2. Разработка компилятора

На данном этапе работы необходимо в соответствии с требованиями, указанными в техническом задании разработать структуру компилятора, а также разработать соответствующий анализатор для каждого из этапов компиляции.

2.1. Разработка структуры компилятора

Для обеспечения трансляции исходного кода в исполняемый модуль разработана структура компилятора, представляющая собой диаграмму прецедентов. Диаграмма прецедентов представлена на  рисунке 1.

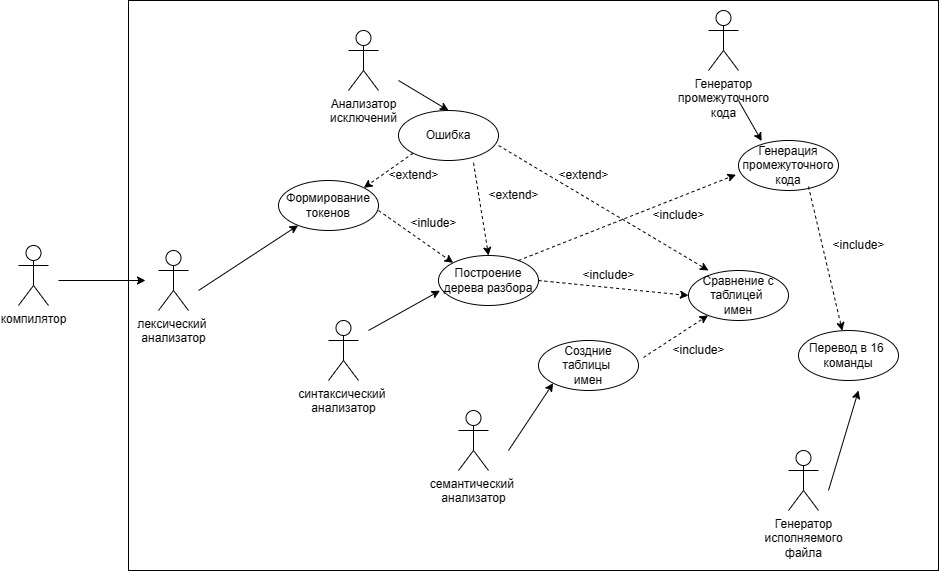


Рисунок 1 – Диаграмма прецедентов

Инициализатором трансляции исходного кода является компилятор, передающий управление лексическому анализатору.

В свою очередь лексический анализатор выполняет формирование токенов за счет выделения слова из текста программы, определения его типа и записи этих данных в структуру. Во время выполнения может возникнуть ошибка при некорректно введенном слове.

Действие «ошибка» инициализирует анализатор исключений.

Синтаксический анализатор может выполнять действие построения дерева разбора, но только в том случае, если до этого было произведено разбиение текста программа на массив токенов. Построение дерева выполняется по разработанной грамматике. Следовательно, при несоответствии анализируемого токена с тем, который задан грамматикой выполняется формирование ошибки.

Семантический анализатор инициализирует действие сравнения с таблицей имен, которое включает в себя создание таблицы имен и построение дерева разбора.

Генерация промежуточного кода выполняет формирование промежуточного кода, которое включает в себя этап синтаксического и лексического анализатора.

Генератор промежуточного может выполнять действие перевода команды в 16-чный формат. Данное действие включает в себя предыдущие этапы компиляции.