F Московский авиационный институт

(национальный исследовательский университет)

Факультет «Прикладная математика и физика»

**Лабораторные работы по курсу**

**«Системное программное обеспечение»**

1. Спроектировать грамматику по заданному языку L

2. Спроектировать конечный автомат, составить диаграмму переходов КА и реализовать

3. Определить свойства КА. Построить НДКА. Реализовать преобразование НДКА в ДКА.

4. Устранить из КС-грамматики бесполезные символы и ε–правила

5 Устранить из KС-грамматики цепные правила и устранить левую рекурсию

6 Определить форму КС-грамматики и сделать ее приведение

7. Спроектировать МП автомат для приведенной КС-грамматики

8. Реализовать МП автомат для приведенной КС-грамматики

9. Для LL(1) анализатора построить управляющую таблицу M

10. Аналитически написать такты работы LL(1) анализатора для выведенной цепочки.

11. Реализовать управляющую таблицу M для LL(1) анализатора.

12. Построение LR(k) анализатора на основе активных префиксов и отношения OBLOW. Построить управляющую таблицу для функции перехода g(х) и действий f(u) и такты работы алгоритма.

13. Реализовать LR(k)-анализатор по управляющей таблице (g,f) для LR(k) грамматики.

14. Построение LR(k) анализатора на основе LR(0)-ситуаций и функций CLOSURE и GOTO. Построить управляющую таблицу для функции перехода g(х) и действий f(u).

15. Построить конечный автомат для переходов между ситуациями и такты работы алгоритма.

16. Реализовать LR(k)-анализатор по управляющей таблице (g,f) для LR(k) грамматики.

*Студент:* Ляшун Д.С.

*Группа:* 08-207Б-19

*Руководитель:* Семёнов А. С.

*Оценка:*

*Дата:*

**Москва 2021**

**1.Спроектировать грамматику по заданному языку L**:

1.1. Задан бесконечный язык L.

L={1(01)\*(01)ω1 | ω1{0,1}\*} = { {101(01)1}, {101(01)01}, {101(01)001}, {101(01)0001}, {101(01)00001}, … }

**Регулярность языка**

(1)Для всех регулярных языков ∀L⊆ ({1(01)\*(01)ω1 | ω1{0,1}\*} =>

(2) существует целое (∃p≥ 3 (минимальная длина цепочки символов) такое что

(3) для всех w = 101(01) ∈L((|101(01)| ≥ p=3) =>

(4) существует (∃x = 1, y = 01, z = (01) ∈ ∑\* такое что (w= 101(01)= xyz =>

1. (|y|=|01| ≥ 1, цикл y должен быть накачан хотя бы длиной 1 и

2. |xy|=|101|=3 ≤ p=3, цикл должен быть в пределах первых p=3 символов и

3. для i=1 ≥ 0, (=∈L)))))))), на x= и z=(01) ограничений не накладывается.

1.2. Преобразовать бесконечный язык L в язык L1 представляющий собой конечное множество цепочек, цепочки символов которого являются подмножество цепочек символов бесконечного языка. Ввести ограничения для языка, если они необходимы.

(1) = {{101(01)1}, {101(01)01}, {101(01)001}, {101(01)0001}, {101(01)00001}}

(2) = L(G) = (язык, порожденный грамматикой ~ заданному)

1.3. Сгенерировать цепочки символов по языку L1. Определить свойства языка L1.

Зададим грамматику , где

**T** = {0, 1, (,)} – конечное множество терминальных символов.

**V** = {} – конечное множество нетерминальных символов.

начальное состояние.

**P** – множество правил:

1.

2.

**Вывод цепочек:**

1.

3.

**2.Спроектировать конечный автомат, составить диаграмму переходов КА и реализовать**

2.1. Спроектировать грамматику для языка L1.

КА = ({0, 1, (, ) }, {}, , ,

1.

2.

3.

4.

5.

6.

7.

8.

9.

Пример конфигурации КА:

2.2. Построить диаграмму переходов и таблицу переходов по грамматике.

0

(

1

0

1

C

E

D

B

A

0

1

)

1

G

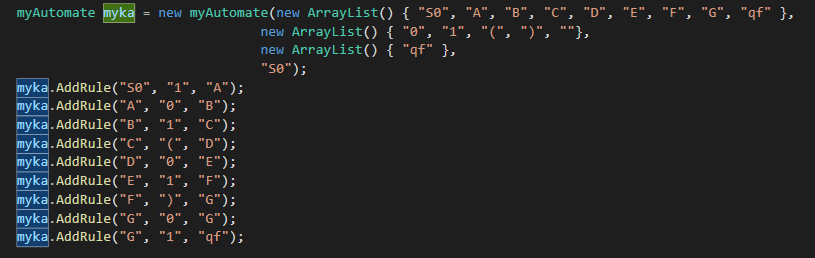
F

**Таблица переходов:**

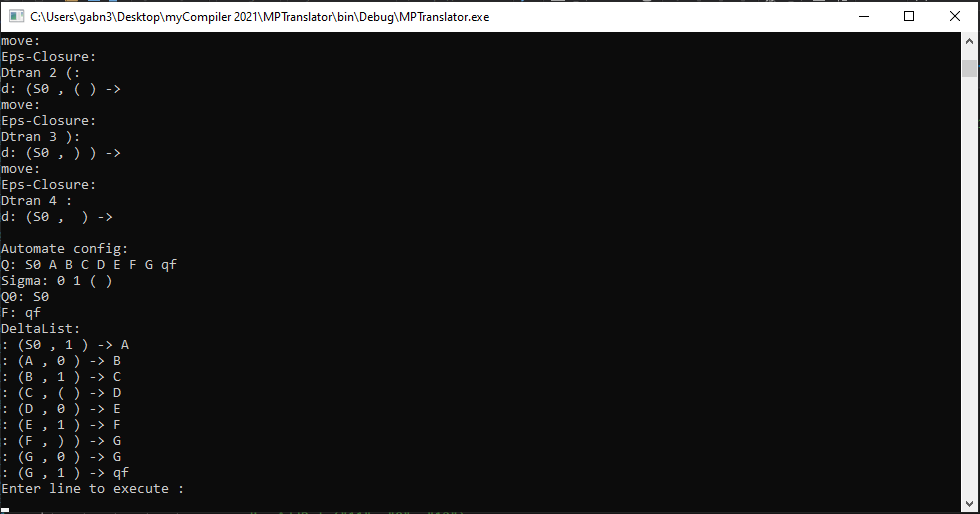
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Состояние** | **Входной символ** | | | |
| 0 | 1 | ( | ) |
|  |  | {A} |  |  |
| A | {B} |  |  |  |
| B |  | {C} |  |  |
| C |  |  | {D} |  |
| D | {E} |  |  |  |
| E |  | {F} |  |  |
| F |  |  |  | {G} |
| G | {G} | { |  |  |

**3.Определить свойства КА. Построить НДКА. Реализовать преобразование НДКА в ДКА.**

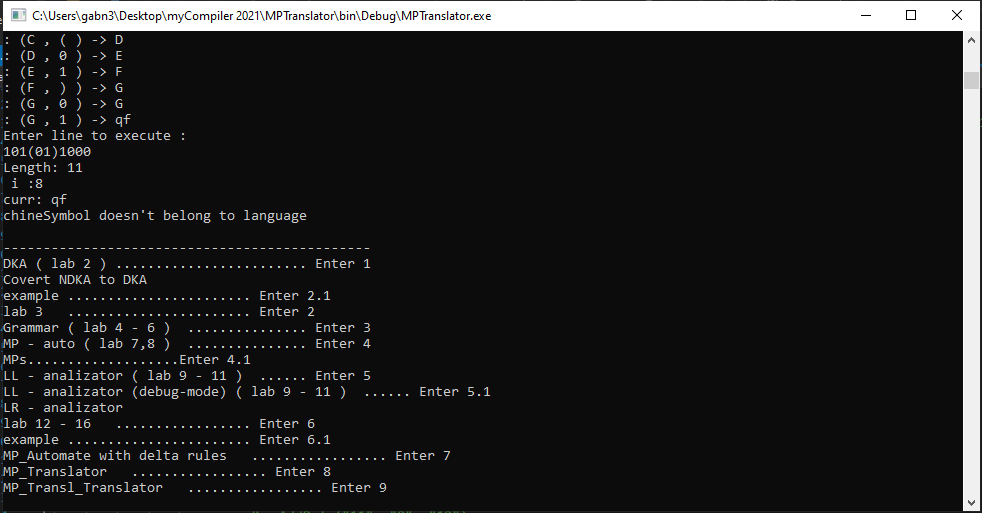
Написанный КА, представленный в исходном коде программы:



Результат работы алгоритма преобразования из НДКА в ДКА:



Пример распознавания КА цепочек символов: 



**4. Устранить из КС-грамматики бесполезные символы и ε–правила**

Задана КС-грамматика:

P = {S cFCB, A Ab, A cC, B cB, B , , F a, F ε}

**Вывод цепочек**

1. S => cFCB => cCB => cBB => cdB => cdd

2. S => cFCB => caCB => caBB => cadB => cadcB => cadcd

3. S => cFCB => cCB => cBB => cdB => cdcB => cdcd

4. S => cFCB => caCB => caBB => cadB => cadcB => cadccB => cadccd

5. S => cFCB => cCB => cBB => cdB => cdcB => cdccB => cdccd

**Алгоритм удаления бесполезных символов**

**Алгоритм удаления непроизводящих символов**

**Вход:**

**Выход:**

Шаг 1:

Шаг 2:

Шаг 3:

Шаг 4:

Шаг 5.

**Алгоритм удаления недостижимых символов**

**Вход:**

**Выход:**

Шаг 1:

Шаг 2:

Шаг 3:

Шаг 4:

**Алгоритм построения множества неукорачивающих нетерминалов**

**Вход:**

**Выход:**

Шаг 1:

Шаг 2:

Шаг 2:

**Алгоритм удаления эпсилон правил**

**Вход:**

**Выход:**

**Грамматика без бесполезных символов и эпсилон правил**

**Вывод цепочек**

1. S => cFCB => caCB => caBB => cadB => cadd

2. S => cFCB => caCB => caBB => cadB => cadcB => cadcd

3. S => cFCB => caCB => caBB => cadB => cadcB => cadccB => cadccd

4. S => cFCB => caCB => caBB => cacBB => cacdB => cacdd

**5. Устранить из KС-грамматики цепные правила и устранить левую рекурсию**

Задана КС-грамматика:

P = {S cFCB, B cB, B , , F a}

**Алгоритм удаления цепных правил**

**Вход:**

**Выход:**

Шаг 1: C

C

Шаг 2: больше цепных правил нет

P = {S cFCB, B cB, B , , F a}

**Алгоритм удаления левой рекурсии**

P = {S cFCB, B Bc, B , , F a}

**Вход:**

**Выход:**

Шаг 1: V = {B, S, F, C} = {

Шаг 2: i = 1. B

Шаг 3: i = 2, j = 1. Подходящих правил нет.

Шаг 4: i = 2. Подходящих правил нет.

Шаг 5. i = 3, j = 1. Подходящих правил нет.

Шаг 6. i = 3, j = 2. Подходящих правил нет.

Шаг 7. i = 3. Подходящих правил нет.

Шаг 8. i = 4, j = 1. Подходящих правил нет.

Шаг 9. I = 4, j = 2. Подходящих правил нет.

Шаг 10. I = 4, j = 3. Подходящих правил нет.

V’ = {S, F, C, B, B’}

P’ = {S cFCB, B Bc, B , B , , F a}

G = { {S, F, B, B’}, {a, c, d}, P’, S}

**Грамматика без цепных правил и левой рекурсии**

P = {S cFCB, B cB, B , , F a}

**Вывод цепочек**

1. S => cFCB => caCB => cdB => cadd

2. S => cFCB => caCB => cadB => cadcB => cadcd

3. S => cFCB => caCB => cadB => cadcB => cadccB => cadccd

4. S => cFCB => caCB => cacBB => cacdB => cacdd

**6. Определить форму КС-грамматики и сделать ее приведение**

**Исходная грамматика**

P = {S cFCB, A Ab, A cC, B cB, B , , F a, F ε}

**Вывод цепочек**

1. S => cFCB => cCB => cBB => cdB => cdd

2. S => cFCB => caCB => caBB => cadB => cadcB => cadcd

3. S => cFCB => cCB => cBB => cdB => cdcB => cdcd

4. S => cFCB => caCB => caBB => cadB => cadcB => cadccB => cadccd

5. S => cFCB => cCB => cBB => cdB => cdcB => cdccB => cdccd

**Приведённая грамматика**

P = {S cFCB, B cB, B , , F a}

Грамматика в нормальной форме Грейбах, так как в ней нет -правил и каждое правило имеет вид , где

Грамматика не в нормальной форме Хомского, так как имеет правила из терминалов и нетерминалов.

**Вывод цепочек**

1. S => cFCB => caCB => cdB => cadd

2. S => cFCB => caCB => cadB => cadcB => cadcd

3. S => cFCB => caCB => cadB => cadcB => cadccB => cadccd

4. S => cFCB => caCB => cacBB => cacdB => cacdd

**7. Спроектировать МП автомат для приведенной КС-грамматики**

**Приведённая грамматика**

P = {S cFCB, B cB, B , , F a}

**МП-автомат**

МП = ({q}, {a, c, d}, {a, c, d, S, F, C, B}, q, S, {q})

1.

2.

3.

4.

5.

6.

7.

**Распознавание цепочек**

1. (q, cadd, S) (q, cadd, cFCB) (q, add, FCB) (q, add, aCB)

(q, dd, CB) (q, dd, dB) (q, d, B) (q, d, d) (q,,)

2. (q, cadcd, S) (q, cadcd, cFCB) (q, adcd, FCB) (q, adcd, aCB)

(q, dcd, CB) (q, dcd, dB) (q, cd, B) (q, cd, cB) (q, d, B)

(q, d, d) (q,,)

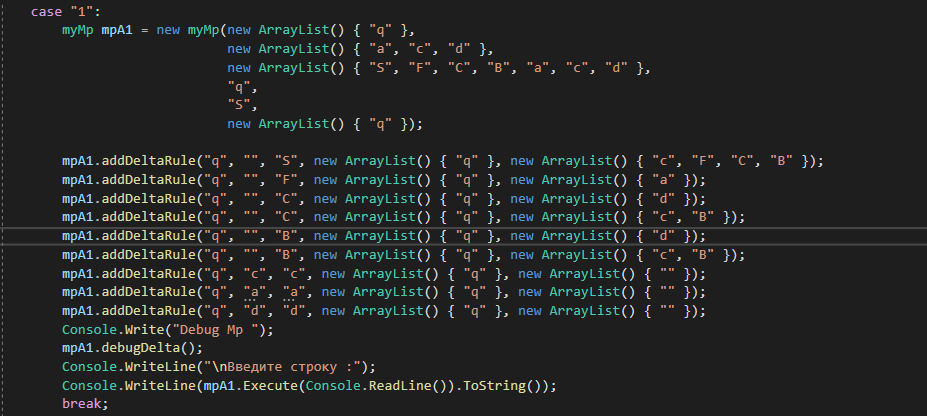
3. (q, cacdd, S) (q, cacdd, cFCB) (q, acdd, FCB) (q, acdd, aCB)

(q, cdd, CB) (q, cdd, cBB) (q, dd, BB) (q, dd, dB) (q, d, B)

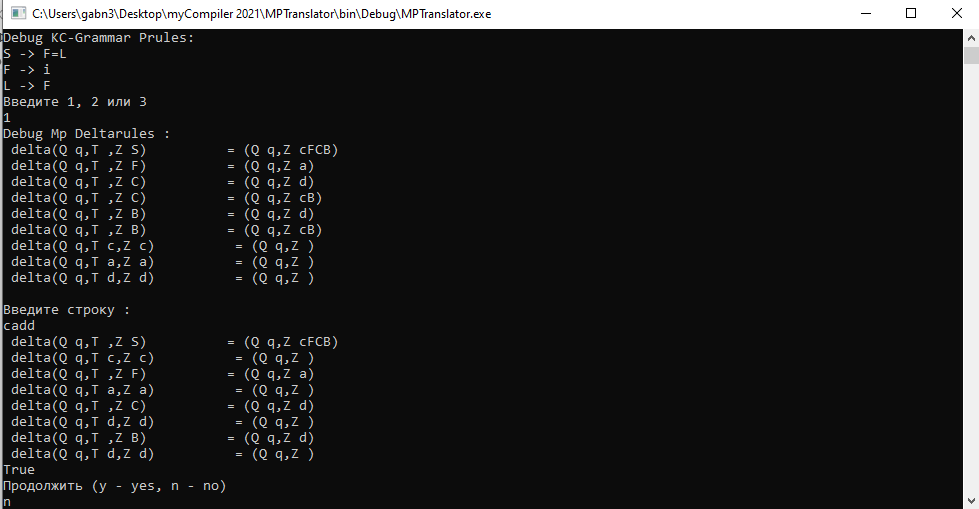
(q, d, d) (q,,)

**8. Реализовать МП автомат для приведенной КС-грамматики**

**МП-автомат**

****

**Распознавание цепочек**

****

**9. Для LL(1) анализатора построить управляющую таблицу M**

Последовательно рассмотреть правила и определить First (pi) и Follow(pk). Построить управляющую таблицу построчно.

Задана КС-грамматика G = (T, V, P, S), где

T = {i, j, \*, :,(,)},

V = {S, F, L},

P:

1. S F:L

2. S (L)

3. F L\*

4. F i

5. Lj

Грамматика принадлежит классу LL(1), так как для всех правил с альтернативами:

1. :

2. :

**Построение функции FIRST**

**Построение функции FOLLOW**

**Построение управляющей таблицы M**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **M** | **i** | **j** | **\*** | **:** | **(** | **)** |  |
| **S** | F:L, 1 | F:L, 1 |  |  | (L), 2 |  |  |
| **F** | i, 4 | L\*, 3 |  |  |  |  |  |
| **L** |  | j, 5 |  |  |  |  |  |
| **i** | ВЫБРОС |  |  |  |  |  |  |
| **j** |  | ВЫБРОС |  |  |  |  |  |
| **\*** |  |  | ВЫБРОС |  |  |  |  |
| **:** |  |  |  | ВЫБРОС |  |  |  |
| **(** |  |  |  |  | ВЫБРОС |  |  |
| **)** |  |  |  |  |  | ВЫБРОС |  |
|  |  |  |  |  |  |  | ДОПУСК |

**10. Аналитически написать такты работы LL(1) анализатора для выведенной цепочки.**

Рассмотрим работу алгоритма для цепочки символов порожденной LL(1) грамматикой.

G = ({i, j, \*, :,(,)}, {S, F, L}, P, S)

P:

1. S F:L

2. S (L)

3. F L\*

4. F i

5. Lj

**Вывод цепочек**

1.

2.

3.

**Распознавание цепочек**

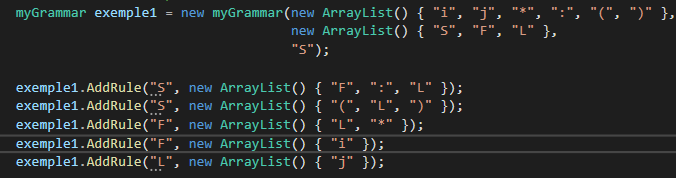
1.

2.

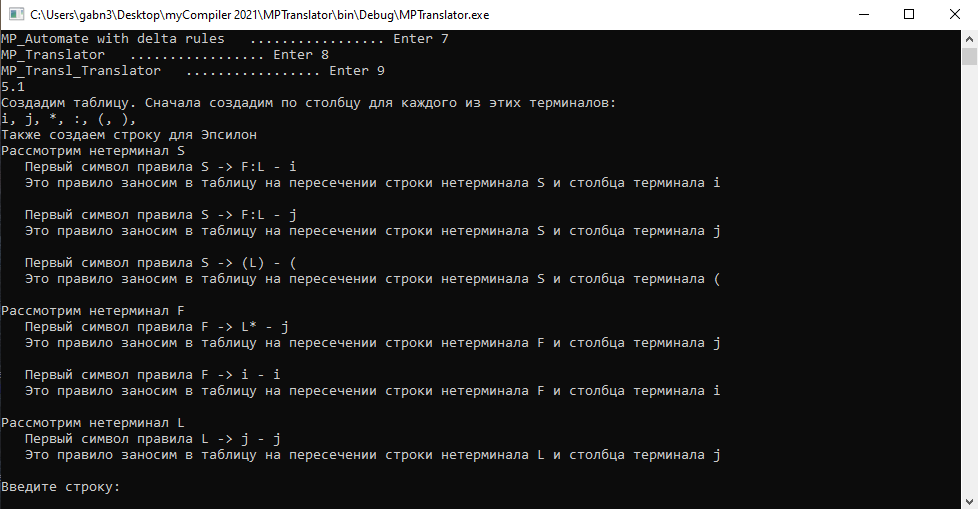
3.

**11. Реализовать управляющую таблицу M для LL(k) анализатора.**

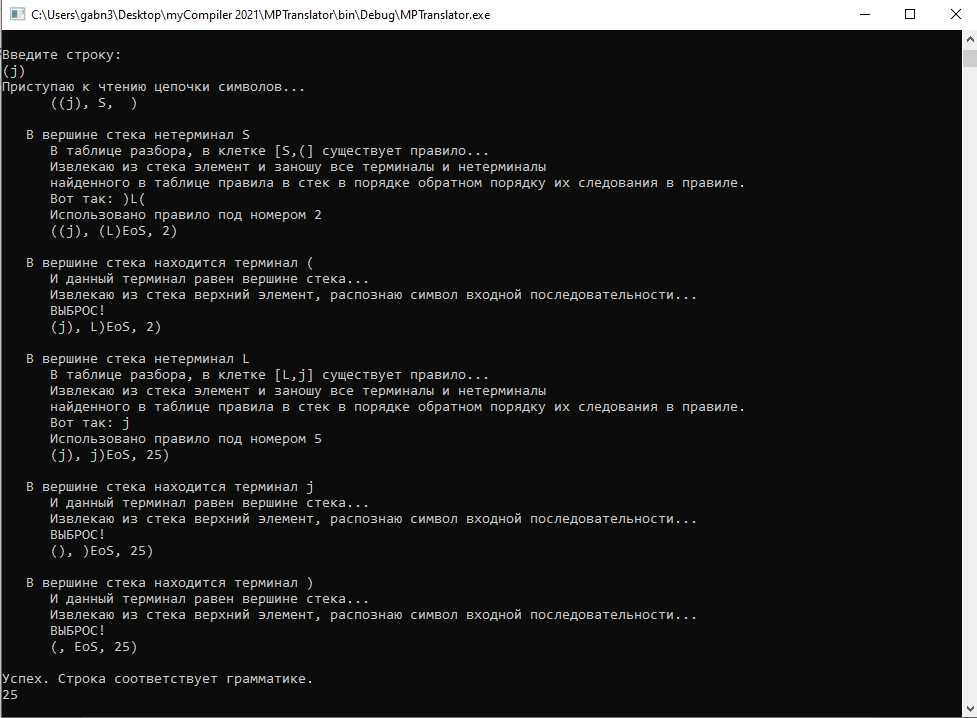
**Задание грамматики**

****

**Результат работы**

****

Построение управляющей таблицы M



Пример распознавания цепочки

**12. Построение LR(k) анализатора на основе активных префиксов и отношения OBLOW. Построить управляющую таблицу для функции перехода g(х) и действий f(u) и такты работы алгоритма.**

**Исходная грамматика**

G = ({i, j, \*, :,(,)},{S, F, L}, P, S)

P:

1. S F:L

2. S (L)

3. F L\*

4. F i

5. Lj

**Пополненная грамматика**

G = ({i, j, \*, :,(,)},{S, S’, F, L}, P, S)

P:

0. S’ S

1. S F:L

2. S (L)

3. F L\*

4. F i

5. Lj

**Определение активных префиксов**

|  |  |  |
| --- | --- | --- |
| Символ грамматики | Магазинный символ | Кодируемая цепочка |
| S |  | S |
| F |  | F |
| L |  | F:L |
|  | (L |
|  | L |
| i |  | i |
| j |  | j |
| \* |  | L\* |
| : |  | F: |
| ( |  | ( |
| ) |  | (L) |

**Матрица отношения OBLOW**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | 1 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  | 1 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | 1 |  |  |  | 1 |  |  |  |  |
|  |  |  |  | 1 |  |  | 1 |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  | 1 | 1 |  | 1 |  | 1 | 1 |  |  | 1 |  |

**Функция переходов g(x)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **g(X)** | **i** | **j** | **\*** | **:** | **(** | **)** | **S** | **F** | **L** |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**13. Реализовать LR(k)-анализатор по управляющей таблице (g,f) для LR(k) грамматики.**

**Управляющая таблица**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **функция действий f(u)** | | | | | | | **функция переходов g(X)** | | | | | | | | |
| **i** | **j** | **\*** | **:** | **(** | **)** |  | **i** | **j** | **\*** | **:** | **(** | **)** | **S** | **F** | **L** |
|  | П | П | П | П | П | П | Д |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |
|  | С(1) | С(1) | С(1) | С(1) | С(1) | С(1) | С(1) |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |
|  | С(4) | С(4) | С(4) | С(4) | С(4) | С(4) | С(4) |  |  |  |  |  |  |  |  |  |
|  | С(5) | С(5) | С(5) | С(5) | С(5) | С(5) | С(5) |  |  |  |  |  |  |  |  |  |
|  | С(3) | С(3) | С(3) | С(3) | С(3) | С(3) | С(3) |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |
|  | С(2) | С(2) | С(2) | С(2) | С(2) | С(2) | С(2) |  |  |  |  |  |  |  |  |  |
|  | П | П | П | П | П | П |  |  |  |  |  |  |  |  |  |  |

**Распознавание цепочек**

1.

2.

3.

**14. Построение LR(k) анализатора на основе LR(0)-ситуаций и функций CLOSURE и GOTO. Построить управляющую таблицу для функции перехода g(х) и действий f(u).**

**Замыкание множества ситуаций**

Шаг 1:

Шаг 2:

Шаг 3:

Шаг 4:

**Функция переходов GOTO**

уже присутствует как

уже присутствует как

**Каноническая форма множества ситуаций**

}

**Диаграмма переходов автомата**

**15. Построить конечный автомат для переходов между ситуациями и такты работы алгоритма.**

**Управляющая таблица**

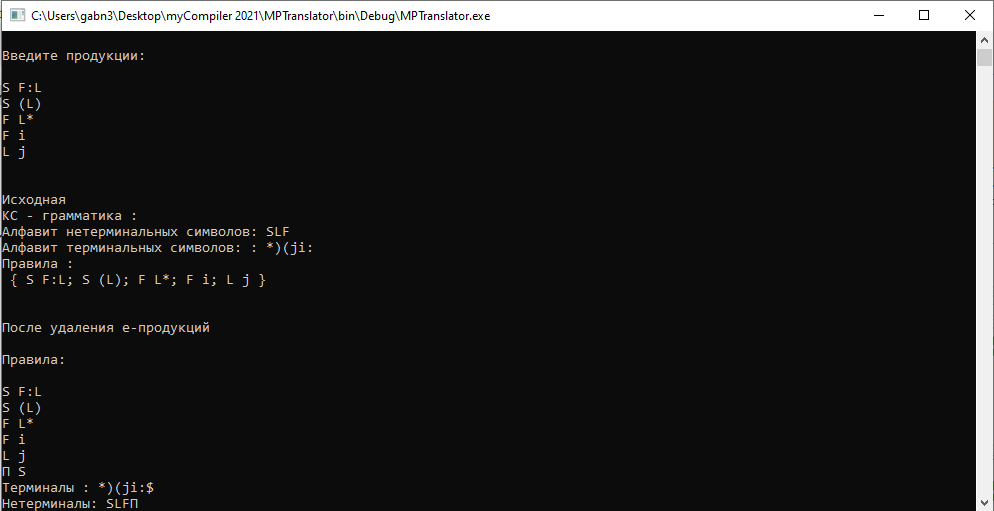
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **I** | **Функция действий** | | | | | | | **Функция переходов** | | |
| **i** | **j** | **:** | **(** | **)** | **\*** |  | **S** | **F** | **L** |
| **0** | П(5) | П(6) |  | П(3) |  |  |  | 1 | 2 | 4 |
| **1** |  |  |  |  |  |  | Д |  |  |  |
| **2** |  |  | П(7) |  |  |  |  |  |  |  |
| **3** |  | П(6) |  |  |  |  |  |  |  | 8 |
| **4** |  |  |  |  |  | П(9) |  |  |  |  |
| **5** | С(4) | С(4) | С(4) | С(4) | С(4) | С(4) |  |  |  |  |
| **6** | С(5) | С(5) | С(5) | С(5) | С(5) | С(5) |  |  |  |  |
| **7** |  | П(6) |  |  |  |  |  |  |  | 10 |
| **8** |  |  |  |  | П(11) |  |  |  |  |  |
| **9** | С(3) | С(3) | С(3) | С(3) | С(3) | С(3) |  |  |  |  |
| **10** | С(1) | С(1) | С(1) | С(1) | С(1) | С(1) |  |  |  |  |
| **11** | С(2) | С(2) | С(2) | С(2) | С(2) | С(2) |  |  |  |  |

**Распознавание цепочек**

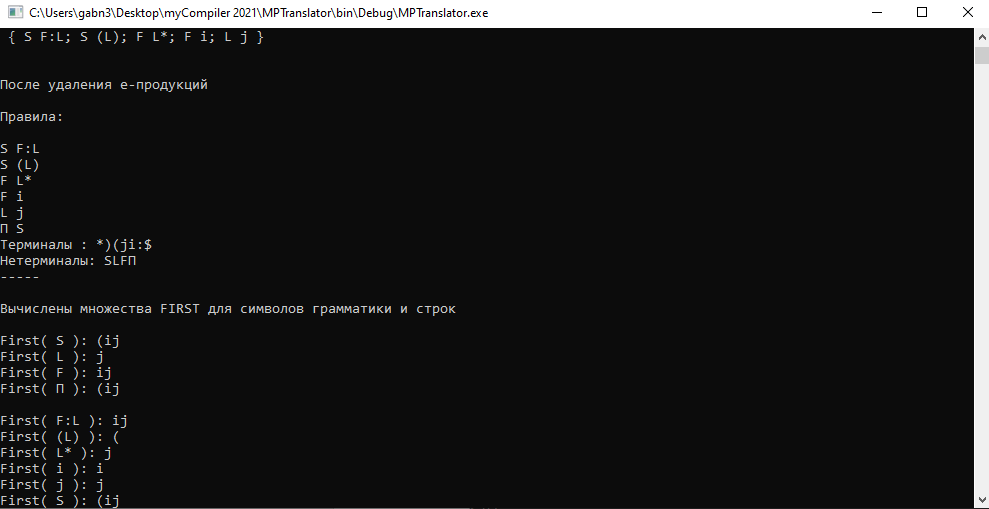
**16. Реализовать LR(k)-анализатор по управляющей таблице (g,f) для LR(k) грамматики.**

**Результат работы программы**

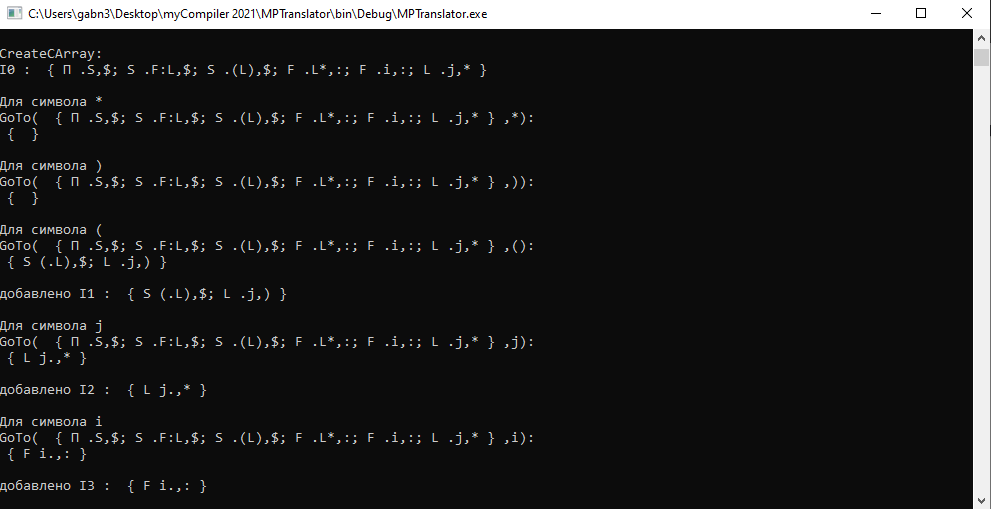
1. Ввод продукций грамматики.



2. Результат преобразования грамматики.



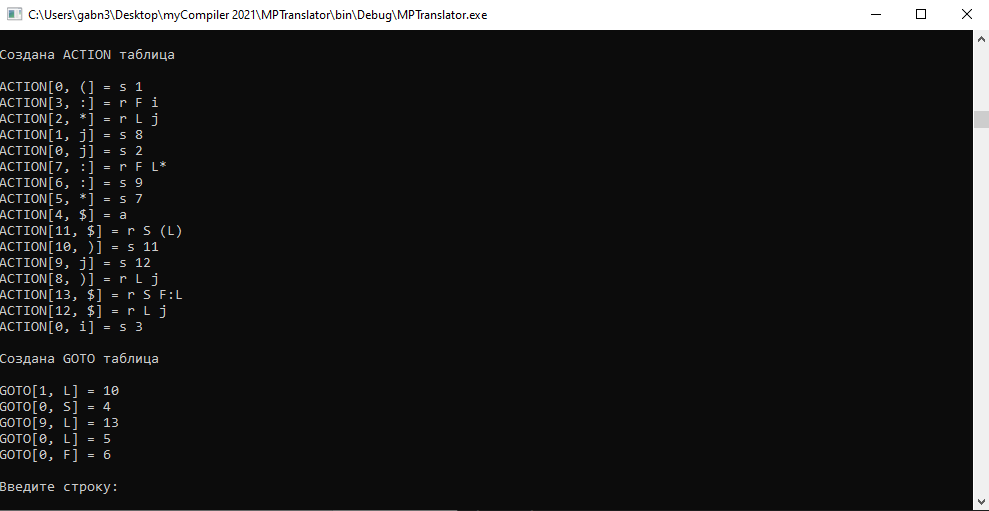
3. Пошаговое построение функции GOTO.



4. Полученная каноническая форма множества ситуаций.



5. Полученная Action и GOTO таблицы.



6. Пример распознавания цепочек.

