|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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)» (МГТУ им. Н. Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**Лабораторная работа № 3**

**Дисциплина:** Конструирование компиляторов

**Студент:** Платонова Ольга

**Вариант:** 3

**Группа:** ИУ7-22M

**Преподаватель:** Ступников А. А.

Москва, 2023 г.

**Цель работы:** приобретение практических навыков реализации метода рекурсивного спуска для синтаксического разбора.

**Задачи работы:**

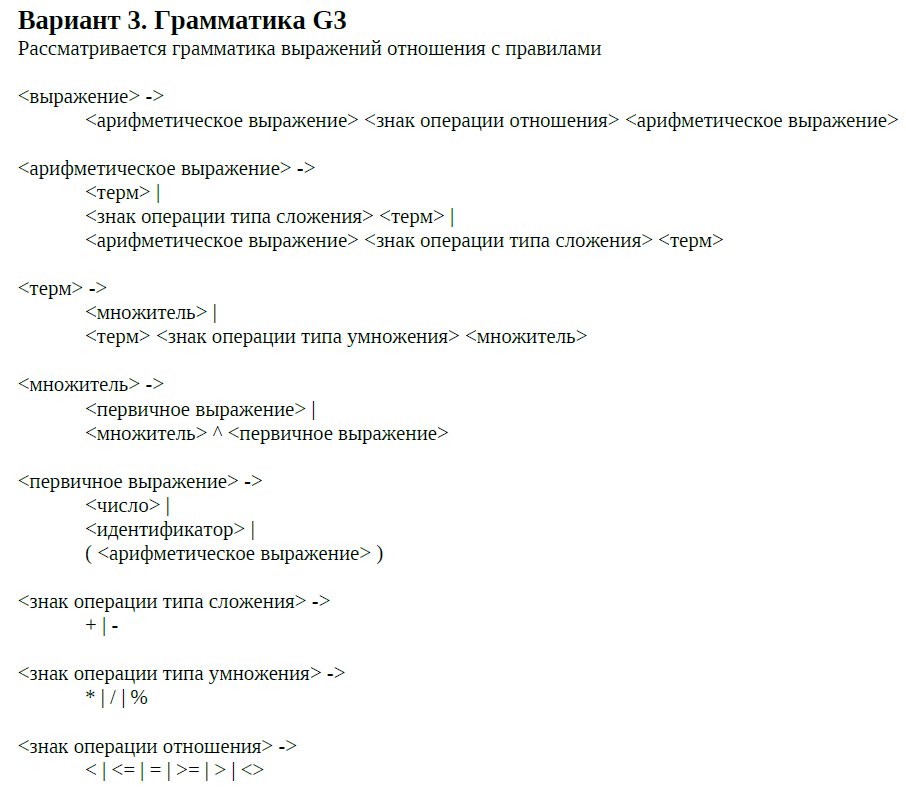
1. Принять к сведению соглашения об обозначениях, принятые в литературе по теории формальных языков и грамматик и кратко описанные в приложении.
2. Познакомиться с основными понятиями и определениями теории формальных языков и грамматик.
3. Детально разобраться в алгоритме метода рекурсивного спуска.
4. Разработать, тестировать и отладить программу нисходящего синтаксического анализа с использованием метода рекурсивного спуска в соответствии с предложенным вариантом.

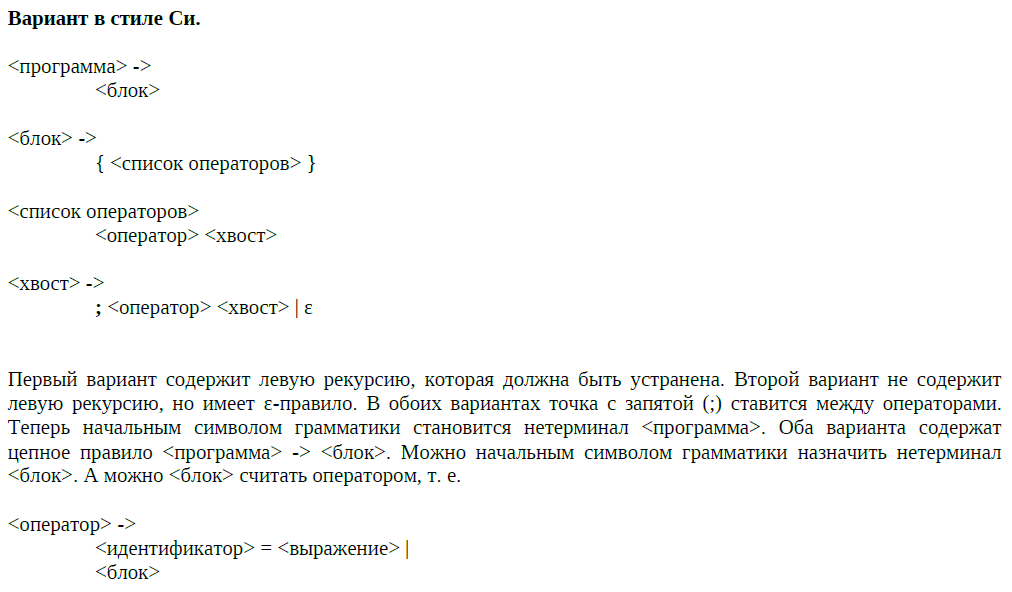
**Теоретическая часть**

Одним из наиболее простых и потому одним из наиболее популярных методов нисходящего синтаксического анализа является метод рекурсивного спуска (*recursive descent method*). Метод основан на «зашивании» правил грамматики непосредственно в управляющие конструкции распознавателя.

В методе рекурсивного спуска полностью сохраняются идеи нисходящего разбора, принятые в LL(1)-грамматиках:

* происходит последовательный просмотр входной строки слева-направо;
* очередной символ входной строки является основанием для выбора одной из правых частей правил группы при замене текущего нетерминала;
* терминальные символы входной строки и правой части правила «взаимно уничтожаются»;
* обнаружение нетерминала в правой части рекурсивно повторяет этот же процесс.





**Результаты работы**

{

a = 1;

b = a + 3;

c = 6 - b;

d = 3 <> c;

e = d / c;

}

Выражение валидно