ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра ТК

курс лекций по дисциплине

Системное программное обеспечение

Тема: Лексические анализаторы

Преподаватель: к.т.н., доцент Карамзина А.Г.

Тема № 11

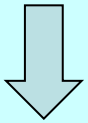
Лексические анализаторы

- Назначение лексического анализатора
- Основные функции ЛА
- Организация взаимосвязи лексического и синтаксического анализа
- Принципы построения лексических анализаторов
- Обобщенный алгоритм работы простейшего ЛА в компиляторе

Назначение лексического анализатора

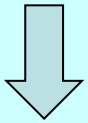
Лексема (*лексическая единица языка*) – это структурная единица языка, которая состоит из элементарных символов языка и не содержит в своем составе других структурных единиц языка.

Язык естественного общения



слова

Язык программирования



идентификаторы,
константы,
ключевые слова языка,
знаки операций и т. п.

Лексический анализатор (*сканер*) – это часть компилятора, которая читает исходную программу и выделяет в ее тексте лексемы входного языка.

Состав возможных лексем каждого конкретного языка программирования определяется синтаксисом этого языка.

На вход лексического анализатора (ЛА) поступает текст исходной программы, а выходная информация передается для дальнейшей обработки компилятором на этапе синтаксического анализа и разбора.

Назначение лексического анализатора

С теоретической точки зрения ЛА не является обязательной частью компилятора. Все его функции могут выполняться на этапе синтаксического анализа, поскольку полностью регламентированы синтаксисом входного языка.

Тем не менее, в состав практически всех компиляторов включают лексический анализ по следующим причинам:

- применение ЛА упрощает работу с текстом исходной программы на этапе синтаксического разбора и сокращает объем обрабатываемой информации;
- для выделения в тексте
эффективную и теоретичес
время как на этапе синта
используются достаточно сл

При такой конструкции компилятора для перехода от одной версии языка к другой достаточно только перестроить относительно простой лексический анализатор

- ЛА отделяет сложный по конструкции синтаксический анализатор от работы непосредственно с текстом исходной программы, структура которого может варьироваться в зависимости от версии входного языка.

Основные функции ЛА:

- исключить из текста исходной программы комментарии, незначащие пробелы, символы табуляции, перевода строки;

- выделить лексемы следующих типов: идентификаторы, строковые, символьные и числовые константы, ключевые (*служебные*) слова входного языка, знаки операций и разделителей.

- четко определить границы лексемы, которые в исходном тексте явно не заданы;

- выполнить действия для сохранения информации об обнаруженной лексеме *(или выдать сообщение об ошибке, если лексема неверна*).

Организация взаимосвязи лексического и синтаксического анализа

- *последовательная* – ЛА просматривает весь текст исходной программы от начала до конца и преобразует его в структурированный набор данных (***таблицу лексем***), в которой ключевые слова языка, идентификаторы и константы, как правило, заменяются на специально оговоренные коды, им соответствующие. Для идентификаторов и констант, кроме того, устанавливается связь между *таблицей лексем* и *таблицей идентификаторов*, которая заполняется параллельно.

В этом варианте реализации ЛА
Просматривает весь текст исходной программы
один раз от начала до конца.
Таблица лексем строится полностью вся сразу,
и больше к ней компилятор не возвращается.
Всю дальнейшую обработку выполняют
следующие фазы компиляции.

- *параллельная* – лексический анализ исходного текста выполняется поэтапно так, что синтаксический анализатор, выполнив разбор очередной конструкции языка, обращается к ЛА за следующей лексемой. При этом он может сообщить информацию о том, какую лексему следует ожидать.

В этом варианте реализации
при возникновении ошибки может
происходить «откат назад»
(для попытки выполнения анализа
текста на другой основе).
В случае успешного выполнения разбора
очередной конструкции языка
синтаксическим анализатором,
лексический анализатор
помещает найденные лексемы
в таблицу лексем
и таблицу идентификаторов и анализ
продолжается дальше в том же порядке.

Пример:

```
…
begin
        if a>b then a:=a-b
                else a:=b*0.3;
end;
…
```

Лексема	Тип лексемы	Значение
begin	Ключевое слово	A1
if	Ключевое слово	A2
a	Идентификатор	*a*:1
>	Знак операции сравнения	>
b	Идентификатор	*b*:2
then	Ключевое слово	A3
a	Идентификатор	*a*:1
:=	Знак присваивания	:=
a	Идентификатор	*a*:1
-	Знак арифметической операции	-
b	Идентификатор	*b*:2
else	Ключевое слово	A4
a	Идентификатор	*a*:1
:=	Знак присваивания	:=
b	Идентификатор	*b*:2
*	Знак арифметической операции	*
0.3	Вещественная константа	0.3
;	Разделитель	;
end	Ключевое слово	A5
;	Разделитель	;

Принципы построения лексических анализаторов

Лексический анализатор имеет дело с такими объектами, как различного рода константы и идентификаторы (*к последним относятся и ключевые слова*).

Язык констант и идентификаторов в большинстве случаев является регулярным, то есть может быть описан с помощью *регулярных грамматик*.

Распознавателями для регулярных языков являются *конечные автоматы* (КА).

Существуют правила, с помощью которых для любой регулярной грамматики может быть построен недетерминированный конечный автомат, распознающий цепочки языка, заданного этой грамматикой.

КА для каждой входной цепочки языка дает ответ на вопрос о том, принадлежит или нет цепочка языку, заданному автоматом.

Принципы построения лексических анализаторов

Поскольку во входном тексте программы лексемы не ограничены никакими специальными символами, то ***выделение границ лексем*** представляет определенную проблему (*в терминах программы-сканера определение границ лексем есть выделение тех строк в общем потоке входных символов, для которых надо выполнять распознавание*).

Для большинства входных языков границы лексем распознаются по заданным терминальным символам (*пробелы, знаки операций, символы комментариев, разделители (запятые, точки с запятой и т. п.)*). Набор таких терминальных символов может варьироваться в зависимости от синтаксиса входного языка.

Лексический анализатор действует по следующему принципу:

очередной символ из входного потока данных добавляется в лексему всегда, когда он может быть туда добавлен.

Как только символ не может быть добавлен в лексему, то считается, что он является границей лексемы и началом следующей лексемы

(*если символ не является пустым разделителем – пробелом, символом табуляции или перевода строки, знаком комментария*).

Принципы построения лексических анализаторов

Лексический анализатор работает *прямо,* если для данного входного текста (*цепочки символов входного языка*) и текущего положения указателя в нем он определяет лексему, расположенную непосредственно справа от указателя, и сдвигает сам указатель вправо от части текста, образующей эту лексему. При *прямой работе* ЛА возможно его *последовательное* взаимодействие с синтаксическим распознавателем.

Лексический анализатор работает *непрямо,* если для данного входного текста (*цепочки символов входного языка*), заданного типа лексемы и текущего положения указателя в нем он определяет лексему, расположенную непосредственно справа от указателя, и если она соответствует требуемому типу, то сдвигает указатель вправо от части текста, образующей эту лексему. В отличие от прямого варианта данному ЛА на входе требуется *задавать* также и *тип* ожидаемой *лексемы*. Поэтому при *непрямой работе* ЛА требуется его *параллельное* взаимодействие с синтаксическим распознавателем.

Обобщенный алгоритм работы простейшего ЛА в компиляторе

- из входного потока выбирается очередной символ, в зависимости от которого запускается тот или иной ЛА (*символ может быть также проигнорирован либо признан ошибочным*);

- запущенный ЛА просматривает входной поток символов программы на исходном языке, выделяя символы, входящие в требуемую лексему (*до обнаружения очередного символа, который может ограничивать лексему, либо до обнаружения ошибочного символа*);

- при успешном распознавании информация о выделенной лексеме заносится в таблицу лексем и ТИ, алгоритм возвращается к первому этапу и продолжает рассматривать входной поток символов с того места, на котором остановился ЛА;

- при неуспешном распознавании выдается сообщение об ошибке, а дальнейшие действия зависят от реализации ЛА – либо его выполнение прекращается, либо делается попытка распознать следующую лексему (*идет возврат к первому этапу алгоритма*).