

**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 L34(P) по курсу**

**«САПР»**

**Студент: Петраков Станислав**

**Группа:** РК6-36Б

**Преподаватель:** Родионов С.В.

Проверил:

Дата:

**Задание**

**Текст задания:**

Разработать лексический анализатор для поиска в программной документации любых записей маршрутных имен файлов OS UNIX, с учетом общепринятых обозначений текущего(.), родительского(..) и домашнего(~) каталогов. Текст программной документации должен передаваться лексическому анализатору строками стандартного ввода. Результаты поиска маршрутных имен должны отображаться в одну колонку стандартного вывода.

**Введение**

Настоящий документ определяет техническое задание на разработку программы лексического анализатора строк в формате системы программирования C.

**Требования к программе**

1. Требования к функциональным характеристикам
   1. Программа должна выделять из программной документации записи маршрутный листов имен файлов в файловой системе OS UNIX.
   2. Каждая запись должна выводится с новой строки
   3. Должна определять любые виды маршрутных листов
2. Требования 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.
   3. Программа должна эксплуатироваться в режиме фильтра, который читает поток стандартного ввода и отображает результаты в потоке стандартного вывода OS UNIX.
2. Требования к информационной и программной совместимости
   1. При разработке программы ЛАК необходимо использовать генератор лексических анализаторов (далее по тексту - LEX) из состава инструментальных средств OS UNIX, который ориентирован на обработку файла спецификаций (далее по тексту - Lex-файл) проектируемого лексического анализатора.
   2. Lex-файл prog.l программы ЛАК должен содержать 3 раздела:

• раздел деклараций;

• раздел правил;

• раздел функций,

которые должны разделять символьные пары %%.

**Требования к программной документации**

1. По результатам разработки необходимо составить пояснительную записку в текстовом файле prog.lex, содержащую описание Lex-файла.
2. Для автоматизации составления пояснительной записки необходимо обработать комментарии Lex-файла lex.l программой ЛАК, используя следующий конвейер команд OS UNIX:

$ cat lex.l | prog > prog.lex

и любой текстовый редактор (например, xedit) для минимальной пост- корректировки содержания полученного описания в файле prog.lex.

**Стадии и этапы разработки**

В процессе разработки программы ЛАК средствами LEX необходимо выполнить следующие этапы:

• подготовить файл спецификаций лексического анализатора (Lex-файл) для программы с именем prog.l, используя любой текстовый редактор OS UNIX;

• получить исходный С-код программы ЛАК в файле prog.c, обработав Lex-файл prog.l командой lex следующим образом:

$ lex -t prog.l > prog.c

• построить по исходному коду prog.c выполняемый файл prog программы, вызвав C-компилятор OS UNIX и подключив стандартную библиотеку объектных модулей LEX libl.a следующим образом:

$ cc -o prog prog.c -ll

**Порядок контроля и приемки**

1. Для контроля функционирования программы ЛАК необходимо разработать контрольный пример, содержащий различные варианты компоновки блоков комментариев, сохранив его в текстовом файле prog.chk.

2. Для реализации контроля программы по контрольному примеру необходимо использовать следующий конвейер команд OS UNIX:

$ cat prog.chk | prog | more

который обеспечивает постраничную или построчную визуализацию результатов обработки комментариев в файле prog.chk.

**Текст спецификации prog.l:**

%%

((\.\.)|\.|~)?(\/[^\/\0\n\t\ ]\*)+\/? ECHO; putchar('\n');

\n|. {}

**Тестовый файл prog.chk:**

kjdshfsjha ../name/name2/names.txt kjfdsgh fdjks

jksdhjk ds jdshgfsdk sd

~/Desktop/Name/

kjfdslo lksd ./dot/dot.c dsf

/bin/

**Выходные данные для тестового файла prog.chk:**

../name/name2/names.txt

~/Desktop/Name/

./dot/dot.c

/bin/

**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)