МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |  |
| --- | --- |
| Институт (факультет) | Информационных технологий |
| Кафедра | Математическое и программное обеспечение ЭВМ |

КУРСОВАЯ РАБОТА

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

|  |  |
| --- | --- |
| на тему | Разработка алгоритмического обеспечения и построение |
| лексического анализатора компилятора | |

|  |
| --- |
| Выполнил студент группы |
| 1ПИб-02-1оп-22 |
| направления подготовки (специальности) |
| 09.03.04, Программная инженерия |
| шифр, наименование |
|  |
| Новиков Игорь Станиславович |
| фамилия, имя, отчество |

|  |
| --- |
| Руководитель |
| Ганичева Оксана Георгиевна |
| фамилия, имя, отчество |
| Доцент |
| должность  Пышницкий Константин Михайлович |
| фамилия, имя, отчество |
| Старший Преподаватель |
| должность |

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

Аннотация

В данной курсовой работе рассматривается тема разработка алгоритмического обеспечения и построение лексического анализатора компилятора. В работе описывается процесс проектирования и реализации программы. Основной целью работы является разработка приложения, которое будет принимать код некоторой программы и обрабатывать его подобно компилятору. Результат работы может быть использован в образовательных целях.

Оглавление

[Введение 4](#_Toc166190503)

[1. Описание предметной области 5](#_Toc166190504)

[1.1. Лексический анализатор 6](#_Toc166190505)

[1.2. Синтаксический анализатор 8](#_Toc166190506)

[2. Основная часть 9](#_Toc166190507)

# Введение

Современное программирование немыслимо без компиляторов и лексических анализаторов, которые являются важными инструментами в развитии языков программирования и интерпретаторов. Создание компилятора лексического анализатора - сложный и увлекательный процесс, требующий глубоких знаний теории компиляции и алгоритмов.

Цель данной курсовой работы - рассмотреть основные этапы создания компилятора лексического анализатора и выявить основные компоненты и принципы решения этой задачи. Изучение данной темы позволит глубже понять функциональные принципы компилятора, его важность в разработке программного обеспечения и потенциал для оптимизации процесса компиляции.

В рамках работы также будут рассмотрены различные подходы к построению компиляторов, изучены их особенности и применение в современном программировании.

Поскольку компиляторы и лексические анализаторы играют важную роль в создании эффективного и надежного программного обеспечения, их изучение имеет большое значение для IT-специалистов. Далее подробно рассматриваются основные этапы создания лексического анализатора-компилятора, его влияние на процесс разработки программного обеспечения и возможности оптимизации этого процесса.

1. Описание предметной области

**Основой любого алгоритмического языка является алфавит, т.е. набор допустимых элементарных знаков (букв, цифр, служебных символов). Они могут объединяться в слова, т. е. в простейшие конструкции языка, неделимые символы, имеющие определенный смысл. Одиночный символ может быть словом. Словарный состав языка (набор допустимых символов) вместе с описанием способов их представления есть лексика языка.**

**Слова могут объединяться в более сложные конструкции: предложения, правильное построение которых описывается синтаксисом. В свою очередь описание смысла предложений составляет семантику языка, т. с. указание на то, как каждое из них выполнять компьютеру. Трансляция программы с одного языка на другой в общем случае состоит в изменении ее алфавита, лексики и синтаксиса, но с сохранением семантики. Трансляция обычно проходит в несколько этапов. На первом из них производится лексический анализ, приводящий входную программу к стандартному виду (редактирование) и переводу ее на внутренний язык, в котором все слова имеют одинаковый формат (длину). Одновременно с этим выполняется лексический контроль, выявляющий недопустимые слова. Содержанием второго этапа является синтаксический анализ, цель которого — распознавание типа предложений и структуры программы, а также синтаксический контроль, выявляющий соответствующие ошибки. Третий этап состоит в семантическом анализе каждого предложения и генерации объектного кода, т. е. исполняемой программы.**

**Лексический анализ представляет собой первую фазу компиляции. Его основная задача состоит в чтении новых символов и выдачи последовательности лексем, используемых синтаксическим анализатором в своей работе.**

# Лексический анализатор

Лексема (лексическая единица языка) — это структурная единица языка, которая состоит из элементарных символов языка и не содержит в своем составе других структурных единиц языка. Лексемами языков естественного общения являются слова. Лексемами языков программирования являются идентификаторы, константы, ключевые слова языка, знаки операций и разделители. Состав возможных лексем каждого конкретного языка программирования определяется синтаксисом этого языка.

Лексический анализатор (или сканер) — это часть компилятора, которая читает исходную программу и выделяет в ее тексте лексемы входного языка. На вход лексического анализатора поступает текст исходной программы. Результатом работы лексического анализатора является перечень всех найденных в тексте исходной программы лексем. Этот перечень лексем можно представить в виде таблицы, называемой таблицей лексем. Каждой лексеме в таблице лексем соответствует некий уникальный условный код, зависящий от типа лексемы, и дополнительная служебная информация. Кроме того, информация о некоторых типах лексем, найденных в исходной программе, должна помещаться в таблицу идентификаторов (или в одну из таблиц идентификаторов, если компилятор предусматривает различные таблицы идентификаторов для различных типов лексем).

Таблица лексем фактически содержит весь текст исходной программы, обработанный лексическим анализатором. В нее входят все возможные типы лексем, кроме того, любая лексема может встречаться в ней любое количество раз. Таблица идентификаторов содержит только определенные типы лексем — идентификаторы и константы. Лексемы в таблице лексем обязательно располагаются в том же порядке, что и в исходной программе (порядок лексем в ней не меняется).

С теоретической точки зрения лексический анализатор не является обязательной частью компилятора. Все его функции могут выполняться на этапе синтаксического разбора, поскольку полностью регламентированы синтаксисом входного языка. Однако существует несколько причин, по которым в состав практически всех компиляторов включают лексический анализ:

* применение лексического анализатора сокращает объем информации, обрабатываемой на этапе синтаксического разбора;
* некоторые задачи, требующие использования сложных вычислительных методов на этапе синтаксического анализа, могут быть решены более простыми методами на этапе лексического анализа (например, задача различения унарного минуса и бинарной операции вычитания, обозначаемых одним и тем же знаком «–»);
* лексический анализатор отделяет сложный по конструкции синтаксический анализатор от работы непосредственно с текстом исходной программы, структура которого может варьироваться в зависимости от архитектуры вычислительной системы, где выполняется компиляция, — при такой конструкции компилятора для перехода на другую вычислительную систему достаточно только перестроить относительно простой лексический анализатор;
* в современных системах программирования лексический анализатор может выполнять обработку текста исходной программы параллельно с его подготовкой пользователем — это дает системе программирования принципиально новые возможности, которые позволяют снизить трудоемкость разработки программ.

Функции, выполняемые лексическим анализатором, и состав типов лексем, которые он выделяет в тексте исходной программы, могут меняться в зависимости от реализации компилятора. То, какие функции должен выполнять лексический анализатор, а какие оставлять для этапа синтаксического разбора, решают разработчики компилятора. Лексические анализаторы также выполняют исключение из текста исходной программы комментариев и незначащих символов (пробелов, символов табуляции и перевода строки). Лексический анализатор имеет дело с такими объектами, как различного рода константы и идентификаторы (к последним относятся и ключевые слова). Язык констант и идентификаторов является регулярным — то есть может быть описан с помощью регулярных грамматик. Распознавателями для регулярных языков являются конечные автоматы. Следовательно, основой для реализации лексических анализаторов служат регулярные грамматики и конечные автоматы. Конечный автомат для каждой входной цепочки языка дает ответ на вопрос о том, принадлежит или нет цепочка языку, заданному автоматом. Однако в общем случае задача лексического анализатора несколько шире, чем просто проверка цепочки символов лексемы на соответствие входному языку. Кроме этого, он должен выполнить следующие действия: определить границы лексем, которые в тексте исходной программы явно не указаны; выполнить действия для сохранения информации об обнаруженной лексеме (или выдать сообщение об ошибке, если лексема неверна).

# Синтаксический анализатор

Синтаксический анализатор (синтаксический разбор) - это часть компилятора, которая отвечает за выявление основных синтаксических конструкций входного языка. В задачу синтаксического анализа входит: найти и выделить основные синтаксические конструкции в тексте входной программы, установить тип и проверить правильность каждой синтаксической конструкции, наконец, представить синтаксические конструкции в виде, удобном для дальнейшей генерации текста результирующей программы. В основе синтаксического анализатора лежит распознаватель текста входной программы на основе грамматики входного языка. Как правило, синтаксические конструкции языков программирования могут быть описаны с помощью КС-грамматик, реже встречаются языки, которые могут быть описаны с помощью регулярных грамматик. Чаще всего регулярные грамматики применимы к языкам ассемблера, а языки высокого уровня построены на основе синтаксиса КС-языков. Распознаватель дает ответ на вопрос о том, принадлежит или нет цепочка входных символов заданному языку – это основная задача синтаксического анализатора. Кроме этого синтаксический анализатор должен иметь некий выходной язык, с помощью которого он передает следующим фазам компиляции всю информацию о найденных и разобранных синтаксических структурах. [4]

Синтаксический разбор — это основная часть компилятора на этапе анализа. Без выполнения синтаксического разбора работа компилятора бессмысленна, в то время как лексический разбор в принципе является необязательной фазой. Все задачи по проверке синтаксиса входного языка могут быть решены на этапе синтаксического разбора. Лексический анализатор только позволяет избавить сложный по структуре синтаксический анализатор от решения примитивных задач по выявлению и запоминанию лексем входной программы. Синтаксический анализатор воспринимает выход лексического анализатора и разбирает его в соответствии с грамматикой входного языка. Лексический и синтаксический анализаторы могут работать параллельно или поочерёдно.

1. Основная часть