*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана***  ***(национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

**Отчет**

**по домашней работе № 2**

**Дисциплина:** Машинно-зависимые языки программирования

**Название лабораторной работы:**

Теоретические основы построения компилирующих программ

Студент гр. ИУ6-42  **\_\_\_\_\_\_24.05.2018 Бурлаков Александр Сергеевич**

(Подпись, дата) (И.О. Фамилия)

Преподаватель  **\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О. Фамилия)

Москва, 2018

**ЗАДАНИЕ**

Вариант 5.

Паскаль. Разработать грамматику и распознаватель описания типизированных констант – множеств с элементами символьного типа. Тип может быть задан как Char или диапазон типа Char. Конструктор может содержать диапазоны и константы символьного типа. Например: Const A7867:set of Char = [‘h’..’z’,’a’]; vbg:set of ‘a’..’z’ = [];

**ОСНОВНАЯ ЧАСТЬ**

1. **Лексический анализ**

Было решено сделать следующие замены в входной строке:

Таблица 1 – лексические замены

|  |  |  |
| --- | --- | --- |
| № | Было | Стало |
| 1 | Const | co |
| 2 | Set | so |
| 3 | of | oo |
| 4 | Char | ch |
| 5 | ‘h’..’z’ | dp |
| 6 | ‘a’ | sy |

В результате строка «Const A7867:set of Char = [‘h’..’z’,’a’];» преобразуется в строку «co id:so oo ch = [dp,sy];»

1. **Синтаксический анализ**

Составим схему для данного задания:

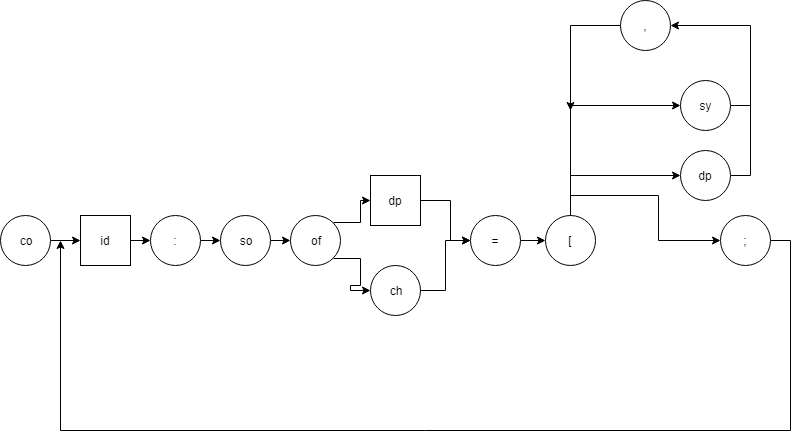


Рисунок 1 – схема синтаксического разбора

Заменим цепочки «id:soofdp=» и «id:soooch=» на «im». Тогда получим следующую схему:

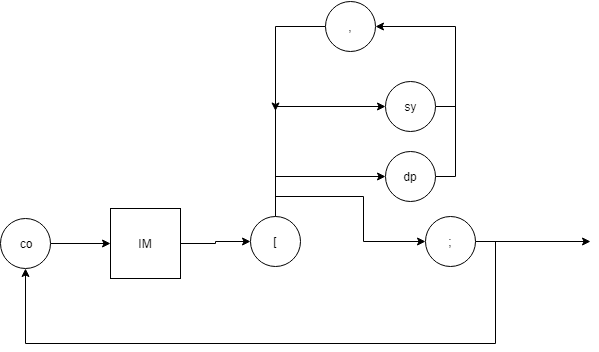


Рисунок 2 – упрощенная схема синтаксического разбора

Разработаем программу по этой схеме:

1. **Код разработанной программы**

**const**

Cword: **array of** string = ('const', 'set', 'of', 'char');

Czam: **array of** string = ('co', 'so', 'oo', 'ch');

Csymb: **set of** char = ['''', '[', ']', ':', '=', '.', ',', ';'];

Mword: **array of** string = ('co', 'id', 'so', 'oo', 'ch', 'dp', 'sy');

Msymb: **array of** string = (':', ',', '[', ']', ';');

**function** id\_zam(S: string): string;

**var**

i: integer;

S1: string;

**begin**

S1 := '';

**while** S <> '' **do**

**begin**

**if** S[1] **in** ['a'..'z', 'A'..'Z'] **then begin**

i := 2;

**while** S[i] **in** ['a'..'z', 'A'..'Z', '0'..'9'] **do**

i := i + 1;

i := i - 1;

**if** (copy(S, 1, i) **in** Cword ) **then**

S1 := S1 + copy(S, 1, i);

**if not** (copy(S, 1, i) **in** Cword ) **then**

S1 := S1 + 'id';

delete(S, 1, i);

**end**

**else**

**if** S[1] = '''' **then begin**

**if** (S[2] **in** ['a'..'z', 'A'..'Z', '0'..'9']) **and** (S[3] = '''') **then**

**if** (copy(S, 4, 3) = '..''') **and** (s[7] **in** ['a'..'z', 'A'..'Z', '0'..'9'])

**and** (s[8] = '''') **then**

**begin**

S1 := S1 + 'dp';

delete(S, 1, 8);

**end**

**else begin**

s1 := s1 + 'sy';

delete(s, 1, 3);

**end**;

**end**

**else begin**

S1 := S1 + S[1];

delete(S, 1, 1);

**end**;

**end**;

id\_zam := S1;

**end**;

**function** scaner(S: string): string;

**var**

S1: string;

i: integer;

**begin**

S := LowerCase(S);

S := id\_zam(S);

S1 := '';

**for** i := 0 **to** 3 **do**

**while** Pos(Cword[i], S) > 0 **do**

**begin**

insert(Czam[i], S, Pos(Cword[i], S));

delete(S, Pos(Cword[i], S), length(Cword[i]));

**end**;

**while** Pos(' ', S) <> 0 **do**

delete(S, Pos(' ', S), 1);

**while** Pos('id:soooch=', S) > 0 **do**

**begin**

i := Pos('id:soooch=', S);

delete(S, i, 10);

insert('im', s, i);

**end**;

**while** Pos('id:sooodp=', S) > 0 **do**

**begin**

i := Pos('id:sooodp=', S);

delete(S, i, 10);

insert('im', s, i);

**end**;

**if** (copy(S, 1, 2) = 'co') **then**

delete(S, 1, 2);

S1 := S;

**while** S1 <> '' **do**

**begin**

**if** (copy(S1, 1, 3) = 'im[') **then**

delete(S1, 1, 3)

**else begin**

scaner := 'Error';

**exit**;

**end**;

**while** (copy(S1, 1, 3) = 'dp,') **or** (copy(S1, 1, 3) = 'sy,') **do**

delete(S1, 1, 3);

**if** (copy(S1, 1, 2) = 'dp') **or** (copy(S1, 1, 2) = 'sy') **then**

delete(S1, 1, 2);

**if** (copy(S1, 1, 2) = '];') **then**

delete(S1, 1, 2)

**else begin**

scaner := 'Error';

**exit**;

**end**;

**end**;

scaner := 'Correct';

**end**;

**var**

S: string;

**begin**

readln(S);

//writeln(id\_zam(LowerCase(S)));

writeln(scaner(S));

**end**.

**Тестирование программы**

Таблица 2 – результаты тестирования программы

|  |  |  |
| --- | --- | --- |
| Входная строка | Ожидаемые результаты | Полученные результаты |
| Const m: set of ‘a’..’z’=[]; | Correct | Correct |
| Const m: set off ‘a’..’z’=[‘a’..’z’]; | Error | Error |
| Const m: set of ‘a’..’z’=[‘z’..’a’]; | Error | Error |





Рисунок 3 – Результаты тестирования программы

**ВЫВОД**

1. Разработан алгоритм, позволяющий решить данное задание.
2. Написан код программы по разработанному алгоритму.
3. Произведено тестирование программы, тестирование показало корректность работы программы.