Лабораторная работа №5
«Лексический распознаватель»

Скоробогатов С. Ю., Коновалов А. В.

29 марта 2016

1 Цель работы

Целью данной работы является изучение использования детерминированных конечных автоматов с размеченными заключительными состояниями (лексических распознавателей) для решения задачи лексического анализа.

2 Исходные данные

Пусть лексическая структура модельного языка состоит из шести лексических доменов:

1. пробелы — непустые последовательности пробельных символов (пробел, горизонтальная табуляция, маркеры конца строки);
2. идентификаторы — непустые последовательности латинских букв и десятичных цифр, начинающиеся с буквы;
3. целочисленные литералы — непустые последовательности десятичных цифр;
4. ключевые слова (варианты ключевых слов перечислены в таблицах 1, 2 и 3);
5. знаки операций (варианты знаков операций перечислены в таблицах 1, 2 и 3);
6. комментарии или строковые литералы (варианты перечислены в таблицах 1, 2 и 3).

Чтобы не усложнять лексический анализатор, разрешим идентификаторам примыкать справа к целочисленным литералам.

3 Задание

Выполнение лабораторной работы состоит из пяти этапов:

1. описание лексических доменов модельного языка в виде регулярных выражений;
2. построение недетерминированного лексического распознавателя для модельного языка;
3. детерминизация построенного лексического распознавателя и факторизация его алфавита;

4. построение массива обобщённых символов, матрицы переходов и массива заключительных состояний для полученного детерминированного лексического распознавателя с факторизованным алфавитом;

5. разработка лексического анализатора, работающего на основе интерпретации построенных структур данных.

Входной поток для лексического анализатора должен загружаться из файла (в ASCII). В результате работы программы в стандартный поток вывода должны выдаваться описания распознанных лексем в формате

```
Тег (координаты_фрагмента): изображение_лексемы
```

При этом лексемы, принадлежащие домену пробелов, должны отбрасываться.

Лексический анализатор должен иметь программный интерфейс для взаимодействия с парсером. Рекомендуется реализовывать его как метод `nextToken()` для императивных языков или функцию, возвращающую список лексем, для функциональных языков.

Входной файл может содержать ошибки, при обнаружении которых лексический анализатор должен выдавать сообщение с указанием координаты, восстанавливаться и продолжать работу.

В лабораторной работе разрешается использовать любой язык программирования, поддерживающий массивы с операцией доступа к элементу по индексу, работающей за константное время.

Таблица 1: Наборы ключевых слов и знаков операций по вариантам

1	`if`, `then`, `else`, `(`, `)`, комментарии ограничены знаками `{`, `}` могут пересекать границы строк текста.
2	`do`, `while`, `<`, `>`, `<>`, строковые литералы ограничены апострофами, не могут пересекать границы строк текста.
3	`begin`, `end`, `>`, `>=`, комментарии начинаются со знака `#` и продолжаются до конца строки.
4	`for`, `range`, `==`, `!=`, строковые литералы ограничены двойными кавычками, не могут пересекать границы строк текста.
5	`while`, `wend`, `<`, `<=`, комментарии начинаются со знака `--` и продолжаются до конца строки.
6	`if`, `fi`, `do`, `od`, `{`, `}`, строковые литералы ограничены обратными кавычками, могут пересекать границы строк текста.
7	`case`, `break`, `*`, `(`, `)`, комментарии ограничены знаками `(*`, `*)` могут пересекать границы строк текста.
8	`class`, `super`, `-`, `->`, строковые литералы ограничены двойными кавычками, могут содержать escape-последовательности вида `\x`, где `x` — любой символ, не могут пересекать границы строк текста.
9	`struct`, `type`, `=`, `==`, комментарии начинаются с `//` и продолжаются до конца строки.
10	`def`, `val`, `var`, `[`, `]`, строковые литералы ограничены одинарными кавычками, для включения одинарной кавычки в строку она удваивается, не могут пересекать границы строк текста.
11	`int`, `float`, `<`, `<<`, комментарии ограничены знаками `@`, могут пересекать границы строк текста.
12	`fun`, `let`, `in`, `:`, `::`, строковые литералы ограничены обратными кавычками, могут содержать escape-последовательности `\n`, `\\`, `` \` `` (остальные недопустимы), не могут пересекать границы строк текста.
13	`select`, `from`, `:)`, `:(`, комментарии ограничены знаками `[` и `]`, не могут пересекать границы строк текста.
14	`for`, `forward`, `&&`, `||`, строковые литералы ограничены двойными кавычками, для включения кавычки в строковой литерал она предваряется знаком «\» (но знак «\» без последующей кавычки ошибкой не является), могут пересекать границы строк текста.
15	`real`, `longreal`, `>=`, `:=`, комментарии начинаются с `::` и продолжаются до конца строки.

Таблица 2: Наборы ключевых слов и знаков операций по вариантам

16	`uint8t`, `uint128t`, `(*, *)`, строковые литералы ограничены знаками `/`, допустимы escape-последовательности вида `\x`, где `x` — любой символ, могут пересекать границы строк текста.
17	`goto`, `gosub`, `\(`, `\)`, комментарии начинаются с `\\` и заканчиваются в конце строки.
18	`delete`, `decltype`, `->`, `<-`, строковые литералы ограничены апострофами, могут пересекать границы строк текста, для включения апострофа внутрь строки он удваивается.
19	`if`, `elif`, `*`, `/`, комментарии ограничены знаками `/*`, `*/` могут пересекать границы строк текста.
20	`lambda`, `quote`, `+`, `=`, `+=`, строковые литералы ограничены двойными кавычками, не могут пересекать границы строк текста.
21	`get`, `set`, `||`, `+`, комментарии начинаются с `|+` и заканчиваются на `+|`, могут пересекать границы строк текста, последовательность знаков `|+` внутри комментария является синтаксической ошибкой.
22	`signed`, `unsigned`, `.`, `..`, строковые литералы ограничены запятыми, могут пересекать границы строк текста.
23	`plus`, `minus`, `>=`, `<=`, `==`, комментарии начинаются с апострофа и продолжаются до конца строки.
24	`eq`, `neq`, `+-`, `-+`, строковые литералы ограничены знаками `:`, для включения двоеточия в строку оно удваивается, не могут пересекать границы строк текста.
25	`key`, `val`, `~~`, комментарии ограничены знаками `~`, могут пересекать границы строк текста.
26	`write`, `read`, `^_^`, `^__^`, строковые литералы ограничены двойными кавычками, кавычку в конце строки допустимо не ставить.
27	`exit`, `exist`, `!`, `!~`, комментарии начинаются со знака `~` и продолжаются до конца строки.
28	`set`, `unset`, `()`, строковые литералы начинаются с `(`, заканчиваются на `)`, не могут содержать внутри круглые скобки и не могут пересекать границы строк текста.
29	`ax`, `eax`, `rax`, `[`, `]`, комментарии начинаются со знака `;` или `#` и продолжаются до конца строки.
30	`begin`, `end`, `{`, `}`, строковые литералы ограничены знаками `$`, допустимы escape-последовательности вида `\x`, где `x` — любой символ, не могут пересекать границы строк текста.

Таблица 3: Наборы ключевых слов и знаков операций по вариантам

31	`def`, `return`, `(`, `)`, `:`, комментарии ограничены знаками `"""`, могут пересекать границы строк текста.
32	`mov`, `jmp`, `->`, `$`, строковые литералы начинаются со знака `#` и оканчиваются знаком `$`, не могут пересекать границы строк текста.
33	`switch`, `case`, `:`, `{`, `}`, комментарии начинаются со знака `!` и продолжаются до конца строки.
34	`open`, `close`, `<<`, `>>`, строковые литералы ограничены одинарными кавычками, для включения кавычки в строковой литерал она предваряется знаком «\», могут пересекать границы строк текста.
35	`Integer`, `Float`, `::`, `->`, `=`, комментарии ограничены знаками `{-`, `-}`, могут пересекать границы строк текста.
36	`all`, `clean`, `:`, `!!`, строковые литералы начинаются со знака `#` и продолжаются до конца строки, и если последним символом в строке является знак `\`, то следующая строка также считается частью строкового литерала, иначе строковый литерал заканчивается.
37	`update`, `where`, `==`, `!=`, комментарии начинаются со знака `!!` и продолжаются до конца строки, и если последним символом в строке является знак `!`, то следующая строка также считается частью комментария, иначе комментарий заканчивается.
38	`define`, `typedef`, `#`, `;`, строковые литералы ограничены знаками `%`, допустимы escape-последовательности вида `\x`, где `x` — любой символ, могут пересекать границы строк текста.
39	`title`, `body`, `<`, `>`, `</`, `/>`, комментарии ограничены знаками `<!--`, `-->`, могут пересекать границы строк текста.
40	`new`, `delete`, `=`, `(`, `)`, строковые литералы ограничены знаками `-`, для включения знака `-` в строку он удваивается, не могут пересекать границы строк текста.