МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра вычислительной техники



**ОТЧЁТ**

**по лабораторной работе №4**

***«*Синтаксис языков программирования. Нисходящий синтаксический анализ*»***

**по дисциплине: «** *Теория формальных языков и компиляторов***»**

Вариант: 24443144

Выполнила:Проверил:

Студентка гр. АВТ-709 *« к.т.н., доцент »*

*« Васильева Н.В. » « Малявко А.А. »*

«\_\_\_» \_\_\_\_\_\_ 2020г.«\_\_\_» \_\_\_\_\_\_ 2020 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (подпись)

Новосибирск

2020

Цели работы

Изучение основных идей и понятий нисходящих методов синтаксического анализа, выявление свойств формальных грамматик, необходимых для реализации нисходящего восстановления дерева грамматического разбора, приобретение навыков построения процедурной и различных автоматных реализаций нисходящего анализа, исследование поведения нисходящих синтаксических акцепторов.

**Вариант: 24443144**

1. *Идентификаторы*

|  |
| --- |
| **2** |
| <бБ><пЦ><бБ>  ( d23U, N1q, x15y, …) |

1. *Константы*

|  |
| --- |
| **4** |
| целые по основанию 10 и указанному в константе;  вещественные;  символьные |

1. *Объявления примитивных типов (целое, вещественное, символьное):*

|  |
| --- |
| **4** |
| **card[inal][\_u]**  **double**  **litera** |

1. *Оператор присваивания:*

|  |
| --- |
| **4** |
| **put**<В> **to** <И> ; |

1. *Условный оператор:*

|  |
| --- |
| **3** |
| **when** <ЛВ> **then** <ОБ> **[ other** <ОБ> **]** |

1. *Оператор цикла:*

|  |
| --- |
| **1** |
| **loop** <ОБ> **until** <ЛВ> |

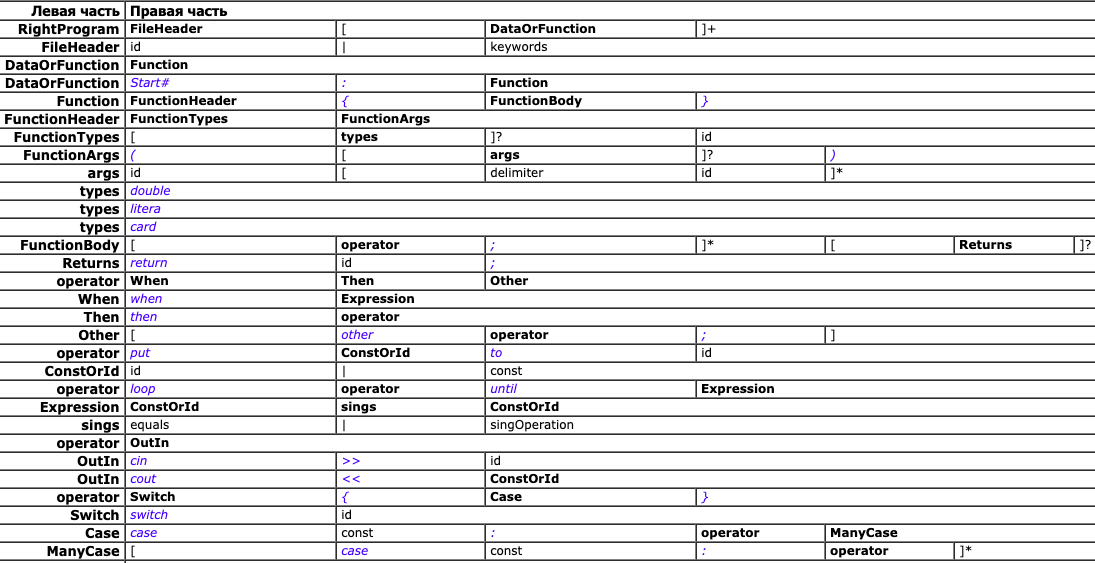
1. *Оператор переключателя*

|  |
| --- |
| **4** |
| **??** <В> **{ ?** <К> **:** <ОБ> **[gout;]** … **[ ?~** **:** <ОБ> **]** **}** |

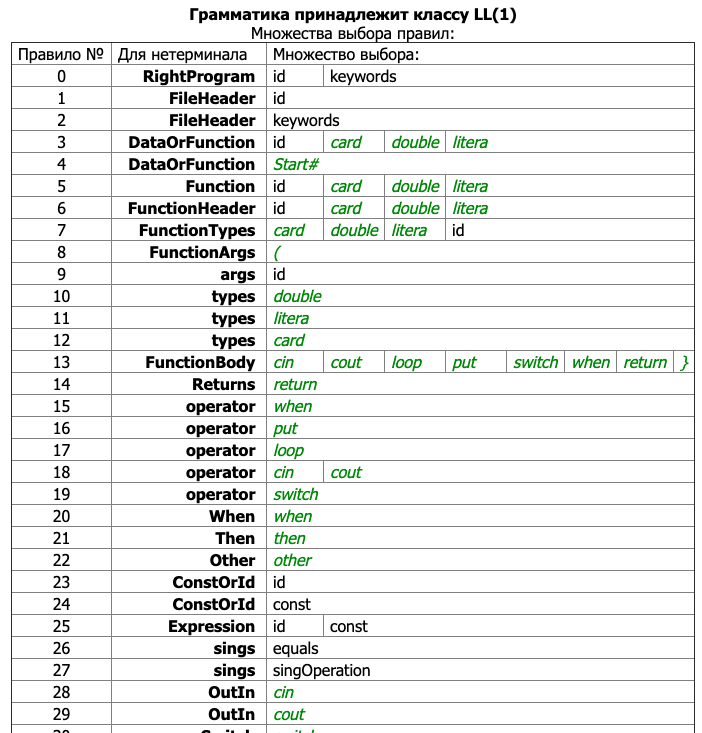
1. *Формат псевдокода*

|  |
| --- |
| **4** |
| Триады <Код> <ОП> <Р> |

# Реализация LL(1)-грамматики для языка, заданного на курсовую работу



*Рисунок 1. LL(1)-грамматики для языка, заданного на курсовую работу*



*Рисунок 2. Множество выбора правил*

# Управляющие таблицы автоматных реализаций нисходящего синтаксического акцептора



*Рисунок 3. Управ табл нисходящего автомата с несколькими состояниями*

Каждому символу каждого правила грамматики должно быть поставлено в соответствии в точности одно состояние автомата. С каждым состоянием должно быть связано множество выбора и два адреса перехода. Под адресом перехода понимается номер состояния.

При соблюдении определенных правил нумерации состояний и введении операции управления остановом по ошибке можно обойтись только одним адресом перехода.

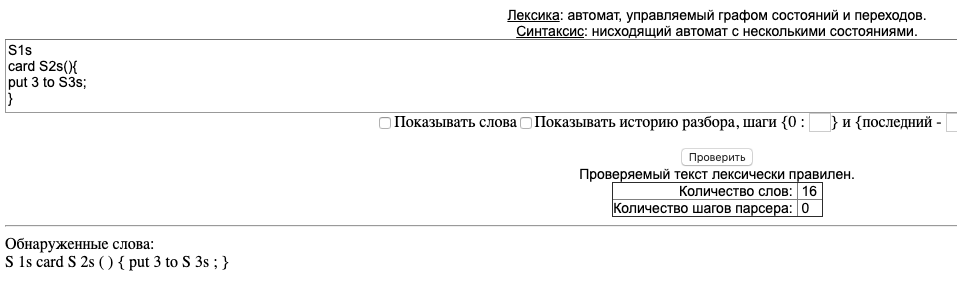
С каждым состоянием должны быть также связаны операции управления стековой памятью и чтением следующего входного символа.

– флаг a управляет чтением следующего входного символа;

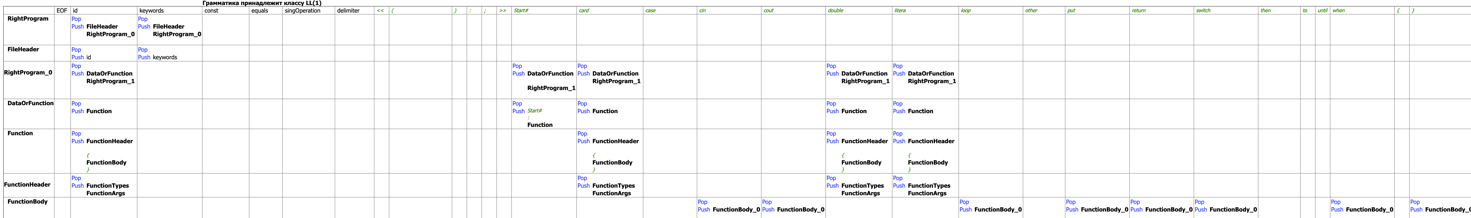
– флаг s управляет занесением адреса точки возврата в стек;

– флаг r обеспечивает переключение автомата в состояние, номер которого снимается с верхушки стека возвратов;

– флаг e запрещает останов по ошибке, когда состояние соответствует нетерминалу из левой части и есть еще хотя бы одно правило для такого нетерминала.



*Рисунок 4. Правильная программа (автомат с несколькими состояниями)*



*Рисунок 5. Управ табл нисходящего автомата с одним состоянием*

LL(1)-грамматику можно преобразовать в конечный автомат с единственным состоянием и стековой памятью, управляемый:

– текущим входным символом;

– символом, находящимся на верхушке стека.

Столбцы таблицы соответствуют входным символам, строки – символам, которые могут находиться в стеке, а в клетках указана некоторая последовательность операций над секом, входным потоком и состоянием автомата.

Обозначения (щелкните здесь, чтобы скрыть):

|  |  |
| --- | --- |
| Pop | Снять верхний символ со стека |
| Push | Поместить в стек цепочку символов, находящуюся в клетке справа от слова Push (верхний символ цепочки становится верхним символом стека) |
| Next | Прочитать следующий терминал со входа |
| Stop | Останов по окончанию восстановления дерева разбора правильного предложения |
| Пустая клетка | Останов по ошибке |

# /var/folders/qr/xrfhvh652dsbglqvj_6l70dm0000gn/T/TemporaryItems/(Документ сохраняется приложением «screencaptureui» 38)/Снимок экрана 2020-04-15 в 18.03.05.png

*Рисунок 6. Правильная программа (автомат с одном состоянием)*

# /var/folders/qr/xrfhvh652dsbglqvj_6l70dm0000gn/T/TemporaryItems/(Документ сохраняется приложением «screencaptureui» 39)/Снимок экрана 2020-04-15 в 18.06.18.png

*Рисунок 7. Неправильная программа (автомат с одном состоянием)*

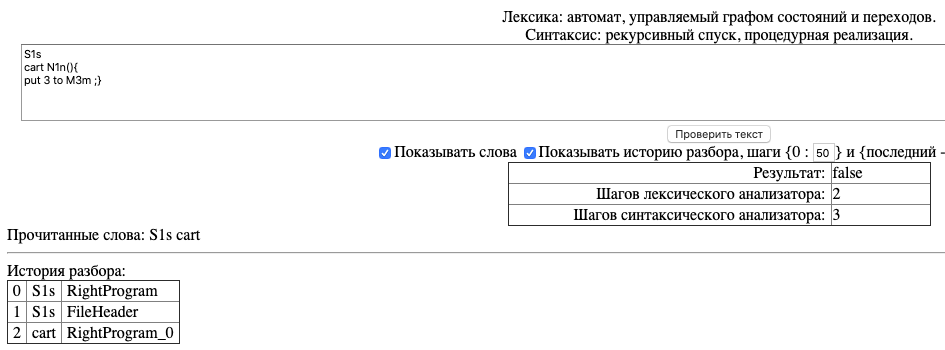
# Процедурная реализация рекурсивного списка

LL(1)-грамматика может быть преобразована в программу синтаксического акцептора, реализующую рекурсивный спуск. Такая реализация называется процедурной в отличии от автоматных (таблицы). Рекурсивный спуск может быть реализован только на таком языке программирования, который допускает рекурсивный вызов функции.

Собственно рекурсивный синтаксический акцептор – это функция RecurciveDescent(), возвращающая логическое значение (true (рис.8), если входное предложение правильное, и false(рис.9) – в противном случае).



*Рисунок 8. Правильный текст программы*



*Рисунок 9. Неправильный текст программы*

# Вывод

Изучили основные идеи и понятия нисходящих методов синтаксического анализа, выявление свойств формальных грамматик, необходимых для реализации нисходящего восстановления дерева грамматического разбора, приобретение навыков построения процедурной и различных автоматных реализаций нисходящего анализа, исследование поведения нисходящих синтаксических акцепторов.