Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ имени Н.Э.БАУМАНА
(национальный исследовательский университет)»

Факультет: Информатика и системы управления

Кафедра: Теоретическая информатика и компьютерные технологии

Лабораторная работа № 7.2

Самоприменимый генератор компиляторов
на основе предсказывающего анализа

Вариант 1 (Scala)

Работу выполнил

студент группы ИУ9-62

Чугунов Д. С.

Москва, 2019

Цель работы

Целью данной работы является изучение алгоритма построения таблиц предсказывающего анализатора.

Исходные данные

В данной лабораторной работе требуется разработать простейший генератор компиляторов, который по описанию грамматики рабочего языка, записанному на входном языке, строит таблицы предсказывающего анализа в виде составного литерала на целевом языке.

Здесь *рабочий язык* – это некоторый формальный язык с LL(1)-грамматикой, синтаксический анализатор которого должен быть построен генератором компиляторов.

В качестве *входного языка* генератора компиляторов должен выступать язык представления правил грамматики, лексику и синтаксис которого можно восстановить по листингу 1.

```
non-terminal E, E1, T, T1, F;
terminal '+', '*', '(', ')', n;

E  ::= T E1;
E1 ::= '+' T E1 | epsilon;
T  ::= F T1;
T1 ::= '*' F T1 | epsilon;
F  ::= n | '(' E ')';

axiom E;
```

Листинг 1 – пример языка алгебраических выражений описанный на данном языке описания грамматик

И, наконец, *целевым языком* мы будем называть язык реализации компилятора рабочего языка. Так как разрабатываемый генератор

компиляторов должен быть самоприменимым, целевой язык должен совпадать с языком реализации генератора компиляторов.

Задание

Лабораторная работа делается на основе выполненной лабораторной работы 7.1. Выполнение данной лабораторной работы состоит из следующих этапов:

1. Переписывание грамматики входного языка на самом входном языке.
2. Добавление в программу, написанную на лабораторной работе 7.1, генератора таблицы разбора на основе дерева разбора. Таблица разбора должна представлять собой инициализированный двумерный массив на рабочем языке (он совпадает с языком реализации генератора компиляторов).
3. Тестирование генератора компиляторов путём написания простейшего калькулятора арифметических выражений на основе грамматики, описанной в листинге 1.
4. Раскрутка генератора компиляторов путём подачи на вход грамматики входного языка, написанной на самом входном языке (пункт 1) и замены таблицы разбора, написанной вручную, на сгенерированную таблицу разбора.

Реализация

Грамматика входного языка на самом входном языке представлена на листинге 2.

На листинге 3 представлена реализация самоприменимого генератора компиляторов на основе предсказывающего анализа, реализованного на языке Scala.

```
  non-terminal …
  terminal …

  <пропущено>

  axiom S;                                          #
```

Листинг 2 – грамматика входного языка на самом входном языке

```
import scala.io.BufferedSource

object lab72 {
  …
  <пропущено>
  …
}
```

Листинг 3 – генератор компиляторов на основе предсказывающего анализа

Результаты

После раскрутки компилятора путём подачи на вход грамматики входного языка, написанной на самом входном языке, на выходе получилась таблица разбора идентичная таблице разбора, составленной самостоятельно. Сгенерированная таблица разбора представлена на листинге 4.

```
  val delta: Map[(String, String), List[(Int, String)]] = Map(
    …
  )
```

Листинг 4 – сгенерированная таблица разбора

Вывод

В ходе лабораторной работы были получены навыки написания самоприменимых генераторов компиляторов на основе предсказывающего анализа. Помимо этого, был разработан генератор компиляторов, способный по собственной грамматике вернуть таблицу разбора идентичную изначальной таблице разбора, другими словами самоприменимый генератор компиляторов.