы лексического анализатора. Более просто она реализуется для тех языков программирования, в которых слова в программе отделяются друг от друга специальными разделителями (например, пробелами), либо запрещено использование служебных слов в качестве переменных, либо классы лексем опознаются по вхождению первого (последнего) символа лексемы.

После этого (или параллельно с этим) проводится идентификация лексемы. Она заключается в сборке лексемы из символов, начиная с выделенного на предыдущем этапе, и проверки правильности записи лексемы данного класса.

Идентификация лексемы из конечного класса выполняется путем сравнения её с эталонным значением. Основная проблема здесь – минимизация времени поиска эталона. В общем случае может понадобиться полный перебор слов данного класса, в особенности для случая, когда выделенное для опознания слово содержит ошибку.

Для идентификации лексем из бесконечных классов используются специальные методы сборки лексем с одновременной проверкой правильности написания лексемы. При построении этих алгоритмов широко применяется формальный математический аппарат – теория регулярных языков, грамматик и конечных распознавателей.

При успешной идентификации значение лексемы из бесконечного класса помещается в таблицу идентификации лексем данного класса. При этом необходимо предварительно проверить: не хранится ли там уже значение данной лексемы, т.е. необходимо проводить просмотр элементов таблицы. Если её там нет, то значение помещается в таблицу. При этом таблица должна допускать расширение. Опять же для уменьшения времени доступа к элементам таблицы она должна быть специальным образом организована, при этом должны использоваться специальные методы ускоренного поиска элементов.

В ходе проведения успешной идентификации лексемы формируется её образ – дескриптор, он помещается в выходной поток лексического анализатора. В случае неуспешной идентификации формируется сообщение об ошибках в написании слов программы.

В ходе лексического анализа могут выполняться и другие виды лексического контроля, в частности, проверяется парность скобок и других парных символов.

Выходной поток с лексического анализатора в дальнейшем поступает на вход синтаксического анализатора. Имеется две возможности их связи:

- Раздельная связь, при которой выход лексического анализатора формируется полностью и затем передается синтаксическому анализатору;
- Нераздельная связь, когда синтаксическому анализатору требуется очередной образ лексемы, он вызывает лексический анализатор, который генерирует требуемый дескриптор и возвращает управление синтаксическому анализатору.

Таким образом, процесс лексического анализа может быть достаточно простым, но в смысле времени компиляции оказывается довольно долгим. Больше половины времени, затрачиваемого компилятором на компиляцию, приходится на этап лексического анализа.

ПРИМЕР РАБОТЫ ЛЕКСИЧЕСКОГО АНАЛИЗАТОРА

Используемый код типов лексем:
10 – ключевое слово;
20 – разделитель;
30 – идентификатор;
40 – константа.
Вход лексического анализатора:

```
PROGRAM PRIMER;
  VAR X, Y, Z : REAL;
   BEGIN
          X:=5;
          Y:=5;
        Z:= X+Y;
   END;
```

Внутренние таблицы лексического анализатора:

Таблица ключевых слов

№ п/п	Ключевое слово
1	PROGRAM
2	BEGIN
3	END
4	FOR
5	REAL
6	VAR
…	…

Таблица разделителей

№ п/п	Разделитель
1	;
2	,
3	+
4	-
5	/
6	*
7	:
8	=
9	.

…	…

Выход лексического анализатора:

Таблица идентификаторов

№ п/п	Идентификатор
1	PRIMER
2	X
3	Y
4	Z

Таблица констант

№ п/п	Значение константы
1	5

Дескрипторный текст:
(10,1) (30,1) (20,1)
(10,6) (30,2) (20,2) (30,3) (20,2) (30,4) (20,7) (10,5) (20,1)
(10,2)
(30,2) (20,7) (20,8) (40,1) (20,1)
(30,3) (20,7) (20,8) (40,1) (20,1)
(30,4) (20,7) (20,8) (30,2) (20,3) (30,3) (20,1)
(10,3) (20,9)

Пример реализации лексического анализатора.

```
#include <stdio.h>
#include <ctype.h>
# define NONE -1
# define NUM 256
main()
{
        int lineno = 1;
        int tokenval = 0;
        int t;
        while(1)
                {
                        t = getchar();
                        if (t == ' ' || t == '\t') ;
                        else if (t == '\n')
                        lineno++;
                        else if (isdigit(t))
                                {
                                        tokenval = t - '0';
                                        t = getchar();
```

```
                        while(isdigit(t))
                            {
                                tokenval = tokenval * 10 + t - '0';
                                t = getchar();
                            }
                        ungetc(t, stdin);
                        return t;
                    }
                else
                    {
                        tokenval = 0;
                        return t;
                    }
            }
}
```

Рисунок 6. – Пример реализации лексического анализатора.

2.3. Порядок (алгоритм) выполнения работы

1. Ознакомиться с теорией компиляции и лексического анализа исходной программы.
2. Написать простейшую программу на любом известном вам языке программирования согласно варианту задания из таблицы 1 (номер варианта выбирается по алфавитному списку группы).
3. Выполнить лексический анализ написанной программы согласно приведенному примеру, т.е. получить дескрипторный текст.
4. Оформить отчет согласно пунктам порядка выполнения работы.

2.4. Задание на выполнение лабораторной работы

Таблица 1

Варианты заданий

Вариант	**Условие задачи**
1	2
1	Даны два числа. Найти среднее арифметическое их квадратов и среднее арифметическое их модулей.
2	Найти периметр и площадь прямоугольного треугольника, если даны длины его катетов a и b.
3	Найти длину окружности и площадь круга заданного радиуса R. В качестве значения Pi использовать 3.14.

1	2
4	Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.
5	Найти площадь кольца, внутренний радиус которого равен R1, а внешний радиус равен R2 (R1 < R2). В качестве значения Pi использовать 3.14.
6	Дана сторона равностороннего треугольника. Найти площадь этого треугольника и радиусы вписанной и описанной окружностей.
7	Найти периметр и площадь равнобедренной трапеции с основаниями a и b (a > b) и углом alpha при большем основании (угол дан в радианах).
8	Найти расстояние между двумя точками с заданными координатами (x1, y1) и (x2, y2).
9	Даны три целых числа. Возвести в квадрат отрицательные числа и в третью степень — положительные (число 0 не изменять).
10	Из трех данных чисел выбрать наименьшее.
11	Из трех данных чисел выбрать наибольшее.
12	Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной сумму этих значений, а если равны, то присвоить переменным нулевые значения.
13	Даны целочисленные координаты точки на плоскости. Если точка не лежит на координатных осях, то вывести 0. Если точка совпадает с началом координат, то вывести 1. Если точка не совпадает с началом координат, но лежит на оси OX или OY, то вывести соответственно 2 или 3.
14	Даны вещественные координаты точки, не лежащей на координатных осях OX и OY. Вывести номер координатной четверти, в которой находится данная точка.
15	На числовой оси расположены три точки: A, B, C. Определить, какая из двух последних точек (B или C) расположена ближе к A, и вывести эту

	точку и ее расстояние от точки A.
16	Дан номер некоторого года (положительное целое число). Вывести соответствующий ему номер столетия, учитывая, что, к примеру, началом 20 столетия был 1901 год.
17	Дан номер некоторого года (положительное целое число). Вывести число дней в этом году, учитывая, что обычный год насчитывает 365 дней, а високосный — 366 дней. Високосным считается год, делящийся на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400 (например, годы 300, 1300 и 1900 не являются високосными, а 1200 и 2000 — являются).
18	Дано целое число, лежащее в диапазоне от –99 до 99. Вывести строку — словесное описание данного числа вида "отрицательное двузначное число", "нулевое число", "положительное однозначное число" и т.д.
19	Дано целое число, лежащее в диапазоне от 1 до 999. Вывести строку — словесное описание данного числа вида "четное двузначное число", "нечетное трехзначное число" и т.д.
20	Дан номер месяца (1 — январь, 2 — февраль, ...). Вывести название соответствующего времени года ("зима", "весна" и т.д.).
21	Дан номер месяца (1 — январь, 2 — февраль, ...). Вывести число дней в этом месяце для невисокосного года.
22	Дано целое число в диапазоне 0 – 9. Вывести строку — название соответствующей цифры на русском языке (0 — "ноль", 1 — "один", 2 — "два", ...).
23	Арифметические действия над числами пронумерованы следующим образом: 1 — сложение, 2 — вычитание, 3 — умножение, 4 — деление. Дан номер действия и два числа A и B (B не равно нулю). Выполнить над числами указанное действие и вывести результат.
24	Единицы длины пронумерованы следующим

	образом: 1 — дециметр, 2 — километр, 3 — метр, 4 — миллиметр, 5 — сантиметр. Дан номер единицы длины и длина отрезка L в этих единицах (вещественное число). Вывести длину данного отрезка в метрах.
25	Единицы массы пронумерованы следующим образом: 1 — килограмм, 2 — миллиграмм, 3 — грамм, 4 — тонна, 5 — центнер. Дан номер единицы массы и масса тела M в этих единицах (вещественное число). Вывести массу данного тела в килограммах.
26	Робот может перемещаться в четырех направлениях ("С" — север, "З" — запад, "Ю" — юг, "В" — восток) и принимать три цифровые команды: 0 — продолжать движение, 1 — поворот налево, –1 — поворот направо. Дан символ C — исходное направление робота и число N — посланная ему команда. Вывести направление робота после выполнения полученной команды.
27	Локатор ориентирован на одну из сторон света ("С" — север, "З" — запад, "Ю" — юг, "В" — восток) и может принимать три цифровые команды: 1 — поворот налево, –1 — поворот направо, 2 — поворот на 180 градусов. Дан символ C — исходная ориентация локатора и числа N1 и N2 — две посланные ему команды. Вывести ориентацию локатора после выполнения данных команд.
28	Дано вещественное число A и целое число N (> 0). Вывести все целые степени числа A от 1 до N.
29	Дано целое число N (> 1). Вывести наименьшее целое K, при котором выполняется неравенство 3K > N, и само значение 3K.
30	Найти минимальный и максимальный из данных десяти ненулевых элементов.

2.5. Основные правила по технике безопасности

Настоящие правила распространяется на студентов, эксплуатирующих средства вычислительной техники и периферийное оборудование и содержат общие указания по безопасному применению электрооборудования в университете.

Эксплуатирующие средства вычислительной техники и периферийное оборудование студенты могут подвергаться опасным и вредным воздействия, которые по природе действия подразделяются на следующие группы:

- поражение электрическим током,
- механические повреждения
- электромагнитное излучение
- инфракрасное излучение
- опасность пожара
- повышенный уровень шума и вибрации

Для снижения или предотвращения влияния опасных и вредных факторов необходимо соблюдать «Санитарные правила и нормы. Гигиенические требования к видеодисплейным терминалам, персональным электронно-вычислительным машинам и организации работы» (Утверждено Постановлением Госкомсанэпиднадзора России от 14 июля 1996 г. N 14 СанПиН 2.2.2.542-96).

При пользовании средствами вычислительной техники и периферийным оборудованием каждый студент должен внимательно и осторожно обращаться с электропроводкой, приборами и аппаратами и всегда помнить, что пренебрежение правилами безопасности угрожает и его здоровью, и его жизни.

Во избежание поражения электрическим током необходимо твердо знать и выполнять следующие правила безопасного пользования электроэнергией:

1. Необходимо убедиться в исправности электропроводки, выключателей, штепсельных розеток и заземления. При обнаружении неисправности немедленно оповестить преподавателя. Продолжение работы возможно только после устранения неисправности.

2. Во избежание повреждения изоляции проводов и возникновения коротких замыканий не разрешается:

а) вешать что-либо на провода;

б) закладывать провода и шнуры за батареи отопительной системы;

в) выдергивать штепсельную вилку из розетки за шнур, усилие должно быть приложено к корпусу вилки.

3. Для исключения поражения электрическим током запрещается:

а) часто включать и выключать компьютер без необходимости;

б) прикасаться к экрану и к тыльной стороне блоков компьютера;

в) работать на средствах вычислительной техники и периферийном оборудовании мокрыми руками;

г) работать на средствах вычислительной техники и периферийном оборудовании, имеющих нарушения целостности корпуса, нарушения изоляции проводов, неисправную индикацию включения питания, с признаками электрического напряжения на корпусе;

д) класть на средства вычислительной техники и периферийное оборудование посторонние предметы.

7. Во избежание поражения электрическим током, при пользовании электроприборами нельзя касаться одновременно каких-либо батарей отопления, металлических конструкций, соединенных с землей.

9. При обнаружении оборвавшегося провода необходимо немедленно сообщить об этом преподавателю.

На рабочем месте запрещается иметь огнеопасные вещества.

В аудиториях запрещается:

а) зажигать огонь;

б) курить;

в) сушить что-либо на отопительных приборах;

г) закрывать вентиляционные отверстия в электроаппаратуре

Источниками воспламенения являются:

а) искра при разряде статического электричества

б) искры от электроборудования

в) искры от удара и трения

г) открытое пламя

При возникновении пожароопасной ситуации или пожара студент должен немедленно оповестить преподавателя о пожаре.

2.6. Перечень оборудования, используемого при выполнении лабораторной работы

2.7. Содержание и форма отчета о проделанной работе

Результат лабораторной работы должен быть оформлен в виде отчёта.
Титульный лист отчёта приведён в Приложении А.
Содержание отчёта:

1. Цель работы
2. Вариант задания (ий)
3. Описание решения задания (ий)
4. Полученный результат

Правила оформления отчёта

Общие требования

Отчёт должен быть выполнен с использованием компьютера и отпечатана на принтере на одной стороне листа белой бумаги формата А4.

Цвет шрифта должен быть черным, высота букв, цифр и других знаков - 14 кегль. Текст отчёта следует печатать, соблюдая следующие размеры полей: правое – 10 мм, верхнее и нижнее – 20 мм, левое– 30 мм, межстрочный интервал – 1, выравнивание по ширине. Выделение фрагментов в тексте не допускается полужирным шрифтом, либо курсивом.

Опечатки, описки и графические неточности, обнаруженные в процессе подготовки отчёта, допускается исправлять подчисткой или закрашиванием

белой краской и нанесением на том же месте исправленного текста (графики) машинописным способом или черными чернилами, пастой или тушью - рукописным способом. Повреждение листов текстовых документов, помарки и следы не полностью удаленного прежнего текста (графики) не допускаются.

Построение отчёта

Основную часть отчёта следует делить на разделы, подразделы. Разделы и подразделы следует нумеровать арабскими цифрами и записывать с абзацного отступа (1,5 см.). Разделы должны иметь порядковую нумерацию в пределах всего текста.

Пример – 1, 2, 3 и т.д.

Номер подраздела или пункта включает номер раздела и порядковый номер подраздела или пункта, разделенные точкой.

Пример – 1.1, 1.2, 1.3 и т.д.

После номера раздела, подраздела в тексте точку не ставят. Если раздел или подраздел имеет только один пункт, то нумеровать его не следует. Разделы, подразделы должны иметь заголовки. Заголовки разделов, подразделов следует печатать с абзацного отступа с прописной буквы без точки в конце, не подчеркивая. Переносы слов в заголовках не допускаются.

Внутри пунктов могут быть приведены перечисления. Перед каждым перечислением следует ставить дефис или, при необходимости ссылки в тексте документа на одно из перечислений, строчную букву русского алфавита (за исключением ё, з, й, о, ч, ь, ы, ъ), после которой ставится скобка. Для дальнейшей детализации перечислений необходимо использовать арабские цифры, после которых ставится скобка, а запись производится с абзацного отступа, как показано в примере.

Пример

а)_____

б)_____

1)_____

2)_____

в)_____

Нумерация страниц отчёта должна быть сквозная. Номер страницы располагается внизу страницы по центру.

Иллюстрации

Иллюстрации (чертежи, графики, схемы, компьютерные распечатки, фотоснимки, диаграммы) следует располагать в отчёте непосредственно после текста, в котором они упоминаются впервые, или на следующей странице. Иллюстрации могут быть в компьютерном исполнении, в том числе и цветные. На все иллюстрации должны быть даны ссылки в записке.

Иллюстрации следует нумеровать арабскими цифрами сквозной нумерацией. Если рисунок один, то он обозначается «Рисунок 1». Слово рисунок и его наименование располагают по середине строки. Слово «Рисунок» и наименование располагают следующим образом: Рисунок 1 – Пример программы.

При ссылках на иллюстрации следует писать «… в соответствии с рисунком 2».

Таблицы

Название таблицы, при ее наличии, должно отражать ее содержание, быть точным, кратким. Название таблицы следует помещать над таблицей слева, без абзацного отступа в одну строку с ее номером через тире. Таблицу следует располагать в отчёте непосредственно после текста, в котором она упоминается впервые, или на следующей странице. На все таблицы должны быть ссылки в отчёте. При ссылке следует писать слово «таблица» с указанием ее номера. Таблицу с большим количеством строк допускается переносить на следующую страницу. При переносе части таблицы на следующую страницу слово «Таблица» и ее номер указывается один раз слева над первой частью таблицы, над другими частями пишут справа слово «Продолжение» и указывают номер таблицы, например: «Продолжение таблицы 1».

Таблицы следует нумеровать арабскими цифрами сквозной нумерацией.

Заголовки граф и строк таблицы следует писать с прописной буквы в единственном числе, а подзаголовки граф – со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблиц точки не ставят. Допускается применять размер шрифта в таблице меньший, чем в тексте.

Разделять заголовки и подзаголовки боковика и граф диагональными линиями не допускается.

Формулы и уравнения

Уравнения и формулы следует выделять из текста в отдельную строку. Выше и ниже каждой формулы или уравнения должно быть оставлено не менее одной свободной строки. Если уравнение не умещается в одну строку, то оно должно быть перенесено после знака равенства (=) или после знаков плюс (+), минус (-), умножения (х), деления (:), или других математических знаков, причем знак в начале следующей строки повторяют. При переносе формулы на знаке, символизирующем операцию умножения, применяют знак “х”.

Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в которой они даны в формуле.

Формулы следует нумеровать порядковой нумерацией в пределах всего отчёта арабскими цифрами в круглых скобках в крайнем правом положении на строке.

$$A=a:Ь, \quad (1)$$

$$B=c:e. \quad (2)$$

Ссылки в тексте на порядковые номера формул дают в скобках.

Пример – ... в формуле (1).

2.8. Диагностические материалы (контрольные вопросы, задания, упражнения)

3. Критерии оценки работы обучающегося

4. Список основной и дополнительной литературы с указанием определённых страниц, необходимых при подготовке обучающихся к лабораторной работе

а) Основная литература:

1) Гагарина, Л. Г., Кокорева, Е. В., Введение в теорию алгоритмических языков и компиляторов, Изд-во: ФОРУМ, 2009. - 176 с.: ил.

2) Дейтел, Харви, Операционные системы / пер. с англ. - 3-е изд., Изд-во: БИНОМ - 2007. - 704 с.: ил. - (в пер.).

3) Иртегов, Д. В., Введение в операционные системы, Изд-во: БХВ-Петербург, 2002. - 614 с.

4) Молчанов, А. Ю., Системное программное обеспечение, Изд-во: Питер, 2003. - 396 с.: ил. - (Учебник для вузов).

5) Шаньгин, В. Ф., Информационная безопасность компьютерных систем и сетей, Изд-во: ФОРУМ: ИНФРА-М, 2009. - 416 с.: ил.

б) Дополнительная литература

1) Гордеев, А. В., Системное программное обеспечение, Изд-во: Питер, 2001. - 734 с.

2) Горнец, Н. Н., Организация ЭВМ и систем, Изд-во: ИЦ "Академия", 2006. - 317 с.

3) Крупник, А. Б., Изучаем Ассемблер, Изд-во: Питер, 2004. - 249 с.

4) Олифер, В. Г., Олифер Н. А., Сетевые операционные системы, Изд-во: Питер, 2002. - 539 с.: ил.

5) Сорокина, С. И., Тихонов, А. Ю., Щербаков, А. Ю., Программирование драйверов и систем безопасности, Изд-во: БХВ-Петербург, 2003. - 243 с.: ил.

6) Цилькер, Б. Я., Орлов, С. А., Организация ЭВМ и систем, Изд-во: Питер, 2004. - 668 с.: ил.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ТЮМЕНСКИЙ ИНДУСТРИАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ГЕОЛОГИИ И НЕФТЕГАЗОДОБЫЧИ
Кафедра Кибернетических систем

ОТЧЁТ

по лабораторной работе
на тему: « »
по дисциплине: «Системное программное обеспечение»

Выполнил
студент *Фамилия И.О.*

группа _____ .

Дата защиты_____ Баллы _____

Тюмень, 2016

Учебное издание

ТЕОРИЯ КОМПИЛЯЦИИ.
ЛЕКСИЧЕСКИЙ АНАЛИЗАТОР.

Методические указания к лабораторной работе по теме «Теория компиляции. Лексический анализатор» по дисциплине «Системное программное обеспечение» для студентов направления 27.03.04 «Управление в технических системах» дистанционной формы обучения.

Составитель
ЛАПТЕВА Ульяна Викторовна

В авторской редакции

Подписано в печать Формат 60х90 1/16. Усл. печ. л. 1,5.

Тираж 35 экз. Заказ №

Библиотечно-издательский комплекс
федерального государственного бюджетного образовательного
учреждения высшего профессионального образования
«Тюменский государственный нефтегазовый университет».
625000, Тюмень, ул. Володарского, 38.

Типография библиотечно-издательского комплекса.
625039, Тюмень, ул. Киевская, 52.