ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Северо-Кавказский филиал
ордена Трудового Красного Знамени федерального государственного бюджетного образовательного учреждения высшего образования
"Московский технический университет связи и информатики"

Методические указания
для проведения лабораторной работы №6

по дисциплине

«СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ»

по теме

«Исследование реализации синтаксического анализатора»

Направление подготовки:
09.03.01 Информатика и вычислительная техника

Профили **Программное обеспечение и интеллектуальные системы**
Вычислительные машины, комплексы, системы и сети

Ростов-на-Дону
2019

УДК 681.3.06 (076)
ББК 32.07

Чикалов А.Н. Системное программное обеспечение. Исследование реализации синтаксического анализатора. Методические указания для проведения лабораторной работы №6. Ростов-на-Дону: Северо-Кавказский филиал МТУСИ, 2019.- 24 с.

В пособии изложены методические рекомендации и содержательные материалы для проведения занятий изучению системных ресурсов и алгоритмов работы файловой системы современных ОС, а также утилит для изучения, контроля и восстановления данных в файловой системе. Кроме того рассматриваются механизмы управления файловой системой программными средствами с помощью интерфейса прикладного программирования.

Пособие содержит необходимые справочные материалы.

Методические указания предназначены для студентов, обучающихся по направлению подготовки 09.03.01 Информатика и вычислительная техника, профиля Вычислительные машины, комплексы, системы и сети, Программное обеспечение и интеллектуальные системы.

Пособие предназначено для использования при изучении дисциплин Системное программное обеспечение, а также может быть использовано преподавателями и студентами при изучении родственных дисциплин и в процессе самостоятельной работы.

Учебное пособие обсуждено и одобрено на заседании кафедры ИВТ
Протокол №1 от 26.08.2019 г.

Рецензент Зав. кафедрой ИВТ д.т.н. профессор Соколов С.В.

Лабораторная работа 6. Исследование реализации синтаксического анализатора.

ЦЕЛЬ РАБОТЫ

Изучить табличные методы синтаксического анализа. Получить представление о методах диагностики и исправления синтаксических ошибок. Научиться проектировать синтаксический анализатор на основе табличных методов.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Разбор или синтаксический анализ включает группирование токенов исходной программы в грамматические фразы, используемые компилятором для синтеза вывода. Обычно грамматические фразы исходной программы представляются в виде дерева. Иерархическая структура программы выражается рекурсивными правилами. Например, если в языке определены только операции ‘+’ и ‘*’, при определении выражения можно придерживаться следующих правил:

1. любой идентификатор есть выражение;
2. любое число есть выражение;
3. если e_1 и e_2 – выражения, то выражениями являются e_1+e_2, e_1*e_2, (e_1).

Точно так же многие языки программирования рекурсивно определяют инструкции языка правилами, аналогичными следующим:

1. Если i_1 – идентификатор, e_2 – выражение, то $i_1 = e_2$ есть инструкция.
2. Если e_1 – выражение, а st_2 – инструкция, то while (e_1) st_2; if (e_1) st_2; являются инструкциями.

Дерево разбора наглядно показывает, как начальный символ грамматики порождает строку языка. Формально для контекстно-свободной грамматики дерево разбора представляет собой структуру со следующими свойствами:

1. Корень дерева помечен начальным символом.
2. Каждый лист помечен терминальным символом грамматики.
3. Каждый внутренний узел представляет нетерминальный символ.
4. Если A является нетерминальным символом и помечает некоторый внутренний узел, а X_1, X_2, ... X_n – отметки его дочерних узлов, перечисленные слева направо, то $A \rightarrow X_1 X_2 \ldots X_n$ – продукция (правило вывода). Здесь X_1, X_2, ... X_n могут представлять собой как терминальные, так и нетерминальные символы.

5. Все листья дерева, прочитанные слева направо, образуют файл токенов.

По результатам разбора для каждого оператора исходной программы можно построить синтаксическое дерево, удовлетворяющее следующим требованиям:

1. Ключевые слова и знаки операций являются корнями непустых поддеревьев.
2. Идентификаторы и константы являются листьями.

Концевой (постфиксный) обход синтаксического дерева позволяет получить постфиксную (обратную польскую) запись.

Пример. Построим дерево разбора, синтаксическое дерево и постфиксную запись для оператора “i=j+2;”. Порождающая грамматика содержит правила:

S→id=s1;

s1→s2+s1

s1→s2

s2→id

s2→const

id→ идентификатор

const→ константа

Терминалами являются “идентификатор”, “константа”, “+” , “=” и “;”, представленные соответствующими токенами. Начальный символ грамматики – S. Результаты проектирования представлены на рис.8 и рис.9.

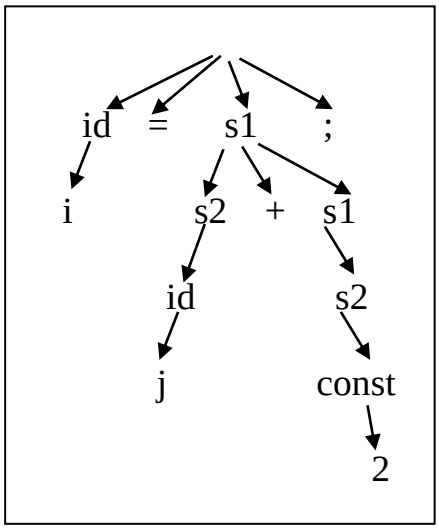

Рисунок 8. Дерево разбора.

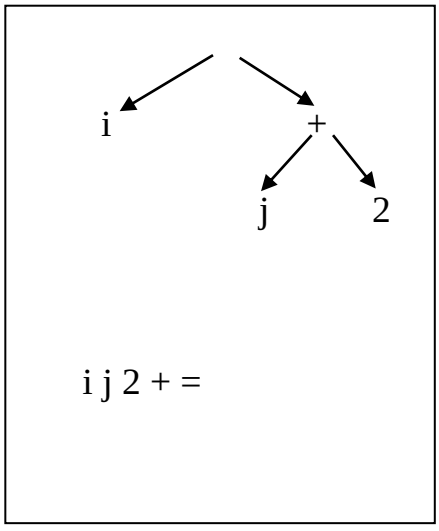

Рисунок 9. Синтаксическое дерево и постфиксная запись.

Семантический анализ имеет целью проверку правильности описания типов объектов программы и корректное их использование в инструкциях. При семантическом анализе используются иерархические структуры, полученные во время синтаксического анализа, для идентификации операторов и операндов выражений и конструкций. Важным ас-

пектом семантического анализа является проверка операторов на использование операндов допустимого спецификациями языка типов. Например, определение многих языков программирования требует, чтобы при использовании действительного числа в качестве индекса элемента массива генерировалось сообщение об ошибке. В то же время спецификация языка может позволить определенное насильственное преобразование типов, например, когда бинарный арифметический оператор применяется к операндам целого и действительного типов. В этом случае компилятору может потребоваться преобразование целого числа в действительное. При использовании метода операторного предшествования создаются семантические подпрограммы, учитывающие атрибуты лексем, с целью устранения неоднозначности разбора.

ЗАДАНИЕ

В соответствии с выбранным вариантом заданий к лабораторным работам реализовать синтаксический анализатор **с использованием одного из табличных методов** (*LL*-, *LR*-метод, метод предшествования).

Этапы проектирования синтаксического анализатора:

1. Сконструировать КС-грамматику в соответствии с вариантом задания.
2. В случае несоответствия построенной грамматики требованиям выбранного табличного метода разбора следует провести эквивалентные преобразования грамматики либо выбрать другой метод разбора.
3. Построить таблицу разбора и запрограммировать драйвер, реализующий работу с этой таблицей.

Исходные данные – файл токенов, таблицы лексем.

Результатом работы синтаксического анализатора является:

- синтаксическое дерево или постфиксная запись;
- файл сообщений об ошибках. В лабораторной работе необходимо реализовать возможности табличного метода по диагностике и исправлению синтаксических ошибок в исходной программе.

СОДЕРЖАНИЕ ОТЧЕТА

- Цели и задачи проекта;
- Вид, структура входных и выходных данных;
- Выбор стратегии разбора;
- Классификация грамматики входного языка, определение необходимости преобразования грамматики к требуемому виду, при необходимости выполнить эти преобразования;
- Схема разбора;
- Таблица разбора;
- Тексты программы анализатора на языке высокого уровня;
- Тестовые примеры.

Контрольные вопросы (ПК-11):

1. Какие задачи выполняют синтаксический анализ?
2. Что лежит в основе работы синтаксического анализатора?
3. Как формулируется формальная задача синтаксического анализа?
4. Каков формат входных данные для синтаксического анализатора?
5. Поясните структуру и форматы данных для построения выходных таблиц лексического анализатора?
6. Какова формальная задача синтаксического разбора (анализа)?
7. Каким образом выполняется преобразование дерева разбора в дерево операций?
8. Какие конструкции языка программирования использованы для реализации синтаксического анализатора?
9. Поясните логику работы операторов, использованных при написании программы.