**Лабораторная работа 14 (4 часa)**

**Языки программирования**

**Разработка лексического распознавателя (IV часть)**

1. Используйте материал лекций № 18.
2. Используйте результаты лабораторных работ № 10-13.
3. Создайте проект(VS2012, C++, консольное приложение) с именем **LPLab14.**
4. Ознакомьтесь со спецификацией языка SVV-2015.

|  |  |
| --- | --- |
| **Компоненты языка** | **Описание** |
| Символы | Windows-1251 |
| Символы-сепараторы | пробел - допускается везде кроме идентификаторов и ключевых слов;  **;**(точка с запятой) – разделители инструкций;  **{}** – программный блок;  () – параметры;  ()-приоритетность операций. |
| Идентификаторы | только малые буквы, от 1 до 5 букв  идентификатор не может совпадать с ключевыми словами  максимальное количество идентификаторов 216 |
| Типы данных | **integer**– целочисленные данные (четыре байта, от –231 до 231-1),автоматическая инициализация 0, LE;  **string** – строка, любые символы, (макс. 255 символов, первый байт длина строки), автоматическая инициализация строкой длины 0 |
| Операции с данными | **+** - бинарный, суммирование, (integer, integer);  **+** - бинарный, конкатенация, (string, string);  **-** - бинарный, вычитание, (integer, integer);  **\*** - бинарный, умножение, (integer, integer);  **/** - бинарный, деление, (integer, integer) |
| Программные конструкции | главная функция (точка входа):  main  {  return <integer-идентификатор> или <integer-литерал>;  }  Функции:  <тип данных> function<идентификатор> (  <тип данных> <идентификатор>,  ...)  {  return <идентификатор>;  }  Параметры:  передаются по значению. |
| Литералы | числа 231 до 231-1, интерпретируются как integer, могут быть только rvalue – — это выражение, которое ***не*** *представляет* собой объект, который занимает идентифицируемое место в памяти.;  строки, символы, заключенные в ‘’(кавычки), могут быть только rvalue |
| Выражения | арифметическое с применением +, -, /, \*, ();  строковые с применением +,() |
| Инструкции | declare <тип данных> идентификатор – объявление переменных;  declare <тип данных> function идентификатор(<тип данных> идентификатор,... (<тип данных> идентификатор ) – объявление внешних функций;  **=** присвоение значения;  print <идентификатор или литерал>;– вывод в стандартный поток вывода |
| Область видимости | сверху вниз (по принципу С++);  параметры – только внутри функции;  объявление внутри функции – видно только внутри функции |
| Память | Все переменные размещаются в стеке |
| Стандартная библиотека | interger strlen (string) – длина строки;  string substr (string, integer, integer) – подстрока; |
| Максимальное количество функций | 216 |
| Максимальное количество идентификаторов в функции | 216 |

1. Текст программы на языке SVV-2015 для первоначального тестирования (контрольного примера):

*integer function fi(integer x, integer y)*

*{*

*declare integer z;*

*z= x\*(x+y);*

*return z;*

*};*

*string function fs (string a, string b)*

*{*

*declare string с;*

*declare string function substr(string a, integer p,*

*integer n);*

*c = substr(a, 1,3)+ b;*

*return c;*

*};*

*main*

*{*

*declare integer x;*

*declare integer y;*

*declare integer z;*

*declare string sa;*

*declare string sb;*

*declare string sc;*

*declare integer strlen(string p);*

*x = 1;*

*y = 5;*

*sa = ‘1234567890’;*

*sb = ‘1234567890’;*

*z = fi(x,y);*

*sc = fs(sa,sb);*

*print ‘контрольный пример’;*

*print z;*

*print sc;*

*print strlen(sc);*

*return 0;*

*};*

1. Разработайте регулярные выражения, постройте графы переходов и соответствующие конечные автоматы для лексем из следующей таблицы (лабораторные 11-12).

|  |  |  |
| --- | --- | --- |
| Фраза SVV-2015 | лексема | примечание |
| integer  string | t | ТИ: integer или  string, значение по умолчанию: для integer – нуль, для string – пустая строка |
| идентификатор | i | ТИ: строка идентификатора, усеченная до 5 символов.  префикс: имя конструкции |
| литералы | l | integer или  string, значение. |
| function | f |  |
| declare | d |  |
| return | r |  |
| print | p |  |
| main | m |  |
| ; | ; |  |
| , | , |  |
| { | { |  |
| } | } |  |
| ( | ( |  |
| ) | ) |  |
| +  -  \*  / | v |  |

1. Разработайте приложение выполняющее следующее:

* вводит текст программы из входного файла (заданного параметром –in:);
* проверяет входные символы на допустимость, в соответствии с таблицей символов (лабораторная 10);
* удаляет лишние пробелы и добавляет сепаратор для вычисления номера строки для каждой лексемы;
* осуществляет лексический анализ всех фраз программы (лабораторная 12-13) и строит таблицу лексем и таблицу идентификаторов;
* формирует протокол работы: протокол работы должен содержать содержимое таблицы лексем с разбивкой по строкам с нумерацией, примерно в следующем вида (ниже представлен фрагмент), а также диагностические сообщения (с применение таблицы ошибок) с указанием номера строки и позиции в исходном тексте;

01 tfi(ti,ti)

02 {

03 dti;

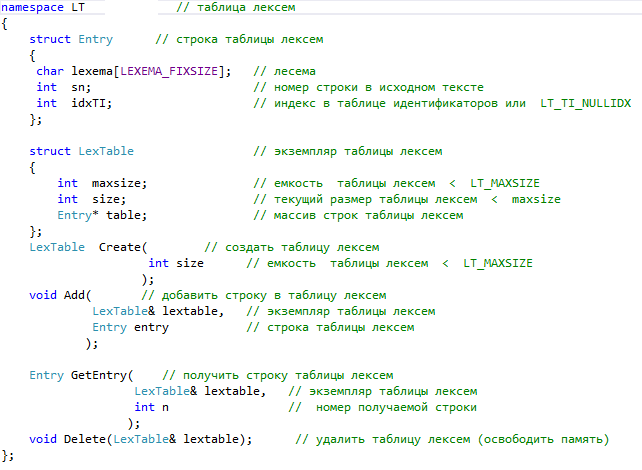
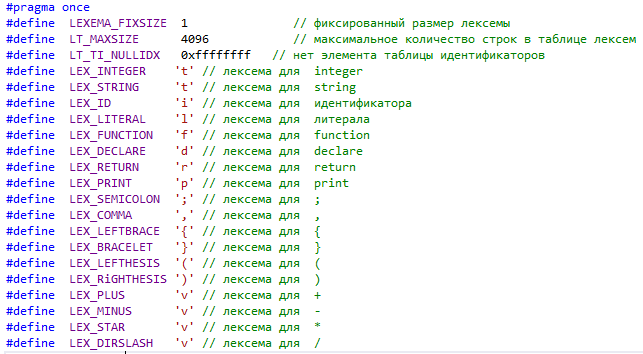
04 i=i\*(i+i);

05 ri;

06 };

* обрабатывает ошибки (использует таблицу ошибок из лабораторной 10, добавьте в нее необходимые коды ошибок), все ошибки должны обрабатываться с помощью таблицы.

1. В контрольном примере, должны вводится строки, которые анализировались в лабораторной 12.
2. Необходимо подготовить примеры, в которых демонстрируется обработка всех типов ошибок.
3. При построении таблицы лексем используйте следующую спецификацию (h-файл LT.h)



1. При построении таблицы идентификаторов используйте следующую спецификацию (h-файл IT.h)

