Тип работы: Курсовая работа

Предмет: Информатика основы

СОДЕРЖАНИЕ

Введение

Компилятор - это программа, которая осуществляет перевод исходной программы в эквивалентную ей объектную программу на языке машинных команд или языке ассемблере.

Несмотря на более чем полувековую историю вычислительной техники, формально годом рождения теории компиляторов можно считать 1957, когда появился первый компилятор языка Фортран, созданный Бэкусом и дающий достаточно эффективный объектный код. До этого времени создание компиляторов было весьма творческим процессом. Лишь появление теории формальных языков и строгих математических моделей позволило перейти от творчества к науке. Именно благодаря этому, стало возможным появление сотен новых языков программирования.

Несмотря на то, что к настоящему времени разработаны тысячи различных языков и их компиляторов, процесс создания новых приложений в этой области не прекращается. Это связно как с развитием технологии производства вычислительных систем, так и с необходимостью решения все более сложных прикладных задач. Такая разработка может быть обусловлена различными причинами, в частности, функциональными ограничениями, отсутствием локализации, низкой эффективностью существующих компиляторов. Поэтому, основы теории языков и формальных грамматик, а также практические методы разработки компиляторов лежат в фундаменте инженерного образования по информатике и вычислительной технике.

Цель курсовой работы:

- закрепление теоретических знаний в области теории формальных языков, грамматик, автоматов и методов трансляции;
- формирование практических умений и навыков разработки собственного компилятора модельного языка программирования;
- закрепление практических навыков самостоятельного решения инженерных задач, развитие творческих способностей студентов и умений пользоваться технической, нормативной и справочной литературой.

В настоящее время в мире появляются более новые языки программирования и не каждый из ныне существующих трансляторов могут прочитать программы, написанный на новом языке, и перевести его в другой язык. Поэтому сейчас разрабатываются новые трансляторы, в этом и заключается актуальность данной курсовой работы.

Задачами выполнения курсовой работы являются:

 разработка генератора таблицы идентификаторов;
 разработка лексического анализатора;
 разработка синтаксического разборщика;
 разработка генератора результирующего кода;
 формирование умений применять теоретические знания при решении практических задач;
 подготовка к практической профессиональной деятельности;
 формирование культуры написания выпускной квалификационной работы.

1. Исходные данные для выполнения курсовой работы

Краткое изложение цели работы и задание по курсовой работе:

- разработка генератора таблицы идентификаторов;
- разработка лексического анализатора;
- разработка синтаксического разборщика;
- разработка генератора результирующего кода;
- формирование умений применять теоретические знания при решении практических задач;
- подготовка к практической профессиональной деятельности;
- формирование культуры написания выпускной квалификационной работы.

№

Тип констант

Оператор цикла или условия

Тип данных
13
2
y3
Char

В данном варианте, компилятор должен быть с двоичным типом констант, тип условных операторов: ifвыражение> thenоператор>;

2. Разработка лексического анализатора
Лексический анализатор (ЛА) – это первый этап процесса компиляции, на котором символы, составляющие исходную программу, группируются в отдельные минимальные единицы текста, несущие смысловую нагрузку – лексемы.
Задача лексического анализа - выделить лексемы и преобразовать их к виду, удобному для последующей обработки. ЛА использует регулярные грамматики.
ЛА необязательный этап компиляции, но желательный по следующим причинам:
1) замена идентификаторов, констант, ограничителей и служебных слов лексемами делает программу более удобной для дальнейшей обработки;
2) ЛА уменьшает длину программы, устраняя из ее исходного представления несущественные пробелы и комментарии;
3) если будет изменена кодировка в исходном представлении программы, то это отразится только на ЛА.
В процедурных языках лексемы обычно делятся на классы:
1) операторы;
2) разделители;
3) числа;
4) идентификаторы.
Каждая лексема представляет собой пару чисел вида (n, k), где n – номер таблицы лексем, k - номер лексемы в таблице.
Входные данные ЛА - текст транслируемой программы на входном языке.
Выходные данные ЛА - списковая структура, содержащая лексемы в числовом представлении.
Для модельного языка таблица операторов(1):
0. dim 1. end 2. % 3. ! 4. $ 5. read 6. write 7. for 8. to 9. Do 10. while 11. if 12. then 13. else 14. true 15. false
Таблица разделителей (2):
0. > 1. = 2. 3. =
4. > 5. >= 6. + 7. –
8. or 9. / 10. not 11.
12. * 13. : 14. ass 15. And
16. (17.) 18. . 19. { 20. }
Таблицы идентификаторов (4) и таблица чисел(3) формируются в ходе лексического анализа.
Опишем результаты работы лексического анализатора для модельного языка М.
Входные данные ЛА:
dim A,I %:
dim Min %:
Min ass 32767:
for I ass 1 to 10 do
read (A)
if A Min then Min ass A: {poiskminimalnogo}
write(Min):
end

Список используемой литературы
1. Антонов А.С. Параллельное программирование с использованием OpenMP: учебное пособие. – М.: Издательство МГУ, 2009. – 77 с.
2. Бьерн Страуструп. Программирование: принципы и практика использования C++: учебное пособие / исправленное издание. – М.: Вильямс, 2011. – 1248 с.

3. Корн Г. Алгебра матриц и матричное исчисление: Справочник по математике / Корн Г., Корн Т. – 4-е издание. – М.: Наука, 1978-804 с.
4. Скотт Мейерс. Эффективный и современный C++: 42 рекомендации по использованию C++12 и C++14: учебное пособие. – М.: Вильямс, 2016. – 304 с.
5. Стивен Прата. Язык программирования C++ (C++11). Ленции и упражнения: учебное пособие. – М.: Вильямс, 2012. – 1248 с.
6. Стенли Б., Липпман, Жози Лажойе, Барбара Э. Язык программирования C++. Базовый курс: учебное пособие. – М.: Вильямс, 2014. – 1120 с.