|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт информационных технологий |
| Кафедра инструментального и прикладного программного обеспечения (ИиППО) |
|  |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1** | |
| **по дисциплине** | |
| **«**Теория автоматов и формальных языков**»**  **по теме** «Лексический анализатор» | |
|  | |
| Выполнил студент группы ИКБО-16-17 | Акжигитов Р. Р. |
| Принял старший преподаватель | Зорина Н. В. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «25» ноября 2019 г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2019 г. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Москва 2019

**Оглавление**

[Цель лабораторной работы 3](#_Toc25591746)

[Задание 3](#_Toc25591747)

[Выполнение лабораторной работы 3](#_Toc25591748)

[Результат работы программы 5](#_Toc25591749)

[Заключение 6](#_Toc25591750)

[Список литературных источников 7](#_Toc25591751)

# Цель лабораторной работы

Целью данной лабораторной работы является изучение и разработка лексического анализатора с использования инструментального средства Flex.

# Задание

Для выполнения лабораторной работы необходимо:

1) написать программу, которая выполняет лексический анализ входного текста в соответствии с заданием и порождает таблицу лексем с указанием их типов. Программа должна выдавать сообщения о наличии во входном тексте ошибок, которые могут быть обнаружены на этапе лексического анализа;

2) в качестве вспомогательного средства для генерации кода лексического анализатора использовать Flex.

**Вариант 2:** входной язык содержит функцию printf, круглые скобки, кавычки, запятую, символ окончания функции (точка с запятой), восклицательный и вопросительные знаки.

# Выполнение лабораторной работы

Для выполнения задания необходимо создать файл input.l. В нем будет описан лексический анализатор. Также необходимы файл y.tab.h, где будут описаны токены и файл simple.txt, где будет описан пример входных данных для анализатора. Опишем токены в файле y.tab.h.

// y.tab.h

#define SEMICOLON 1

#define BECOMES 2

#define PLUS 3

#define MINUS 4

#define TIMES 5

#define SLASH 6

#define LPAR 7

#define RPAR 8

#define HEXNUM 9

#define IDENT 10

#define UNKNOWN 11

#define QUOTE 12

#define QUEST 13

#define EXCLAM 14

#define COMMA 15

#define PRINTF 16

Здесь описаны все необходимые токены, которые лексический анализатор должен разобрать в ходе анализа. У каждого токена есть свой идентификатор, например, 1 и т.д. Затем нужно описать, собственно, сам лексический анализатор. Также необходимо описать функцию Main, которая будет считывать файл и анализировать токены и выводить номера токенов, которые она смогла распознать. Будет представлена в общем листинге.

// input.l

%option noyywrap yylineno

%{

#include "y.tab.h"

%}

digit [0-9]

hexdigit [a-f]

letter [a-zA-Z]

%%

"(" { return LPAR; }

")" { return RPAR; }

"\"" { return QUOTE; }

"," { return COMMA; }

";" { return SEMICOLON; }

"!" { return EXCLAM; }

"?" { return QUEST; }

"printf" { return PRINTF; }

{digit}({hexdigit}|{digit})\* { return HEXNUM; }

{letter}({letter}|{digit})\* { return IDENT; }

%%

int main(int argc, char \*\*argv) {

if (argc < 2) {

printf("\nNot enough arguments. Please specify filename.\n");

return -1;

}

if ((yyin = fopen(argv[1], "r")) == NULL) {

printf("\nCannot open file %s.\n", argv[1]);

return -1;

}

yylineno = 1;

while (1) {

int token = yylex();

if (token == 0)

break;

printf("%d\n", token);

}

fclose(yyin);

return 0;

}

Последний файл simple.txt, в котором представлен пример входных данных.

// simple.txt

printf("?,!?,!?");

# Результат работы программы

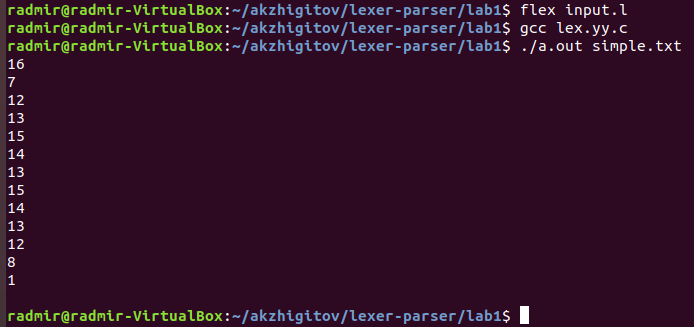


Рисунок 1 – результат работы лексического анализатора

# Заключение

В данной лабораторной работе были изучены принципы построения лексического анализатора (состав, принцип работы, способы описания, инструментальные средства для генерации анализаторов, например, flex). Выполнено задание, в котором описаны необходимые для варианта токены, также описана главная функция для считывания символов из файла и передачи их в лексер, для последующего выделения лексем.

# Список литературных источников

1. Альфред В. Ахо, Моника С. Лам, Рави Сети, Джеффри Д. Ульман. Компиляторы: принципы, технологии и инструментарий = Compilers: Principles, Techniques, and Tools. — 2-е изд. — М.: Вильямс, 2008. — ISBN 978-5-8459-1349-4.
2. Робин Хантер. Основные концепции компиляторов = The Essence of Compilers. — М.: «Вильямс», 2002. — С. 256. — ISBN 5-8459-0360-2.
3. Молчанов А.Ю. Системное программное обеспечение: Учебник для вузов. 3-е изд. – СПб.: Питер, 2010 – 400 с.
4. Генератор синтаксических анализаторов, совместимый с YACC для Bison версии 1.35, 25 февраля 2002 [URL: https://www.opennet.ru/docs/RUS/bison\_yacc/]