|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ | | | | | | | | |  |
|  | | | |  | | | |  | | |
|  | | | Пермский государственный национальный исследовательский университет | | | | | |  | |
|  | | | |  | | Кафедра математического обеспечения  вычислительных систем | | | | |
|  | | **Отчёт**  по дисциплине  «Формальные грамматики и методы трансляции» | | | | | | |  | |
|  | | | |  | | | |  | | |
| Работу выполнил:  студент группы ПМИ-1  Вахрушев Андрей  «\_\_\_» \_\_\_\_\_\_\_\_ 2021 | | | | |  | | Проверил:  ассистент кафедры МОВС  Пономарев Филипп  Александрович  «\_\_\_» \_\_\_\_\_\_\_\_ 2021 | | | |
|  | | | |  | | | |  | | |
|  | | | | Пермь 2021 | | | |  | | |

# Задача

Глобальное задание: написать компилятор для подмножества языка Паскаль.

Задание разбивается на отдельные этапы:

1. Модуль ввода-вывода (8 баллов, оценивается совместно с лексическим анализатором).
2. Лексический анализатор (12 баллов, оценивается совместно с модулем ввода-вывода).
3. Синтаксический анализатор (12 баллов) с нейтрализацией синтаксических ошибок (8 баллов).
4. Семантический анализатор с нейтрализацией семантических ошибок (20 баллов).
5. Генерация кода (25 баллов).

*Общая минимальная часть.* Основные разделы программы: раздел описания переменных, раздел операторов. Переменные стандартных типов (Boolean, integer, real, char). Числовые константы. Арифметическое выражение (в выражении допустимы только константы, переменные, операции +, –, \*, / и скобки). Оператор присваивания и составной оператор.

Все исходники программы можно посмотреть на https://github.com/Daizman/CompilV2 .

# Теория

Компилятор – это программа, которая переводит программу на языке высокого уровня в эквивалентную программу на другом (объектном) языке. Обычно компилятор выдаёт листинг, содержащий текст исходной программы и сообщения обо всех обнаруженных ошибках. Структура компилятора представлена на рисунке 1.

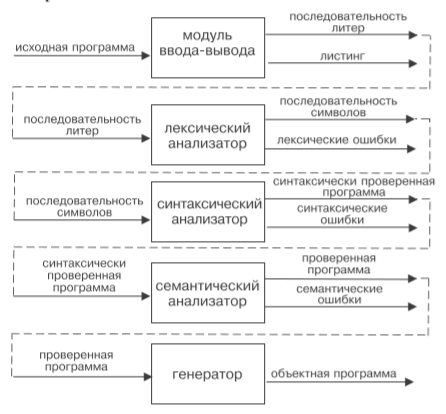


Рисунок 1 – структура компилятор.

В данной работе генератор не был реализован.

# Модуль ввода-вывода

Модуль ввода-вывода считывает последовательность литер исходной программы с внешнего устройства и передаёт их лексическому анализатору. Лексический анализатор, в свою очередь, обрабатывает переданную последовательность литер, и, в случае необходимости, формирует сообщения об ошибках.

Лексический анализатор, может запросить следующую литеру, либо отправить сообщение об ошибках.

Отдельно существуют классы для таких сущностей, как ошибка, идентификатор, оператор и значение.

Ошибка инкапсулирует в себе информацию о внештатных ситуациях. В ней хранятся: номер строки, номер символа в строке, ее описание, а так же код.

Идентификатор хранит в себе сущность, которая нужна для обозначения других сущностей в программе, заданных пользователем.

Оператор нужен для разделения синтаксиса и зарезервированных слов языка от идентификаторов.

Значения инкапсулируют данные, с которыми работает пользователь: строки, числа, символы, числа с плавающей точкой.

С модулем ввода-вывода могут взаимодействовать все для вывода ошибок.

# Лексический анализатор

Его задача сводится к предоставлению компилятору токенов.

Токен - является единицей языка, оператором, ключевым словом, значением или идентификатором. Лексический анализатор преобразует входящий в него поток литер в исходящий поток токенов.

Токеном может быть: значение, идентификатор, оператор.

Значение может быть четырех типов: целое, строковое, числа с плавающей точкой и символьное.

Операторы: ключевые слова или символы, которые используются в выражениях языка, например, точка с запятой.

Идентификатором является всё, что не является оператором и начинается с буквы, и состоит из букв, и опционально цифр.

Анализатор выявляет максимально возможное количество ошибок за один просмотр исходной программы. Информация об ошибках заносится в таблицу ошибок, каждый элемент которой обязательно содержит код и местоположение ошибки. Анализатор запоминает информацию об ошибках в таблице ошибок в результате обращения к функции.

# Синтаксический анализатор

Синтаксический анализатор проверяет, удовлетворяет ли программа формальным правилам.

Синтаксический анализатор выполняет синтаксический анализ последовательности символов. На этом этапе символы рассматриваются как неделимые, и их представление как последовательности литер несущественно. Анализатор, правильно анализирует программы, которые удовлетворяют формальным правилам описания языка. Фактически такой анализатор работает верно только до первой ошибки в исходной программе. Чтобы просканировать всю программу анализатор дополняют нейтрализацией. Основная идея нейтрализации ошибок, которую мы будем рассматривать, состоит в следующем: после выявления ошибки надо пропустить один или несколько символов, чтобы найти символ, начиная с которого можно возобновить анализ.

Все правила, описывающие синтаксис языка, имеют вид:

<S> ::= L (1)

где S — обозначение синтаксической конструкции, а L может состоять из символов языка, синтаксических конструкций и метасимволов «|» и «{ }».

# Семантический анализатор

Суть семантического анализатора состоит в том, чтобы выяснить, не нарушены ли неформальные правила описания языка.

Семантический анализатор - модуль компилятора, выполняющий семантический анализ.

Информация о существующих идентификаторах хранится в таблице идентификаторов.

Таблица типов (ТТ) создается для каждой области действия. Когда начинается анализ области действия, соответствующая ТТ пуста. По мере обработки объявлений типов дескрипторы типов добавляются в ТТ.