Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Отчёт по лабораторной работе №1**

Дисциплина: Транслирующие системы

Тема: Программирование лексического разбора на языке Lex

Выполнил студент гр. 5130901/20103 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А. Вагнер

(подпись)

Принял старший преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.Н. Цыган

(подпись)

“ ” 2024 г.

Санкт-Петербург

2024

**Цель работы.** Ознакомиться с основными возможностями утилиты Lex. Lex — это генератор программ лексической обработки текстов.

**Ход работы:**

Рассмотрим программу, которая передает в выходной поток все литеры входного потока кроме пробелов и/или табуляций в начале строки.

Листинг 1 (ex01.l) – Удаление пробелов и табуляций в начале строк

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Поскольку действие пустое, то последовательности, соответствующие этому шаблону, игнорируются. Литеры, не распознанные ни одним правилом, передаются в выходной поток.

Было выполнена трансляция командой **lex ex01**.l После вызова **./ex01** были введены две строки: без начальных пробелов и с пробелами. Ввод и вывод выполняются через стандартные потоки, и можно использовать перенаправление: **./ex01 <test.in >test.out**.

Для проверки правильности работы программы, которая передает в выходной поток все литеры входного потока кроме пробелов и/или табуляций в начале строки, были написаны следующие тесты:

Изображение выглядит как текст, Шрифт, снимок экрана, черный

Автоматически созданное описание

Рис. 1 – Текстовый файл с тестами для ex01.l

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рис. 2 – Результаты тестирования для ex01.l

Программа ex01.l была успешно протестирована.

Если необходимо, чтобы программа не пропускала на выход непонятные ей литеры, то действие по умолчанию нужно блокировать, указав ключ -s при вызове lex. Проверим это на примере ex01.l, где действие по умолчанию превалирует.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Было получено предупреждение уже при трансляции; выполнение ./ex1v2 с файлом test.in закончилось сообщением ― flex scanner jammed (заклинило).

В примере ex02.l есть правила и для '\n', и для точки (означает "любой символ кроме \n"). То есть программа распознает все литеры, и поэтому нет нужды задавать ключ -s — он, кстати, и не рекомендуется стандартом POSIX.

Листинг 2 (ex02.l) – Подсчет числа строк

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Для проверки правильности работы программы был написан следующий тест:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рис. 3 – Текстовый файл с тестом для ex02.l



Рис. 4 – Результаты тестирования для ex02.l

Программа ex02.l была успешно протестирована.

Распознанная входная последовательность литер сохраняется в массиве yytext; ее длина записывается в переменную yyleng.

Пользователь может исправлять содержимое yytext в пределах первых yyleng позиций. Первая литера найденной строки доступна как yytext[0], а последняя — как yytext[yyleng-1].

В следующем примере задан подсчет последовательностей, которые обозначают знаковые целые числа; каждый раз при обнаружении такой последовательности выводится текущее значение счетчика чисел и текст лексемы.

Листинг 3 (ex03.l) – Подсчет и вывод знаковых целых чисел

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тесты:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 5 – Текстовый файл с тестами для ex03.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 6 – Результаты тестирования для ex03.l

Программа ex03.l была успешно протестирована.

Вывод yytext — это настолько частое действие, что для него определена макрокоманда ECHO. В следующем примере в выходной поток передаются идентификаторы и беззнаковые числа, по одному на строке, а все прочее отсеивается. Литера '|' справа от шаблона означает "то же действие, что и для следующего правила''.

Листинг 4 (ex04.l) – Вывод идентификаторов и беззнаковых целых чисел

Изображение выглядит как текст, снимок экрана, Шрифт, Графика

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 7 – Текстовый файл с тестами для ex04.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 8 – Результаты тестирования для ex04.l

Программа ex04.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

В этом примере уже достаточно много правил, чтобы проверить отладочный режим lex. Выполним трансляцию: **lex –d ex04.l** — и протестируем программу ex04\_debug.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Далее демонстрируется вывод для программы ex04\_debug, на вход которой были даны тестовые последовательности из файла test.in. В отладочном режиме можно увидеть какими именно шаблонами распознается та или иная последовательность.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеl

Рис. 9 – Результаты работы в отладочном режиме для ex04.l

В отладочном режиме в выходной поток выводится трассировка работы лексического анализатора. Как только срабатывает какой-то шаблон (совпало содержание входной цепочки с описанным в шаблоне), то выводится и сама эта цепочка в выходной поток, и выводится номер строки программы, которой соответствует этот шаблон.

Также рассмотрим вариант вызова, с раздельным перенаправлением потоков stdout и stderr. Это вариант вызова удобен при тестировании примеров на тему lex и yacc, когда включен отладочный режим. Вывод этих программ по умолчанию идет в stdout, а отладочные сообщения направляются в stderr.

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

Рис. 10 – Результаты работы в отладочном режиме с раздельным перенаправлением потоков stdout и stderr для ex04.l

Использование переменной yyleng показано в программе подсчета идентификаторов по длине. Результат — гистограмма длин слов в диапазоне от 1 до 40, в виде текста. Обратим внимание на первое правило, которое состоит только из действия. Это действие выполняется один раз при запуске программы.

Листинг 5 (ex05.l) – Подсчет и вывод гистограммы длин слов

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 11 – Текстовый файл с тестами для ex05.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 12 – Результаты тестирования для ex05.l

Программа ex05.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

После успешного выполнения пяти пунктов работы мне было выдано контрольное задание. Было необходимо составить шаблон на основе следующей спецификации:



Листинг 6 (task.l) – Вывод слов длинной не более чем из 40 букв, первая буква – ‘a’, две последние – ‘yz’

Изображение выглядит как текст, снимок экрана, Шрифт, Графика

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 13 – Текстовый файл с тестами для task.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 14 – Результаты тестирования для task.l

Функции yymore, yyless(n) дают дополнительные возможности по управлению yytext. yymore — отключает режим перезаписи для следующего (одного) сопоставления, т. е. литеры следующей лексемы будут добавлены к текущему содержимому yytext. yyless(n) — сокращает строку в yytext до n первых литер, возвращая остаток во входной поток.

Листинг 7 (ex06.l) – Вывод строки наискосок при помощи yyless

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 15 – Текстовый файл с тестами для ex06.l

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рис. 16 – Результаты тестирования для ex06.l

Программа ex06.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Пользователь может обращаться к функциям низкоуровневого ввода-вывода, которые используются лексическим анализатором:

* input — чтение следующей литеры из входного потока (в конце потока считывается null-литера);
* output(c) — запись литеры c в выходной поток;
* unput(c) — запись литеры c во входной поток.

В следующем примере функция input используется для поиска конца комментария, заданного в стиле языка C — /\* \*/. Также демонстрируются макроопределения 16-ричных цифр H, десятичных цифр D и букв L и их подстановки: {H}, {D} и {L}.

Листинг 8 (ex07\_1.l) – Макросы и ввод-вывод низкого уровня

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рис. 17 – Текстовый файл с тестами для ex07\_1.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 18 – Результаты тестирования для ex07\_1.l

Программа ex07\_1.l для данных тестовых последовательностей была успешно протестирована.

Функция unput в skip\_comments предназначена для обработки частного случая /\*?\*\*/ (подряд более одной '\*' перед '/'). Но в ex07\_1.l нет проверки конца входного потока, так что незакрытый комментарий приведет к зацикливанию в процедуре skip\_comments.

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рис. 19 – Текстовый файл с незакрытым комментарием

Правильное решение — всегда проверять результат input на равенство EOF.

Листинг 9 (ex07\_2.l) – Проверка конца входного потока при использовании input

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рис. 21 – Текстовый файл с тестами для ex07\_2.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 22 – Результаты тестирования для ex07\_2.l

Программа ex07\_2.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

В рассмотренном примере (листинги 8 и 9) определены правила для распознавания имен и чисел (десятичных и 16-ричных чисел в стиле ассемблера a86). Для сокращения записи этих правил в разделе определений заданы макроопределения шаблонов, обозначающих буквы, десятичные и 16-ричные цифры; подстановки заданы именами макрокоманд в фигурных скобках.

Если представить себе входной поток в виде магнитофонной ленты, то функция input считывает ее при воспроизведении, а unput — это запись на перемотке в начало. В следующем примере задано реверсирование идентификаторов, начинающихся с '@':

Листинг 10 (ex08\_1.l) – Функция unput

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 23 – Текстовый файл с тестами для ex08\_1.l

Этот пример годится не для всех реализаций lex. Возможно, что функция unput будет изменять величину yyleng (уменьшать на 1) и содержимое yytext (удалять крайнюю литеру).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 24 – Результаты тестирования для ex08\_1.l

Результаты испытаний не соответствуют спецификации (слово первая половина слова совпадает с изначальным вариантом, вторая половина – является отзеркаленной первой половиной), что вполне логично. Поэтому лучше использовать копию yytext и yyleng.

Листинг 11 (ex08\_2.l) – Дублирование yytext и yyleng при работе с unput

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex08\_1.l.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 25 – Результаты тестирования для ex08\_2.l

Программа ex08\_2.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Рассмотрим выбор правил при сопоставлении и управление множеством правил.

Если при поиске лексемы входная последовательность может быть распознана несколькими шаблонами, то набор правил двусмысленный. В этой ситуации правило выбирается по следующей схеме:

* Предпочтение отдается соответствию большей длины;
* Если одна и та же последовательность соответствует нескольким правилам, предпочтение отдается тому правилу, которое задано раньше других.

Листинг 12 (ex09.l) – Двусмысленный набор правил

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 26 – Текстовый файл с тестами для ex09.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 27 – Результаты тестирования для ex09.l

Программа ex09.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Ввод "ready'' принимается вторым правилом, поскольку "[a-z]+'' распознает все 5 литер ("ready''), в то время как первое правило — только 4 ("read''). При вводе "read'' оба правила распознают одинаковое число литер — 4, и будет выбрано первое правило, т. к. оно задано раньше. Ввод меньшей длины, например, "re'', не приводит к неопределенности, поскольку воспринимается только вторым правилом.

Для правил с выражениями типа ".\*'' поиск наиболее длинного соответствия приводит к неожиданным результатам. Например, для выявления строк в одиночных кавычках может показаться подходящим следующее решение.

Листинг 13 (ex10.l) – Неправильный шаблон для распознавания строки в кавычках

Изображение выглядит как снимок экрана, Шрифт, текст

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Рис. 28 – Текстовый файл с тестами для ex10.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 29 – Результаты тестирования для ex10.l

Результаты испытаний не соответствуют спецификации, так как шаблон задает поиск самой дальней закрывающей кавычки, хотя и в пределах строки.

То есть при вводе 'first' here, 'second' there будет выявлено 'first' here, 'second'. Хорошо, что поиск по шаблону ".\*'' ограничен текущей входной строкой, т. к. '.' означает любую литеру кроме новой строки. Попытка обойти это ограничение с помощью шаблона (.|\n)+ приведет к бесконечному сопоставлению.

Правильное решение формулируется так: между кавычками могут быть любые литеры кроме кавычки и конца строки.

Листинг 14 (ex11.l) – Правильный шаблон для распознавания строки в кавычках

Изображение выглядит как Шрифт, снимок экрана, текст, Графика

Автоматически созданное описание

Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex10.l.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 30 – Результаты тестирования для ex11.l

Программа ex11.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Стартовые условия позволяют на ходу изменить множество действующих правил и тем самым приспособиться к изменению контекста. Но сначала рассмотрим более простой способ с использованием переменной состояния. Предположим, требуется в каждой строке заменить "magic'' на "first'', "second" или "third' в зависимости от того, какая цифра была в начале строки — 1, 2, или 3.

Листинг 15 (ex12.l) – Использование переменной состояния

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Автоматически созданное описание

Рис. 31 – Текстовый файл с тестами для ex12.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 32 – Результаты тестирования для ex12.l

Программа ex12.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Теперь решим эту задачу при помощи стартовых условий. Чтобы воспользоваться ими, их нужно сначала объявить:

%start cond1, cond2, ...

Эти условия можно добавить к правилам, записав:

<cond>шаблон

Это правило действительно тогда, когда текущее стартовое условие анализатора - cond. Текущее стартовое условие устанавливается макрокомандой:

BEGIN (cond)

Вернуться к исходному (нулевому) стартовому условию можно так:

BEGIN (INITIAL)

Листинг 16 (ex13\_1.l) – Решение при помощи стартовых условий

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex12.l.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 33 – Результаты тестирования для ex13\_1.l

Программа ex13\_1.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

На уровне реализации стартовые условия — это целые числа (в частности, INITIAL = 0). Это обстоятельство позволяет проводить трассировку стартовых условий.

Макроопределение YY\_USER\_ACTION, по умолчанию пустое, позволяет задать код, который выполняется перед действием любого правила. Макрокоманда YYSTATE возвращает численное значение текущего стартового условия.

Листинг 17 (ex13\_2.l) – Трассировка стартовых условий

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex12.l.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 34 – Результаты тестирования для ex13\_2.l

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 35 – Отдельно выходной поток stdout для результатов тестирования ex13\_2.l



Рис. 36 – Отдельно выходной поток stderr для результатов тестирования ex13\_2.l

Программа ex13\_2.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Во всех рассмотренных программах выявляются смежные (примыкающие друг к другу) последовательности. Анализ вложенных и перекрывающихся последовательностей требует применения специальных средств.

В следующем примере запрограммирован счет последовательностей "she'' и "he''. Но эта программа не выявляет экземпляры "he'' внутри "she'', т. к. после распознавания "she'' эти литеры уходят из входной последовательности.

Листинг 18 (ex14\_1.l) – Подсчет количества she и he без учета he внутри she

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Для проверки правильности работы программы были написаны следующие тестовые последовательности:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Рис. 37 – Текстовый файл с тестами для ex14\_1.l



Рис. 38 – Результаты тестирования для ex14\_1.l

Программа ex14\_1.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

Для выявления вложенной последовательности нужно:

1. вернуть принятую последовательность во входной поток;

2. исключить правило, которым была распознана эта последовательность;

3. возобновить сопоставление.

Первая фаза этого действия может быть реализована вызовом yyless(0), вторая — при помощи стартовых условий. Но можно задать это действие одной макрокомандой REJECT.

Листинг 19 (ex14\_2.l) – Подсчет всех экземпляров she и he

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex14\_1.l.



Рис. 39 – Результаты тестирования для ex14\_2.l

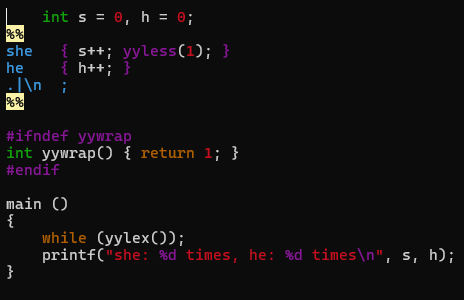
Программа ex14\_2.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

При обнаружении "she'' увеличивается счетчик s, команда REJECT отвергает правило и возвращает "she'' на вход. Затем предпринимается попытка заново сопоставить тот же ввод с оставшимися шаблонами.

В этом примере можно учесть то, что "she'' включает в себя "he'', но не наоборот, и убрать REJECT из второго действия. Но когда в шаблонах задано повторение, невозможно предугадать, сколько литер каким правилом будет распознано.

В примере с "she" и "he" можно заменить REJECT на yyless.

Листинг 20 (ex14\_3.l) – Подсчет she и he с использованием yyless



Для проверки правильности работы программы были использованы те же тестовые последовательности, что и для тестирования программы ex14\_1.l.



Рис. 40 – Результаты тестирования для ex14\_3.l

Программа ex14\_3.l была успешно протестирована. Результаты испытаний соответствуют спецификации.

**В****ывод:** в ходе выполнения лабораторной работы были протестированы примеры программ лексического разбора на языке lex (было проверено их соответствие спецификации). Были изучены структура и синтаксис программ на языке lex. Для одной из программ была проверена работа режима отладки. Также было выполнено небольшое индивидуальное задание, целью которого было написать программу для выявления определённых последовательностей.