|  |  |
| --- | --- |
| 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**

|  |  |
| --- | --- |
| **Тема Синтаксический разбор с использованием метода рекурсивного спуска**  **Вариант №3**  **Студент \_Коноваленко В.Д.\_\_\_\_\_\_\_\_**  **Группа \_ИУ7-21М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Ступников А.А.** |  |

Москва.

2024 г.

**Задание**

Рассматривается грамматика выражений отношения с правилами

<выражение> -> <арифметическое выражение> <знак операции отношения> <арифметическое выражение>

<арифметическое выражение> -> <терм> | <знак операции типа сложения> <терм> | <арифметическое выражение> <знак операции типа сложения> <терм>

<терм> -> <множитель> | <терм> <знак операции типа умножения> <множитель>

<множитель> -> <первичное выражение> | <множитель> ^ <первичное выражение>

<первичное выражение> -> <число> | <идентификатор> | ( <арифметическое выражение> )

<знак операции типа сложения> -> + | -

<знак операции типа умножения> -> \* | / | %

<знак операции отношения> -> < | <= | == | >= | > | <>

**Грамматика после удаления левой рекурсии:**

<выражение> -> <арифметическое выражение> <знак операции отношения> <арифметическое выражение>

<арифметическое выражение> -> <терм> <арифметическое выражение>' | <знак операции типа сложения> <терм> <арифметическое выражение>'

<терм> -> <множитель> <терм>'

<множитель> -> <первичное выражение> <множитель>'

<первичное выражение> -> <число> | <идентификатор> | ( <арифметическое выражение> )

<знак операции типа сложения> -> + | -

<знак операции типа умножения> -> \* | / | %

<знак операции отношения> -> < | <= | == | >= | > | <>

<арифметическое выражение>' -> <знак операции типа сложения> <терм> <арифметическое выражение>' | ϵ

<терм>' -> <знак операции типа умножения> <множитель> <терм>' | ϵ

<множитель>' -> ^ <первичное выражение> <множитель>' | ϵ

**Задание на лабораторную работу**

Дополнить грамматику блоком, состоящим из последовательности операторов присваивания.

**Вариант в стиле Си**

<программа> ->

<блок>

<блок> ->

{ <список операторов> }

<список операторов> ->

<оператор><хвост>

<хвост> ->

; <оператор><хвост> | ε

Точка с запятой (;) ставится между операторами. Теперь начальным символом грамматики становится нетерминал . Можно начальным символом грамматики назначить нетерминал. А можно <> считать оператором, т. е.

<оператор> ->

<идентификатор> = <выражение> |

<блок>

В последнем случае возможна конструкция с вложенными блоками. Для модифицированной грамматики написать программу нисходящего синтаксического анализа с использованием метода рекурсивного спуска.

**Итоговая грамматика**

<программа> -> <блок>

<блок> -> { <список операторов> }

<список операторов> -> <оператор> <хвост>

<оператор> -> <идентификатор> = <выражение> | { <список операторов> }

<хвост> -> ; <оператор> <хвост> | ε

<выражение> -> <арифметическое выражение> <знак операции отношения> <арифметическое выражение>

<арифметическое выражение> -> <терм> <арифметическое выражение>' | <знак операции типа сложения> <терм> <арифметическое выражение>'

<терм> -> <множитель> <терм>'

<множитель> -> <первичное выражение> <множитель>'

<первичное выражение> -> <число> | <идентификатор> | ( <арифметическое выражение> )

<знак операции типа сложения> -> + | -

<знак операции типа умножения> -> \* | / | %

<знак операции отношения> -> < | <= | == | >= | > | <>

<арифметическое выражение>' -> <знак операции типа сложения> <терм> <арифметическое выражение>' | ϵ

<терм>' -> <знак операции типа умножения> <множитель> <терм>' | ϵ

<множитель>' -> ^ <первичное выражение> <множитель>' | ϵ

**Пример построенного дерева**

Выражение: { i = 5 == 4 }

