Лабораторная работа №1
Лексический анализ

Цель работы: Изучение основных понятий теории регулярных грамматик, ознакомление с назначением и принципами работы лексических анализаторов, получение практических навыков построения сканера на примере заданного входного языка.

Теоретические сведения.

Лексема (лексическая единица языка) – это структурная единица языка, которая состоит из элементарных символов языка и не содержит в своем составе других структурных единиц языка. Состав возможных лексем определяется синтаксисом языка программирования. Обычно принято выделять следующие типы лексем: идентификаторы, константы, ключевые слова, ограничители (разделители).

Лексический анализатор – это часть компилятора, которая читает литеры программы и строит из них слова (лексемы) исходного языка. Лексический анализ включает в себя сканирование исходного текста программы, распознавание лексем и их классификацию. Кроме этого на этапе лексического анализа отбрасываются комментарии. Программу, которая выполняет лексический анализ, называют лексическим анализатором или сканером.

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. Сканер можно запрограммировать как отдельный проход, на котором выполняется полный лексический анализ исходной программы, и который выдаёт лексическому анализатору таблицу, содержащую исходную программу в форме внутреннего представления. С другой стороны, его можно запрограммировать в виде подпрограммы синтаксического анализатора, которая вызывается всякий раз, когда анализ предыдущего символа завершён и синтаксическому анализатору требуется новый символ. Последний вариант, вообще говоря лучше, поскольку не требует хранения внутреннего представления всей программы. Но мы будем использовать первый подход, т.к. он позволяет отодвинуть на более позднее время решение о выборе метода синтаксического анализа. В современных компиляторах используются оба этих подхода.

Лексический анализатор выделяет из текста лексемы различных типов: идентификаторы, литералы (числовые и символьные константы), разделители. Выделение (сборка) лексемы сопровождается проверкой её правильности. Обнаруженные лексические ошибки фиксируются. Язык описания лексических единиц в большинстве случаев является регулярным, то есть может быть описан с помощью регулярных грамматик. Распознавателями регулярных языков являются конечные автоматы. Одним из способов описания конечного автомата является графическое его представление в виде маркированного однонаправленного графа, в котором узлы соответствуют состояниям конечного автомата, дуги отображают

переходы из одного состояния в другое, а символы маркировки дуг соответствуют функции перехода конечного автомата.

Работа сканера заключается в моделировании различных конечных автоматов для распознавания идентификаторов, зарезервированных слов, констант и разделителей. Почти для каждого языка программирования можно выделить следующие классы литер:

– Литеры, которые могут появиться в символе «целое» (например цифры).

– Литеры, которые могут появиться в качестве первой литеры «идентификатора» (например буквы).

– Литеры, которые могут появиться «идентификатора» (обычно буквы или цифры).

– Литеры, которые сканер полностью игнорирует или исключает.

– Литеры, которые сигнализируют о конце формируемой лексемы, но которые во всех других случаях игнорируются (например пробел).

– Литеры-разделители, которые сами являются лексемами (например +).

Иногда бывает необходимо непересекающихся групп:

– Разделители, с которых не начинаются двулитерные разделители и ключевые слова (например ;).

– Разделители, с которых начинается по крайней мере одно ключевые слово, но ни один двулитерный разделитель (например точка в Fortran).

– Разделители, с которых начинается по крайней мере один двулитерный разделитель, но ни одно ключевые слово (например двоеточие).

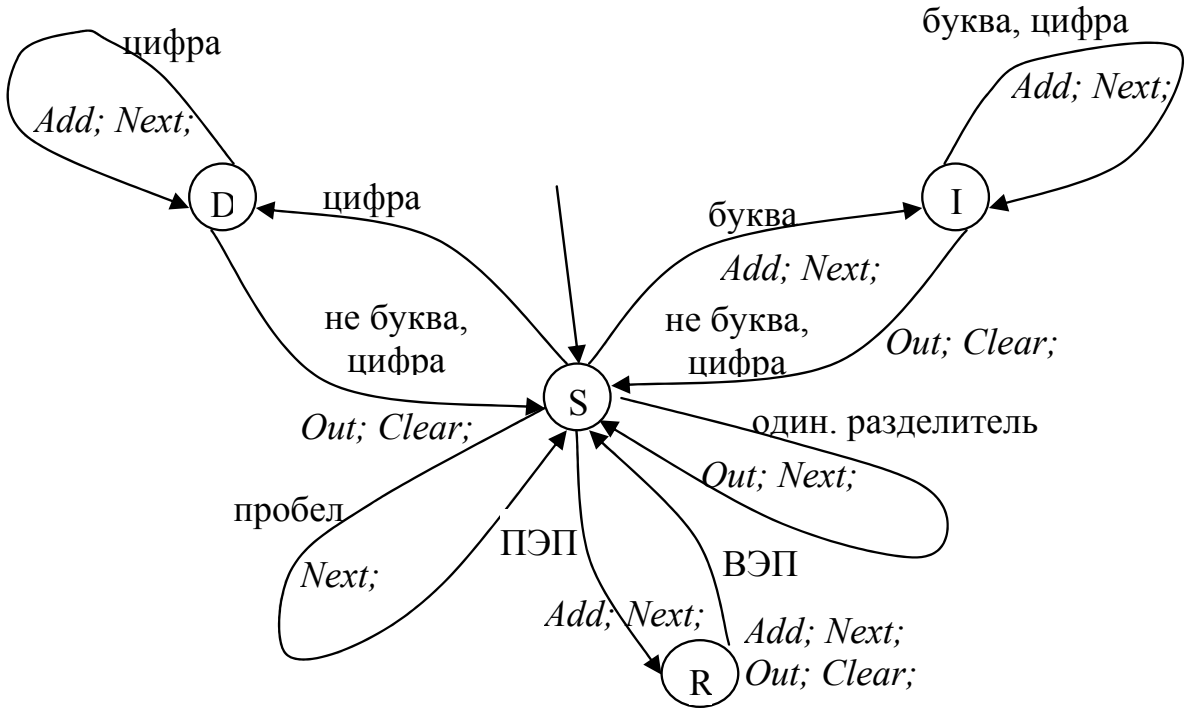

Рис. 1. Пример диаграммы состояний с действиями.

На рисунке 1 приведена диаграмма действий автомата, реализующего лексический анализ.

– Add – добавить очередной символ в конец буфера.

– Next – переместиться к следующему элементу входного потока.

– Out(лексема, тип) – выдать информацию о накопленной лексеме в выходной поток. Тип задаётся веткой диаграммы состояний по которой была собрана лексема.

– Clear – очистить буфер.

– ПЭП и ВЭП – первый и второй элемент пары соответственно.

– Все состояния диаграммы – конечные.

Алгоритм разбора цепочек символов по диаграмме состояний с действиями:

Входные данные лексического анализатора – текст транслируемой программы на входном языке.

Выходные данные – последовательность лексем (с указанием их предварительного типа).

1. Объявляем текущим начальное состояние S диаграммы.

2. До тех пор, пока не будет достигнуто конечное состояние диаграммы на

последнем элементе входного потока или состояние ERROR, считываем очередной символ анализируемой строки и переходим из текущего состояния диаграммы в другое по дуге, помеченной этим символом, выполняя при этом соответствующие действия. Состояние, в которое попадаем, становится текущим.

3. Выходной поток формируется вызовом подпрограммы out.

Результатом работы сканера является перечень всех найденных в тексте программы лексем. Его можно представить в виде таблицы пар (лексема, её предварительный тип) или списка. Тип лексемы будем уточнять в следующей лабораторной работе.

Таблица лексем фактически содержит весь текст исходной программы, разбитый на лексемы в том порядке, в каком они встречаются в программе. Если лексема встречается в программе несколько раз, то столько раз она войдёт и в таблицу лексем.

Пример 1. Описать результаты разбора на лексические единицы фрагмента программы.

В таблице 1 представлен результат обработки сканером следующего предложения языка Pascal:

```
for i := 10 to 100 do y := i + x;
```

Таблица 1. Таблица лексем.

Лексема	Предварительный тип
for	идентификатор
i	идентификатор
:=	разделитель
10	литерал
to	идентификатор
100	литерал
do	идентификатор

y	идентификатор
:=	разделитель
i	идентификатор
+	разделитель
x	идентификатор
;	разделитель

Лексический анализатор строится в три этапа:

– проанализировать терминальные и нетерминальные элементы языка, продумать структуру лексических единиц;

– построить диаграмму состояний с действиями для распознавания лексем;

– по полученной диаграмме с действиями написать программу сканирования текста исходной программы, синтеза лексем, их классификации, формирования внутреннего представления программы.

Задание на лабораторную работу:

Написать программу, которая выполняет лексический анализ входного текста, подготовленного в соответствии с заданием и порождает таблицу лексем с указанием их предварительных типов. Текст на входном языке задаётся в виде символьного (текстового) файла.

Программа должна выдавать сообщения о наличие во входном тексте ошибок, которые могут быть обнаружены на этапе лексического анализа.

Длину идентификаторов и строковых констант ограничить 8 символами.

Содержание отчета

- Титульный лист
- Текст задания
- Программный код реализации задания
- Скриншоты работы программы

Вопросы для самоконтроля

1. Отличие транслятора от компилятора
2. Основные этапы работы транслятора
3. Назначение лексического анализа
4. Что такое лексема
5. Отличие ключевых слов от имен переменных с точки зрения лексического анализатора

Список литературы

1. Шульга, Т. Э. Теория автоматов и формальных языков : учебное пособие / Т. Э. Шульга. — Саратов : Саратовский государственный технический университет имени Ю.А. Гагарина, ЭБС АСВ, 2015. — 104 с. — ISBN 987-5-7433-2968-7. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/76519.html (дата обращения: 15.04.2021). — Режим доступа: для

авторизир. пользователей. - DOI: https://doi.org/10.23682/76519 - https://www.iprbookshop.ru/76519.html

2. Алымова, Е. В. Конечные автоматы и формальные языки : учебник / Е. В. Алымова, В. М. Деундяк, А. М. Пеленицын. — Ростов-на-Дону, Таганрог : Издательство Южного федерального университета, 2018. — 292 с. — ISBN 978-5-9275-2397-9. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/87427.html (дата обращения: 15.04.2021). — Режим доступа: для авторизир. пользователей - https://www.iprbookshop.ru/87427.html

3. Пентус, А. Е. Математическая теория формальных языков : учебное пособие / А. Е. Пентус, М. Р. Пентус. — 3-е изд. — Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. — 218 с. — ISBN 978-5-4497-0662-1. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/97548.html (дата обращения: 15.04.2021). — Режим доступа: для авторизир. пользователей - https://www.iprbookshop.ru/97548.html

4. Миронов, С. В. Формальные языки и грамматики : учебное пособие для студентов факультета компьютерных наук и информационных технологий / С. В. Миронов. — Саратов : Издательство Саратовского университета, 2019. — 80 с. — ISBN 978-5-292-04613-4. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/99047.html (дата обращения: 15.04.2021). — Режим доступа: для авторизир. пользователей - https://www.iprbookshop.ru/99047.html

5. Малявко, А. А. Формальные языки и компиляторы : учебник / А. А. Малявко. — Новосибирск : Новосибирский государственный технический университет, 2014. — 431 с. — ISBN 978-5-7782-2318-9. — Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. — URL: https://www.iprbookshop.ru/47725.html (дата обращения: 15.04.2021). — Режим доступа: для авторизир. пользователей - https://www.iprbookshop.ru/47725.html