**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа № 4

По дисциплине: Теория автоматов и формальных языков

Тема: «Нисходящая обработка контекстно-свободных языков»

Выполнила: ст.группы ПВ - 31

Зановская А.И.

Проверил:

Рязанов Ю.Д.

Белгород

2018

**Нисходящая обработка контекстно-свободных языков**

**Цель работы:** изучить и научиться применять нисходящие методы обработки формальных языков

Вариант 2

1. S → S;O
2. S → O
3. O → Y[S]
4. O → Y[S][S]
5. O → a=E
6. Y → a=a
7. Y → a<a
8. Y → !(Y)
9. E → (E+E)
10. E → (E\*E)
11. E → a

**З а д а н и е**

1. Преобразовать исходную КС-грамматику в LL(1)-грамматику (см. варианты заданий).

Проведем левую факторизацию:

|  |  |
| --- | --- |
| 3. O → Y[S]  4. O → Y[S][S]  Добавим новые правила:  M → [S]  M → ε  Заменим правила 3, 4 на одно:  O → Y[S]M | Результат:   1. S → S;O 2. S → O 3. O → Y[S]M 4. O → a=E 5. Y → a N 6. Y → !(Y) 7. E → (EP 8. E → a 9. M → [S] 10. M → ε 11. N → = a 12. N → < a 13. P → +E) 14. P → \*E) |
| 6. Y → a = a  7. Y → a < a  Добавим новые правила:  N → = a  N → < a  Заменим правила 6, 7:  Y → a N |
| 9. E → (E+E)  10. E → (E\*E)  Добавим новые правила:  P → +E)  P → \*E) |

|  |  |
| --- | --- |
| Устраним левую рекурсию в правиле 1   1. S → S;O 2. S → O   S → OR  R → ;OR  R → ε | Результат:   1. S → OR 2. O → Y[S]M 3. O → a=E 4. Y → aN 5. Y → !(Y) 6. E → (EP 7. E → a 8. M → [S] 9. M → ε 10. N → = a 11. N → < a 12. P → +E) 13. P → \*E) 14. R → ;OR 15. R → ε |

|  |  |  |
| --- | --- | --- |
| **1**  O → Y[S]M  O → a=E  Y → aN  Y → !(Y) | **5**  X → =a[S]M  X → =E  X → <a[S]M | Результат:   1. S → OR 2. O → aX 3. O → !(Y)[S]M 4. X → =L 5. X → <a[S]M 6. Y → aN 7. Y → !(Y) 8. L → aK 9. L → (EP 10. K → [S]M 11. K → ε 12. E → (EP 13. E → a 14. M → [S] 15. M → ε 16. N → = a 17. N → < a 18. P → +E) 19. P → \*E) 20. R → ;OR 21. R → ε |
| **2**  O → aN[S]M  O → a=E  O → !(Y)[S]M  Y → aN  Y → !(Y) | **6**  X → =L  L → a[S]M  L → E |
| **3**  **O → aX**  X → N[S]M  X → =E | **7**  L → a[S]M  L → (EP  L → a |
| **4**  O → aX  X → N[S]M  X → =E  N → = a  N → < a | **8**  L → aK  L → (EP  K → [S]M  K → ε |

2. Определить множества ПЕРВЫХ для каждого символа LL(1)-грамматики.

3. Определить множества СЛЕДУЮЩИХ для каждого символа LL(1)-грамматики.

|  |  |  |
| --- | --- | --- |
| S | a, ! | ] |
| O | a, ! | **|**, ; |
| Y | a, ! | ) |
| L | (, a | **|**, ; |
| K | [ | **|, ;** |
| X | =, < | **|**, ; |
| E | (, a | =, |, ;, ) |
| M | [, ε | ; , | |
| N | =, < | ), [ |
| P | +, \* | | |
| R | ; , ε | ], | |

4. Определить множество ВЫБОРА для каждого правила LL(1)-грамматики.

|  |  |
| --- | --- |
| ВЫБОР (#1) = { a, !}  ВЫБОР (#2) = { a }  ВЫБОР (#3) = { ! }  ВЫБОР (#4) = { = }  ВЫБОР (#5) = { < }  ВЫБОР (#6) = { a }  ВЫБОР (#7) = { ! }  ВЫБОР (#8) = { a }  ВЫБОР (#9) = { ( }  ВЫБОР (#10) = { [ }  ВЫБОР (#11) = { ;, |, ] }  ВЫБОР (#12) = { ( }  ВЫБОР (#13) = { a }  ВЫБОР (#14) = { [ }  ВЫБОР (#15) = { ; , | }  ВЫБОР (#16) = { = }  ВЫБОР (#17) = { < }  ВЫБОР (#18) = { + }  ВЫБОР (#19) = { \* }  ВЫБОР (#20) = { ; }  ВЫБОР (#21) = { ], | } | Результат:  S → OR  O → aX  O → !(Y)[S]M  X → =L  X → <a[S]M  Y → aN  Y → !(Y)  L → aK  L → (EP  K → [S]M  K → ε  E → (EP  E → a  M → [S]  M → ε  N → = a  N → < a  P → +E)  P → \*E)  R → ;OR  R → ε |

5. Написать программу-распознаватель методом рекурсивного спуска. Программа должна выводить последовательность номеров правил, применяемых при левом выводе обрабатываемой цепочки.

public class RecursiveDescent

{

int i = 0;

public void Compile(string str)

{

S(str);

}

void S(string str)

{

if (str[i] == 'a' || str[i] == '!')

{

Console.WriteLine("1. S -> OR");

O(str);

R(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void O(string str)

{

if (str[i] == 'a')

{

Console.WriteLine("2. O -> aX");

i++;

X(str);

}

else if (str[i] == '!')

{

Console.WriteLine("3. O -> !(Y)[S]M");

i++;

if (str[i] == '(')

{

i++;

Y(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

if (str[i] == ')')

{

i++;

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

if (str[i] == '[')

{

i++;

S(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

if (str[i] == ']')

{

i++;

M(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void X (string str)

{

if (str[i] == '<')

{

Console.WriteLine("5. X -> <a[S]M");

i++;

i++;

i++;

S(str);

if (str[i] == ']')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

M(str);

}

else if (str[i] == '=')

{

Console.WriteLine("4. X -> =L");

i++;

L(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void Y(string str)

{

if (str[i] == 'a')

{

Console.WriteLine("6. Y -> aN");

i++;

N(str);

}

else if (str[i] == '!')

{

i++;

if (str[i] == '(')

{

Console.WriteLine("7. Y -> !(Y)");

i++;

Y(str);

if (str[i] == ')')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void L(string str)

{

if (str[i] == 'a')

{

Console.WriteLine("8. L -> aK");

i++;

K(str);

}

else if (str[i] == '(')

{

Console.WriteLine("9. L -> (EP");

i++;

E(str);

P(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void K(string str)

{

if (str[i] == '[')

{

Console.WriteLine("10. K -> [S]M");

i++;

S(str);

if (str[i] == ']')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

M(str);

}

else if (str[i] == ';' || str[i] == '|' || str[i] == ']')

Console.WriteLine("11. K -> e");

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void E(string str)

{

if (str[i] == 'a')

{

Console.WriteLine("13. E -> a");

i++;

}

else if (str[i] == '(')

{

Console.WriteLine("12. E -> (EP");

i++;

E(str);

P(str);

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void M(string str)

{

if (str[i] == '|' || str[i] == ';')

{

Console.WriteLine("15. M -> e");

}

else if (str[i] == '[')

{

Console.WriteLine("14. M -> [S]");

i++;

S(str);

if (str[i] == ']')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void N(string str)

{

if (str[i] == '=')

{

Console.WriteLine("16. N -> =a");

i++;

if (str[i] == 'a')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else if (str[i] == '<')

{

Console.WriteLine("17. N -> <a");

i++;

if (str[i] == 'a')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void P(string str)

{

if (str[i] == '+')

{

Console.WriteLine("17. P -> +E)");

i++;

E(str);

if (str[i] == ')')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else if (str[i] == '\*')

{

Console.WriteLine("18. P -> \*E)");

i++;

E(str);

if (str[i] == ')')

i++;

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

else

{

Console.WriteLine("Не допущено");

Console.ReadKey();

Environment.Exit(0);

}

}

void R(string str)

{

if (str[i] == '|' || str[i] == ']')

Console.WriteLine("21. R -> e");

else if (str[i] == ';')

{

Console.WriteLine("20. R -> ;OR");

i++;

O(str);

R(str);

}

else

{

Console.WriteLine("Не допущено");

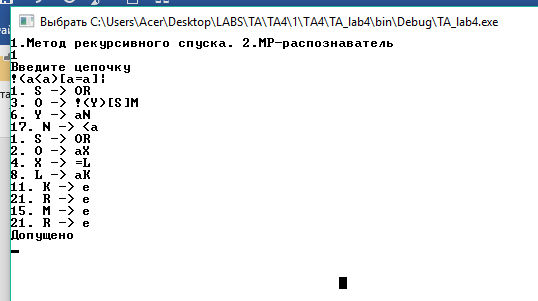
Console.ReadKey();

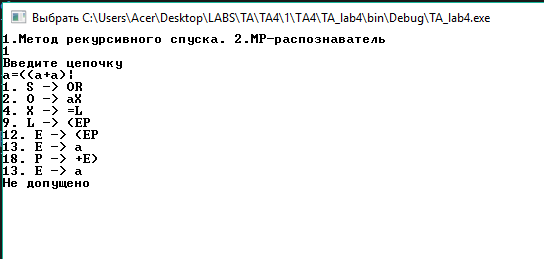
Environment.Exit(0);

}

}

}





8. Построить нисходящий МП-распознаватель по LL(1)-грамматике.

|  |  |
| --- | --- |
| #1: ЗАМ( RO ), держать  #2: ЗАМ( X ), сдвиг  #3: ЗАМ( M]S[)Y ), сдвиг  #4: ЗАМ( L ), сдвиг  #5: ЗАМ( M]S ), сдвиг  #6: ЗАМ( N ), сдвиг  #7: ЗАМ( )Y( ), сдвиг  #8: ЗАМ( K ), сдвиг  #9: ЗАМ( PE ), сдвиг  #10: ЗАМ( M]S ), сдвиг  #11: вытолкнуть, держать  #12: ЗАМ( PE ), сдвиг  #13: вытолкнуть, сдвиг  #14: ЗАМ( ]S ), сдвиг  #15: вытолкнуть, держать  #16: вытолкнуть, сдвиг  #17: вытолкнуть, сдвиг  #18: ЗАМ( E ), сдвиг  #19: ЗАМ( E ), сдвиг  #20: ЗАМ( RO ), сдвиг  #21: вытолкнуть, держать | S → OR  O → aX  O → !(Y)[S]M  X → =L  X → <a[S]M  Y → aN  Y → !(Y)  L → aK  L → (EP  K → [S]M  K → ε  E → (EP  E → a  M → [S]  M → ε  N → = a  N → < a  P → +E)  P → \*E)  R → ;OR  R → ε |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | [ | ] | a | = | ! | ( | ) | < | + | \* | ; | | |
| S |  |  | #1 |  | #1 |  |  |  |  |  |  |  |
| O |  |  | #2 |  | #3 |  |  |  |  |  |  |  |
| X |  |  |  | #4 |  |  |  | #5 |  |  |  |  |
| Y |  |  | #6 |  | #7 |  |  |  |  |  |  |  |
| L |  |  | #8 |  |  | #9 |  |  |  |  |  |  |
| K | #10 |  |  |  |  |  |  |  |  |  | #11 | #11 |
| E |  |  | #13 |  |  | #12 |  |  |  |  |  |  |
| M | #14 |  |  |  |  |  |  |  |  |  | #15 | #15 |
| N |  |  |  | #16 |  |  |  | #17 |  |  |  |  |
| P |  |  |  |  |  |  |  |  | #18 | #19 |  |  |
| R |  | #21 |  |  |  |  |  |  |  |  | #20 | #21 |
| [ | выт→ |  |  |  |  |  |  |  |  |  |  |  |
| ] |  | выт→ |  |  |  |  |  |  |  |  |  |  |
| a |  |  | выт→ |  |  |  |  |  |  |  |  |  |
| = |  |  |  | выт→ |  |  |  |  |  |  |  |  |
| ( |  |  |  |  |  | выт→ |  |  |  |  |  |  |
| ) |  |  |  |  |  |  | выт→ |  |  |  |  |  |
| ^ |  |  |  |  |  |  |  |  |  |  |  | ДОП |

9. Написать программу-распознаватель, реализующую построенный нисходящий МП-распознаватель. Программа должна выводить на каждом шаге номер применяемого правила и промежуточную цепочку левого вывода.

public class MP\_recognizer

{

private Stack<char> mag;

int[][] table;

char[] magSymbols;

char[] chainSymbols;

public MP\_recognizer(string tableFleName)

{

mag = new Stack<char>();

InitTable(tableFleName);

}

private void InitTable(string tableFleName)

{

StreamReader reader = new StreamReader(tableFleName);

string[] arr = reader.ReadLine().Split();

int countMagSymbols = int.Parse(arr[0]);

int countChainSymbols = int.Parse(arr[1]);

magSymbols = new char[countMagSymbols];

chainSymbols = new char[countChainSymbols];

arr = reader.ReadLine().Split();

for (int i = 0; i < arr.Length; i++)

chainSymbols[i] = char.Parse(arr[i]);

table = new int[countMagSymbols][];

for (int i = 0; i < countMagSymbols; i++)

{

table[i] = new int[countChainSymbols];

arr = reader.ReadLine().Split();

magSymbols[i] = char.Parse(arr[0]);

for (int j = 0; j < countChainSymbols; j++)

table[i][j] = int.Parse(arr[j + 1]);

}

}

private bool regulations(int rule, ref Stack<char> mag, ref int i)

{

switch (rule)

{

case 1:

Console.WriteLine("1. S -> OR");

mag.Pop();

mag.Push('R');

mag.Push('O');

break;

case 2:

Console.WriteLine("2. O -> aX");

mag.Pop();

mag.Push('X');

i++;

break;

case 3:

Console.WriteLine("3. O -> !(Y)[S]M");

mag.Pop();

mag.Push('M');

mag.Push(']');

mag.Push('S');

mag.Push('[');

mag.Push(')');

mag.Push('Y');

mag.Push('(');

i++;

break;

case 4:

Console.WriteLine("4. X -> =L");

mag.Pop();

mag.Push('L');

i++;

break;

case 5:

Console.WriteLine("5. X -> a[S]M");

mag.Pop();

mag.Push('M');

mag.Push(']');

mag.Push('S');

mag.Push('[');

i++;

break;

case 6:

Console.WriteLine("6. Y -> aN");

mag.Pop();

mag.Push('N');

i++;

break;

case 7:

Console.WriteLine("7. Y -> !(Y)");

mag.Pop();

mag.Push(')');

mag.Push('Y');

mag.Push('(');

i++;

break;

case 8:

Console.WriteLine("8. L -> aK");

mag.Pop();

mag.Push('K');

i++;

break;

case 9:

Console.WriteLine("9. L -> (EP");

mag.Pop();

mag.Push('P');

mag.Push('E');

i++;

break;

case 10:

Console.WriteLine("10. K -> [S]M");

mag.Pop();

mag.Push('M');

mag.Push(']');

mag.Push('S');

i++;

break;

case 11:

Console.WriteLine("11. K -> e");

mag.Pop();

break;

case 12:

Console.WriteLine("12. K -> (EP");

mag.Pop();

mag.Push('P');

mag.Push('E');

i++;

break;

case 13:

Console.WriteLine("13. E -> a");

mag.Pop();

break;

case 14:

Console.WriteLine("14. M -> [S]");

mag.Pop();

mag.Push(']');

mag.Push('S');

i++;

break;

case 15:

Console.WriteLine("15. M -> e");

mag.Pop();

break;

case 16:

Console.WriteLine("16. N -> =a");

mag.Pop();

mag.Push('a');

i++;

break;

case 17:

Console.WriteLine("17. N -> <a");

mag.Pop();

mag.Push('a');

i++;

break;

case 18:

Console.WriteLine("18. P -> +E)");

mag.Pop();

mag.Push(')');

mag.Push('E');

i++;

break;

case 19:

Console.WriteLine("19. P -> \*E)");

mag.Pop();

mag.Push(')');

mag.Push('E');

i++;

break;

case 20:

Console.WriteLine("20. R -> ;OR");

mag.Pop();

mag.Push('R');

mag.Push('O');

i++;

break;

case 21:

Console.WriteLine("21. R -> e");

mag.Pop();

break;

case 22:

Console.WriteLine($"Выт, сдвиг");

mag.Pop();

i++;

break;

case 23: // Допустить

return true;

}

return false;

}

public void Compile(string chain)

{

char endOfMag = magSymbols[magSymbols.Length - 1];

char startSymbol = magSymbols[0];

mag.Push(endOfMag);

mag.Push(startSymbol);

int currIndex = 0;

int i;

int j;

bool f = false;

while (!f)

{

i = -1;

for (int k = 0; k < magSymbols.Length; k++)

if (magSymbols[k] == mag.Peek())//если магазинный символ равен верхнему значению стека

{

i = k;

break;

}

j = -1;

for (int k = 0; k < chainSymbols.Length; k++)

{

if (chainSymbols[k] == chain[currIndex])

{

j = k;

break;

}

}

if (table[i][j] == 0)

{

Console.WriteLine("Отвергнуть");

Console.ReadKey();

Environment.Exit(0);

}

for (int k = 0; k < currIndex; k++)

Console.Write(chain[k]);

for (int k = 0; k < mag.Count - 1; k++)

{

Console.Write(mag.ElementAt(k));

}

Console.WriteLine();

f = regulations(table[i][j], ref mag, ref currIndex);

Console.Write("\t");

}

Console.WriteLine("Допустить");

Console.ReadKey();

Environment.Exit(0);

}

}

