Ргр по дисциплине “Технологии разработки языков программирования”

Вариант 1 5.

Описание синтаксиса учебного языка с использованием РБНФ.

1. Программа = "program" ИмяПрограммы "\n" Блок "end" “program" ИмяПрограммы.
2. Имя = Буква { Буква | Цифра | "\_" }.
3. Блок = Описания "BEGIN" ПоследОператоров "END".
4. Объявления = [ОпредТипов] ОписПеременных.
5. ОпредТипов = ОпредТипа “\n”.
6. ОписаниеПерем = " Тип "::" ИмяПоля "\n".
7. ОписаниеПеременных = "VAR" "\n" ОписаниеГруппыПеременных { "\n" ОписаниеГруппыПеременных }.
8. ОписаниеГруппыПеременных = СписокИмен "::" Тип.
9. СписокИмен = Имя { "," Имя }.
10. Тип = ПростойТип | ИмяТипа.
11. ПростойТип = "INTEGER" | "REAL" | "LOGICAL".
12. ПоследовательностьОператоров = Оператор { "\n" Оператор }.
13. Оператор = Присваивание | Цикл | УсловныйОператор | Комментарий.
14. Присваивание = Переменная "=" Выражение.
15. Переменная = Имя | ПолеЗаписи.
16. ПолеЗаписи = ИмяЗаписи "%" ИмяПоля.
17. Цикл = "DO" "\n" ПоследовательностьОператоров "\n" "WHILE" "(" Выражение ")".
18. УсловныйОператор = "IF" "(" Выражение ")" "THEN" "\n" ПоследовательностьОператоров [ "\n" "ELSE" "\n" ПоследовательностьОператоров ] "\n" "END" "IF".
19. Выражение = ЛогическоеВыражение | АрифметическоеВыражение.
20. ЛогическоеВыражение = ПростоеЛогВыражение [ ЛогическаяОперация ПростоеЛогВыражение ].
21. ПростоеЛогВыражение = [ "NOT" ] (ЛогическаяКонстанта | Переменная | "(" ЛогическоеВыражение ")" | Отношение ).
22. Отношение = АрифметическоеВыражение ОперацияОтношения АрифметическоеВыражение.
23. ОперацияОтношения = "<" | "<=" | ">" | ">=" | "==" | "/=".
24. ЛогическаяОперация = "AND" | "OR".
25. АрифметическоеВыражение = [ "+" | "-" ] Терм { АддитивнаяОперация Терм }.
26. Терм = Фактор { МультипликативнаяОперация Фактор }.
27. МультипликативнаяОперация = "\*" | "/".
28. Фактор = Число | Переменная | "(" АрифметическоеВыражение ")".
29. Число = Целое | Вещественное.
30. Целое = Цифра { Цифра }.
31. Вещественное = Целое "." Целое.
32. ЛогическаяКонстанта = ".TRUE." | ".FALSE.".
33. Комментарий = "!" { ЛюбойСимвол } "\n".

Семантические соглашения

1. Язык имеет строгую явную статическую типизацию.
2. Все переменные должны быть объявлены до их использования.
3. Базовые типы:
   1. INTEGER - целочисленный тип
   2. REAL - вещественный тип
   3. LOGICAL - логический тип (значения .TRUE. и .FALSE.)
4. Поддерживается определение пользовательских типов - записей (структур) с помощью конструкции TYPE.
5. Цикл REPEAT-UNTIL выполняется минимум один раз, так как условие проверяется после выполнения тела цикла.
6. Приоритет операций:
   1. Скобки
   2. Унарные операции (+, -, NOT)
   3. Мультипликативные операции (\*, /, AND)
   4. Аддитивные операции (+, -, OR)
   5. Операции отношения (<, <=, >, >=, ==, /=)
7. Комментарии начинаются с символа ! и продолжаются до конца строки. Идентификаторы чувствительны к регистру.
8. Для доступа к полям записи используется оператор %.
9. Логические константы записываются как .TRUE. и .FALSE.
10. Программа начинается с ключевого слова PROGRAM и заканчивается END PROGRAM.

Перечень лексических классов (токенов) для языка *DemoLang* представлен в табл. 1 (в качестве формальных шаблонов токенов используются регулярные выражения).

Таблица 1. Таблица лексических классов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №  п/п | Токен | Имя (обозначение) | Код токена | Формальный шаблон | Значение атрибута |
|  | **program** | **prog** | *lcProg* | prog | 0 |
|  | **end** | **start** | *lcStart* | start | 0 |
|  | **begin** | **stop** | *lcStop* | stop | 0 |
|  | **var** | **var** | *lcVar* | var | 0 |
|  | **do** | **do** | *lcDo* | do | 0 |
|  | **while** | **while** | *lcWhile* | while | 0 |
|  | **if** | **if** | *lcIf* | if | 0 |
|  | **then** | **then** | *lcThen* | then | 0 |
|  | **else** | **else** | *lcElse* | else | 0 |
|  | **integer** | **integer** | *lcInteger* | integer | 0 |
|  | **real** | **real** | *lcReal* | real | 0 |
|  | **logical** | **logical** | *lcLogical* | logical | 0 |
|  | **.TRUE.** | **true** | *lcTrue* | true | 0 |
|  | **.FALSE.** | **false** | *lcFalse* | false | 0 |
|  | запятая | **,** | *lcComma* | **,** | 0 |
|  | откр. скобка | **(** | *lcOpPar* | **(** | 0 |
|  | закр. скобка | **)** | *lcClPar* | **)** | 0 |
|  | присваивание | **=** | *lcAss* | **=** | 0 |
|  | аддитивная  операция | **+** | *lcAdd* | **+ | –** | *opAdd* (+)  *opSub* (–) |
|  | мультипл.  операция | **\*** | *lcMult* | **\* | /** | *opMult* (\*)  *opDiv* (/) |
|  | идентификатор | **id** | *lcId* | ***l* (*l* | *d*)\***  *l* – буква, *d* – цифра | указатель на запись  в таблице символов |
|  | число | **num** | *lcNum* | ***d*+ | *d*+.*d*+**  *d* – цифра | указатель на запись  в таблице символов |
|  | Число с плавающей точкой | **real** | *lcReal* | ***r+ . r+ (e/E [+-]? Цифра+)*** | указатель на запись  в таблице символов |
|  | пробел | **sp** |  | (**sp** | **tab** | **lf**)+  sp – пробел,  tab – табуляция,  lf – перевод строки | токен не формируется |
|  | комментарий | **com** |  | **!⊥**  **⊥** – любой символ кроме **!**) | токен не формируется |
|  | маркер конца ввода | **eof** | *lcEof* |  | 0 |
|  | лексическая ошибка | **err** | *lcErr* |  | 0 |

Таблица переходов конечного автомата

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Входные  символы | Состояния автомата (за исключением конечных состояний) | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|  | prog | 10 | - | - | - | - | - | - | - | - | - | - |
|  | start | 10 | - |  |  |  |  |  |  |  | - | - |
|  | stop | 10 | - |  |  |  |  |  |  |  | - | - |
|  | var | 10 | - |  |  |  |  |  |  |  | - | - |
|  | do | 10 | - |  |  |  |  |  |  |  | - | - |
|  | while | 10 | - |  |  |  |  |  |  |  | - | - |
|  | if | 10 | - |  |  |  |  |  |  |  | - | - |
|  | then | 10 | - |  |  |  |  |  |  |  | - | - |
|  | else | 10 | - |  |  |  |  |  |  |  | - | - |
|  | integer | 10 | - |  |  |  |  |  |  |  | - | - |
|  | real | 10 | - |  |  |  |  |  |  |  | - | - |
|  | logical | 10 | - |  |  |  |  |  |  |  | - | - |
|  | true | 10 | - |  |  |  |  |  |  |  | - | -- |
|  | false | 10 | - |  |  |  |  |  |  |  | - | - |
|  | **,** | -16 | - |  |  |  |  |  |  |  | - | - |
|  | **(** | -17 | - |  |  |  |  |  |  |  | - | - |
|  | **)** | -18 | - |  |  |  |  |  |  |  | - | - |
|  | **=** | -19 | - |  |  |  |  |  |  |  | - | - |
|  | **+** | -20 | - |  |  |  |  |  |  |  | - | - |
|  | **-** | -21 | - |  |  |  |  |  |  |  | - | -- |
|  | **\*** | -22 | - |  |  |  |  |  |  |  | - | - |
|  | **/** | -23 | - |  |  |  |  |  |  |  | - | - |
|  | ***l*** | 2 | 2 |  |  |  |  |  |  |  | - | - |
|  | ***d*** | 3 | 2 | 3 | 5 |  |  |  |  |  | - | - |
|  | ***r*** | 4 | - | 4 |  |  |  |  |  |  | - | - |
|  | ***E/e*** |  |  |  |  |  |  |  |  |  |  |  |
|  | **sp** | 7 | - |  |  |  |  |  |  |  | - | - |
|  | **tab** | 7 | - |  |  |  |  |  |  |  | - | - |
|  | If | 7 | - | - | - | - | - | - | - | - | - | - |
|  | ! | 8 | - |  |  |  |  |  |  |  |  |  |
|  | eof | lcEof | - |  |  |  |  |  |  |  |  |  |