ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ст. преподаватель |  |  |  | М. Д. Поляк |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| «Нисходящая трансляция на основе вычисления наследуемых и синтезируемых атрибутов» |
| по курсу: Методы трансляции |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

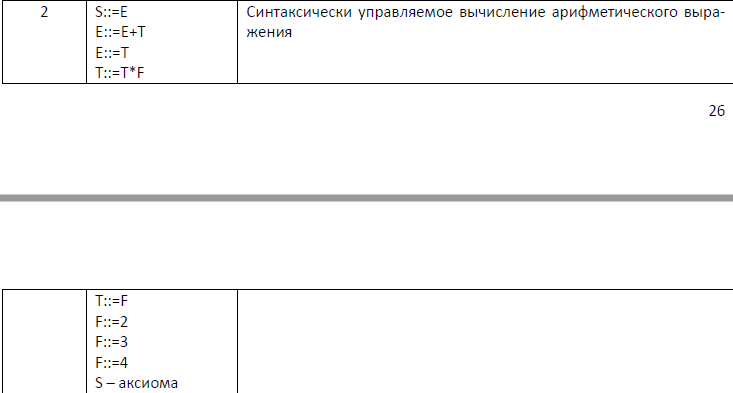
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4832 |  |  |  | М. В. Чупраков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2020

**Цель**

Выполните программную реализацию решения задачи, выбранной по номеру варианта, в виде синтаксически управляемой нисходящей трансляции с использованием наследуемых и синтезируемых атрибутов.

**Вариант 2**



**Ход работы**

1) S ::= E;

2) E ::= E + T;

3) E ::= T;

4) T ::= T \* F;

5) T ::= F;

6) F ::= 2;

7) F ::= 3;

8) F ::= 4;

S – аксиома

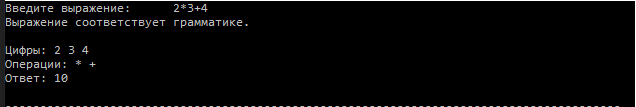
Преобразуем грамматику:

1. S::=E
2. E::= TZ {Ti=Ts}
3. Z::=+TZ {Ti=’+’}
4. T::=FH {Fi=Fs; Ts=Fs}
5. H::=\*FH {Fi=’\*’}
6. F::=2 {Fs=2}
7. F::=3 {Fs=3}
8. F::=4 {Fs=4}
9. Z::=$
10. H::=$

Таблица разбора:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 2 | 3 | 4 | + | \* | # |
| S | 1 | 1 | 1 | ошибка | ошибка | ошибка |
| E | 2 | 2 | 2 | ошибка | ошибка | ошибка |
| Z | ошибка | ошибка | ошибка | 3 | ошибка | 9 |
| T | 4 | 4 | 4 | ошибка | ошибка | Ошибка |
| H | ошибка | ошибка | ошибка | 10 | 5 | 10 |
| F | 6 | 7 | 8 | ошибка | ошибка | ошибка |
| 2 | выброс | ошибка | ошибка | ошибка | ошибка | ошибка |
| 3 | ошибка | выброс | ошибка | ошибка | ошибка | ошибка |
| 4 | ошибка | ошибка | выброс | ошибка | ошибка | ошибка |
| + | ошибка | ошибка | ошибка | выброс | ошибка | ошибка |
| \* | ошибка | ошибка | ошибка | ошибка | выброс | ошибка |
| # | ошибка | ошибка | ошибка | ошибка | ошибка | доступ |

**Результаты выполнения работы**



**Вывод**

В ходе выполнения лабораторной работы была выполнена программная реализация решения задачи в виде нисходящей трансляции на основе вычисления наследуемых и синтезируемых атрибутов.

**Приложение 1 (Исходный код)**

#include <iostream>

#include <string>

#include <vector>

using namespace std;

string gramm[] = { "Blanket", "E","TZ","+TZ","FH","\*FH","2","3","4","","" }; // грамматика

string table\_left = "SEZTHF234+\*#"; // строки таблицы (терминалы + нетерминалы + символ конца строки)

string table\_top = "234+\*#"; // столбцы таблицы (терминалы + символ конца строки)

int table[12][6] =

{//-2 - ошибка; -1 - выброс; 0 - допуск; с 1 до 10 - номера строк ПП, где происходит замена символа левой части на правую часть

{ 1, 1, 1, -2, -2, -2 },

{ 2, 2, 2, -2, -2, -2 },

{ -2, -2, -2, 3, -2, 9 },

{ 4, 4, 4, -2, -2, -2 },

{ -2, -2, -2, 10, 5, 10 },

{ 6, 7, 8, -2, -2, -2 },

{ -1, -2, -2, -2, -2, -2 },

{ -2, -1, -2, -2, -2, -2 },

{ -2, -2, -1, -2, -2, -2 },

{ -2, -2, -2, -1, -2, -2 },

{ -2, -2, -2, -2, -1, -2 },

{ -2, -2, -2, -2, -2, 0 }

};

vector<int> stack\_digit; //стек символов

vector<char> stack\_operation; //стек операций

bool check(string str)

{

string stack = "#S";

str += '#';

//добавление аксиомы и символа конца строки

for (int i = 1; !str.empty(); i++)

{

char symb = str.front();

char stack\_state = stack.back();

int index\_top = table\_top.find(symb);

int index\_left = table\_left.find(stack\_state);

if (index\_top == std::string::npos)

return false;

int command = table[index\_left][index\_top];

switch (command)

{

case 0:

return true;

case -2:

return false;

case -1:

{

if (symb == '2' || symb == '3' || symb == '4')

stack\_digit.push\_back(symb - '0');

if (symb == '\*' || symb == '+') stack\_operation.push\_back(symb);

str.erase(0, 1);

stack.pop\_back();

continue;

}

}

string state = gramm[command];

stack.pop\_back();

stack += string(state.rbegin(), state.rend());;

}

return false;

}

int Ans()

{

int c = 0;

int s = stack\_operation.size();

for (int i = 0; i < s; )

{

if (stack\_operation[i] == '\*')

{

stack\_digit[i] = stack\_digit[i] \* stack\_digit[i + 1];

stack\_digit.erase(stack\_digit.begin() + i + 1);

stack\_operation.erase(stack\_operation.begin() + i);

i = 0;

s = stack\_operation.size();

}

else

i++;

}

cout << "\nОтвет: ";

for (int i = 0; i < stack\_digit.size(); i++)

c += stack\_digit[i];

return c;

}

int main()

{

setlocale(LC\_ALL, "rus");

while (true)

{

string str;

cout << "Введите выражение: \t"; cin >> str;

if (check(str))

cout << "Выражение соответствует грамматике.\n";

else

cout << "\nНекорректный ввод!\n";

cout << "\nЦифры: ";

for (int i = 0; i < stack\_digit.size(); i++) {

cout << stack\_digit[i] << " ";

}

cout << "\nОперации: ";

for (int i = 0; i < stack\_operation.size(); i++) {

cout << stack\_operation[i] << " ";

}

cout << Ans() << endl;

stack\_digit.clear();

stack\_operation.clear();

cout << "\n";

}

return 0;

}