

**Министерство образования и науки Российской Федерации**

**Федеральное агентство по образованию**

**Государственное образовательное учреждение высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Робототехника и комплексная автоматизация» (РК)**

**Кафедра «Системы автоматизированного проектирования» (РК6)**

****

**Отчет по лабораторной работе \_\_\_\_\_A18 (N)\_\_\_\_\_\_\_ по курсу**

**«САПР»**

**Студент: Сергеева Диана**

**Группа:** РК6-36Б

**Преподаватель:** Волосатова Т.М.

Проверил:

Дата:

2020

**Задание**

**Текст задания:**

Разработать конечный автомат для распознавания записей целых чисел в системе счисления по основанию 16, где все цифры расположены в неубывающем порядке. Записи всех целых чисел должны передаваться конечному автомату через строки потока стандартного ввода. Результаты их распознавания должны отображаться строками потока стандартного вывода. Программная реализация конечного автомата должна формироваться генератором LEX по правилам с предусловиями.

**Введение**

Настоящий документ определяет техническое задание на разработку программы лексического анализатора, реализующего конечный автомат записей чисел в системе счисления по основанию 16 в формате системы программирования C.

**Основания для разработки**

Программа разрабатывается в рамках лабораторной работы по курсу "Лингвистическое обеспечение САПР" для практического изучения этапа лексического анализа в процедурах трансляции формальных языков.

**Назначение разработки**

Программа предназначается для распознавания целых чисел в системе счисления по основанию 16, где все цифры расположены в неубывающем порядке.

**Требования к программе**

1. Требования к функциональным характеристикам
   1. Программа должна в интерактивном режиме распознавать шестнадцатеричные наборы.
   2. Шестнадцатеричные наборы должны быть в неубывающем порядке.
   3. Шестнадцатеричные наборы должны передаваться строками стандартного ввода.
   4. Программа обнаруживает соответствие или несоответствие шестнадцатеричных последовательностей заданным правилам.
   5. При нарушении ввода или несоответствии последовательности заданным правилам, программа должна вывести сообщение об ошибке.
   6. Результат распознавания программой шестнадцатеричных последовательностей, соответствующей заданным правилам, должен отображаться в поток стандартного вывода.
2. Требования к надежности

Программа не должна иметь каких-либо ограничений по числу символов в анализируемой шестнадцатеричной последовательности, кроме внутренних ограничений инструментальных средств, использованных для ее реализации.

1. Условия к составу и параметрам технических средств

Программа должна быть разработана исходя из возможности реализации на стандартном составе технических средств компьютеров любой архитектуры, после соответствующей трансляции исходного кода.

1. Условия эксплуатации
   1. Программа должна быть ориентирована на эксплуатацию в среде OS UNIX.
   2. Программа должна быть реализована в виде выполняемого файла, по которому она должна вызываться средствами любого командного процессора OS UNIX.
2. Требования к информационной и программной совместимости
   1. Чтобы обеспечить выполнение требуемых технических характеристик, программа должна реализовывать лексический анализ любой входной строки, содержащая шестнадцатеричный набор, из потока стандартного ввода.
   2. Лексический анализатор программы должен обеспечивать лексический разбор шестнадцатеричных наборов с целью установить соответствие или несоответствие содержащей их строки потока стандартного ввода заданным правилам.
   3. Для разработки лексического анализатора программы, необходимо использовать генератор лексических анализаторов из состава OS UNIX, инструментальные средства которого ориентированы на обработку файла спецификаций проектируемого лексического анализатора.
   4. При разработке лексического анализатора программы необходимо составить файл спецификации, отражающий специфику лексического разбора.
   5. Проектируемый файл спецификации должен состоять из 3-х секций: описания, правил и подпрограмм. Разделителем секций должны быть символические пары %%, расположенные в начальных позициях содержащих их строк.
   6. В разделе деклараций файла программы необходимо: объявить метки предусловий обработки каждого числа в шестнадцатеричной системе счисления, используя директиву %Start в первой строке файла.
   7. В разделе правил файла программы должны быть введены правила, которые обеспечивают распознавание необходимых лексем и их функциональную обработку посредством блоков действий.
   8. Правила раздела правил файла программы должны включать метки предусловий, которые обеспечивают распознавание следующих лексем: весь набор цифр шестнадцатеричной системы счисления, любые символы вне этого набора.
   9. Правила раздела правил файла программы должны обеспечивать выполнение следующих действий при распознавании лексем: установка и инициализация меток предусловий с помощью оператора BEGIN, запускать состояние некорректности введенных данных, если есть лексемы, не относящиеся к шестнадцатеричной системе счисления.
   10. Исходный код секции функций файла спецификации образует исходный код программы, который должен формироваться генератором лексических анализаторов в файле с предопределенным именем lex.yy.c в текущем рабочем каталоге файловой системы OS UNIX. Выполняемый модуль программы должен быть создан по файлу исходного кода lex.yy.c в выполняемом файле a.out средствами компилирующей системы программирования C.

**Стадии и этапы разработки**

В процессе разработки программы необходимо выполнить следующие этапы:

* подготовить файл спецификаций лексического анализатора для программы с именем A18.l, используя любой текстовый редактор OS UNIX;
* получить исходный С-код программы в файле lex.yy.c, обработав файл спецификации командой lex следующим образом: lex A18.l
* построить по исходному коду lex.yy.c выполняемый файл A18 программы, вызвав C-компилятор OS UNIX следующим образом: cc -o A18 A18.c.

**Порядок контроля и приемки**

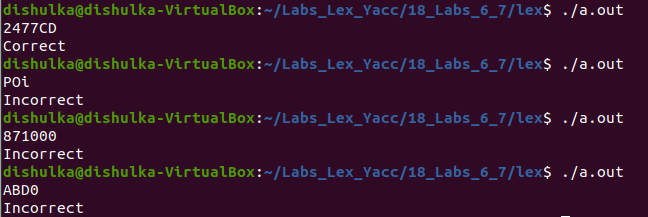
1. Для проверки функционирования программы должны быть предложены контрольные примеры, предусматривающий стандартный ввод корректных и некорректных данных.

2. Для приемки программы должен быть организован вызов выполняемого файла в консольном режиме работы OS.

**Текст программы:**

%START L1 L2 L3 L4 L5 L6 L7 L8 L9 LA LB LC LD LE LF  
%{  
[#include](https://vk.com/im?sel=63386677&st=%23include) <stdio.h>  
char\* mas[] = {"Correct", "Incorrect"};  
%}   
  
%%  
  
<LF>[^0-9A-Fa-f\n] return 1;  
<LF>F BEGIN LF;  
<LF>[0-9A-Ea-e] return 1;  
<LF>\n return 0;  
  
<LE>[^0-9A-Fa-f\n] return 1;  
<LE>F BEGIN LF;  
<LE>E BEGIN LE;  
<LE>[0-9A-Da-d] return 1;  
<LE>\n return 0;  
  
<LD>[^0-9A-Fa-f\n] return 1;  
<LD>F BEGIN LF;  
<LD>E BEGIN LE;  
<LD>D BEGIN LD;  
<LD>[0-9A-Ca-c] return 1;  
<LD>\n return 0;  
  
<LC>[^0-9A-Fa-f\n] return 1;  
<LC>F BEGIN LF;  
<LC>E BEGIN LE;  
<LC>D BEGIN LD;  
<LC>C BEGIN LC;  
<LC>[0-9A-Ba-b] return 1;  
<LC>\n return 0;  
  
<LB>[^0-9A-Fa-f\n] return 1;  
<LB>F BEGIN LF;  
<LB>E BEGIN LE;  
<LB>D BEGIN LD;  
<LB>C BEGIN LC;  
<LB>B BEGIN LB;  
<LB>[0-9Aa] return 1;  
<LB>\n return 0;  
  
<LA>[^0-9A-Fa-f\n] return 1;  
<LA>F BEGIN LF;  
<LA>E BEGIN LE;  
<LA>D BEGIN LD;  
<LA>C BEGIN LC;  
<LA>B BEGIN LB;  
<LA>A BEGIN LA;  
<LA>[0-9] return 1;  
<LA>\n return 0;  
  
<L9>[^0-9A-Fa-f\n] return 1;  
<L9>F BEGIN LF;  
<L9>E BEGIN LE;  
<L9>D BEGIN LD;  
<L9>C BEGIN LC;  
<L9>B BEGIN LB;  
<L9>A BEGIN LA;  
<L9>9 BEGIN L9;  
<L9>[0-8] return 1;  
<L9>\n return 0;  
  
<L8>[^0-9A-Fa-f\n] return 1;  
<L8>F BEGIN LF;  
<L8>E BEGIN LE;  
<L8>D BEGIN LD;  
<L8>C BEGIN LC;  
<L8>B BEGIN LB;  
<L8>A BEGIN LA;  
<L8>9 BEGIN L9;  
<L8>8 BEGIN L8;  
<L8>[0-7] return 1;  
<L8>\n return 0;  
  
<L7>[^0-9A-Fa-f\n] return 1;  
<L7>F BEGIN LF;  
<L7>E BEGIN LE;  
<L7>D BEGIN LD;  
<L7>C BEGIN LC;  
<L7>B BEGIN LB;  
<L7>A BEGIN LA;  
<L7>9 BEGIN L9;  
<L7>8 BEGIN L8;  
<L7>7 BEGIN L7;  
<L7>[0-6] return 1;  
<L7>\n return 0;  
  
<L6>[^0-9A-Fa-f\n] return 1;  
<L6>F BEGIN LF;  
<L6>E BEGIN LE;  
<L6>D BEGIN LD;  
<L6>C BEGIN LC;  
<L6>B BEGIN LB;  
<L6>A BEGIN LA;  
<L6>9 BEGIN L9;  
<L6>8 BEGIN L8;  
<L6>7 BEGIN L7;  
<L6>6 BEGIN L6;  
<L6>[0-5] return 1;  
<L6>\n return 0;  
  
<L5>[^0-9A-Fa-f\n] return 1;  
<L5>F BEGIN LF;  
<L5>E BEGIN LE;  
<L5>D BEGIN LD;  
<L5>C BEGIN LC;  
<L5>B BEGIN LB;  
<L5>A BEGIN LA;  
<L5>9 BEGIN L9;  
<L5>8 BEGIN L8;  
<L5>7 BEGIN L7;  
<L5>6 BEGIN L6;  
<L5>5 BEGIN L5;  
<L5>[0-4] return 1;  
<L5>\n return 0;  
  
<L4>[^0-9A-Fa-f\n] return 1;  
<L4>F BEGIN LF;  
<L4>E BEGIN LE;  
<L4>D BEGIN LD;  
<L4>C BEGIN LC;  
<L4>B BEGIN LB;  
<L4>A BEGIN LA;  
<L4>9 BEGIN L9;  
<L4>8 BEGIN L8;  
<L4>7 BEGIN L7;  
<L4>6 BEGIN L6;  
<L4>5 BEGIN L5;  
<L4>4 BEGIN L4;  
<L4>[0-3] return 1;  
<L4>\n return 0;  
  
<L3>[^0-9A-Fa-f\n] return 1;  
<L3>F BEGIN LF;  
<L3>E BEGIN LE;  
<L3>D BEGIN LD;  
<L3>C BEGIN LC;  
<L3>B BEGIN LB;  
<L3>A BEGIN LA;  
<L3>9 BEGIN L9;  
<L3>8 BEGIN L8;  
<L3>7 BEGIN L7;  
<L3>6 BEGIN L6;  
<L3>5 BEGIN L5;  
<L3>4 BEGIN L4;  
<L3>3 BEGIN L3;  
<L3>[0-2] return 1;  
<L3>\n return 0;  
  
<L2>[^0-9A-Fa-f\n] return 1;  
<L2>F BEGIN LF;  
<L2>E BEGIN LE;  
<L2>D BEGIN LD;  
<L2>C BEGIN LC;  
<L2>B BEGIN LB;  
<L2>A BEGIN LA;  
<L2>9 BEGIN L9;  
<L2>8 BEGIN L8;  
<L2>7 BEGIN L7;  
<L2>6 BEGIN L6;  
<L2>5 BEGIN L5;  
<L2>4 BEGIN L4;  
<L2>3 BEGIN L3;  
<L2>2 BEGIN L2;  
<L2>[0-1] return 1;  
<L2>\n return 0;  
  
<L1>[^0-9A-Fa-f\n] return 1;  
<L1>F BEGIN LF;  
<L1>E BEGIN LE;  
<L1>D BEGIN LD;  
<L1>C BEGIN LC;  
<L1>B BEGIN LB;  
<L1>A BEGIN LA;  
<L1>9 BEGIN L9;  
<L1>8 BEGIN L8;  
<L1>7 BEGIN L7;  
<L1>6 BEGIN L6;  
<L1>5 BEGIN L5;  
<L1>4 BEGIN L4;  
<L1>3 BEGIN L3;  
<L1>2 BEGIN L2;  
<L1>1 BEGIN L1;  
<L1>0 return 1;  
<L1>\n return 0;  
  
1 BEGIN L1;  
2 BEGIN L2;  
3 BEGIN L3;  
4 BEGIN L4;  
5 BEGIN L5;  
6 BEGIN L6;  
7 BEGIN L7;  
8 BEGIN L8;  
9 BEGIN L9;  
A BEGIN LA;  
B BEGIN LB;  
C BEGIN LC;  
D BEGIN LD;  
E BEGIN LE;  
F BEGIN LF;  
  
.|\n return 1;  
  
%%  
  
int main(){  
int ret = yylex();  
printf("%s\n", mas[ret]);  
return ret;  
}

**Тесты:**

****

**Используемая литература:**

1. Кристиан К. Введение в операционную систему UNIX, M.: Финансы bстатистика, 1985.
2. Тихомиров В.П., Давидов М.И. Операционная система ДЕМОС: инструментальные средства программирования, М.: Финансы и статистика, 1988.
3. SCO XENIX, Development System, LEX Programmer Guide, SCO Inc., 1986.
4. bigor.bmstu.ru – база и генератор образовательных ресурсов
5. Лекции по курсу САПР
6. Автоматизация Проектирования Лексических Анализаторов (Т.М. Волосатова, С.В. Родионов)