Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра ПОИТ

Дисциплина «Компиляторные технологии»

ОТЧЁТ

к лабораторной работе №1

РАСПОЗНАВАНИЕ И ПОИСК ЛЕКСЕМ

Вариант 3

Студент группы №451001

Цурко В.Е.

Руководитель, преподаватель

Болтак С.В.

Минск 2025

**СОДЕРЖАНИЕ**

[1 Задание к лабораторной работе 3](#_Toc177731385)

[2 Классы символов и регулярное выражение 4](#_Toc177731386)

[3 Недетерминированный конечный автомат (НКА) 4](#_Toc177731387)

[4 Детерминированный конечый автомат (ДКА) 4](#_Toc177731388)

[5 Тестирование программы 5](#_Toc177731389)

[6 Текст программы 5](#_Toc177731390)

## 1 Задание к лабораторной работе

В таблице 1 представлен перечень пронумерованных заданий. Номер задания соответствует номеру варианта (третий).

Таблица 1 – Варианты заданий к лабораторной работе №1

|  |  |  |
| --- | --- | --- |
| № варианта | Требования | Примеры корректных строк |
| 1 | Строковый литерал языка C с поддержкой следующих экранирующих последовательностей:   * \n * \t * \\ * \" | "Hello, world!"  "This \"string\" is valid."  "Hello\nworld\t!"  "Quotes are written this way: \\\"." |
| 2 | Строковый литерал языка Pascal (апостроф записывается, как два апострофа подряд). | 'Hello, world!'  'I don''t know.' |
| 3 | 16-чное число в синтаксисе C или Pascal. | $c001c0De  0xCa5EFeED |
| 4 | Классический абсолютный путь к файлу в Windows:   * латинская буква от A до Z; * двоеточие; * обратный слеш; * 0 или более блоков из допустимых символов, разделённых обратными слешами.   К допустимым относятся все символы, кроме «\*», «|», «\», «:», «"», «<», «>», «?» и «/».  (Для поиска подстрок недопустимым символом пути следует также считать пробел « ».) | C:\Windows\winmine.exe  D:\WebServer\home\site.by\www\.htaccess  Z:\autoexec.bat  N:\testfile.  X:\testfile2 |
| 5 | Двоичное, восьмеричное или 16-чное число в синтаксисе FASM (постфиксная форма записи). 16-чное число должно начинаться с цифры от 0 до 9. | 01101011B  54212110o  51245H  0B8Ch |
| 6 | Корректный идентификатор: начинается с буквы или нижнего подчёркивания, затем — произвольное количество букв, цифр и/или нижних подчёркиваний. | abCde  x0  Y  z\_1  test\_value\_is\_here  \_  \_\_custom |
| 7 | Адрес электронной почты (упрощённо).  Имя ящика — произвольный набор букв, цифр и символов «.» (точка). Доменное имя — одна или более групп, состоящих из букв, цифр и дефисов, разделённые точками. | test@localhost  vasya@pupkin.ru  ivan.ivanov@mail.bsuir.by  a.b.c@d-e-f.com |

## 2 Классы символов и регулярное выражение

К выделенным в ходе выполнения лабораторной работы классам символов относятся:

* *Letters* ::= [A..Fa..f];
* *Digit* ::= [0..9];
* *StartN* ::= ($, 0x, 0X).

Регулярное выражение, соответствующее условию задачи, выглядит следующим образом: *StartN(Letters+ | Digit+)\*.*

## 3 Недетерминированный конечный автомат (НКА)

На рисунке 3.1 представлена схема недетерминированного конечного автомата, обрабатывающего строку на входе и определяющего, является ли она строковым литералом в языке Pascal.

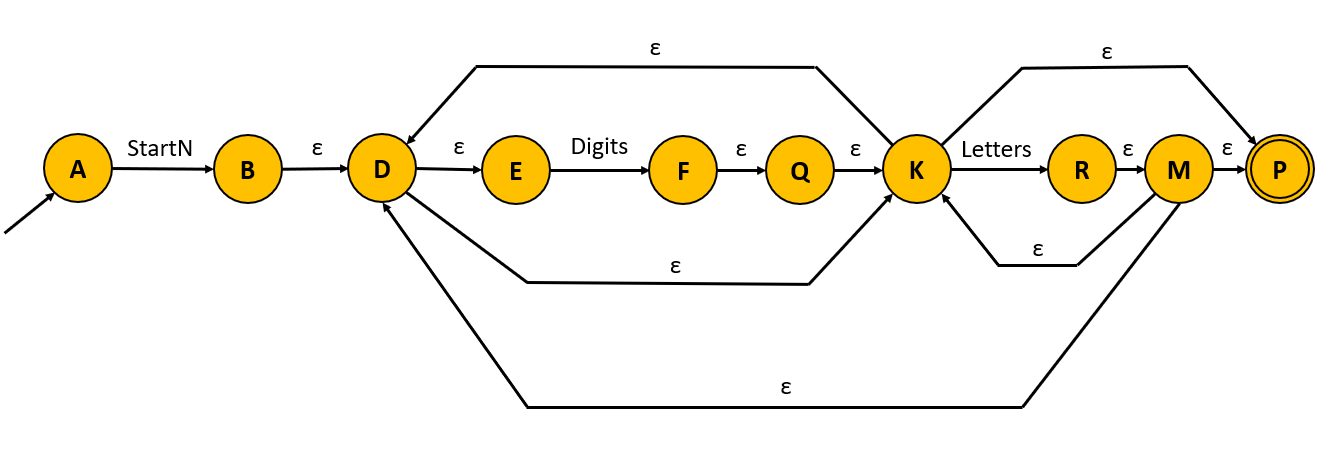


Рисунок 3.1 – Недетерминированный конечный автомат для 16-чного числа в С и Pascal

## 4 Детерминированный конечный автомат (ДКА)

На рисунке 4.1 представлена схема детерминированного конечного автомата, обрабатывающего 16-чное число в С и Pascal.

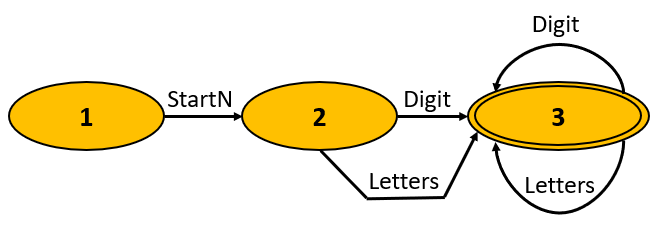


Рисунок 4.1 – Детерминированный конечный автомат для 16-чного числа в С и Pascal

## 5 Тестирование программы

В таблице 2 представлены ключевые тесты, проверяющие корректность результата работы программы.

Ожидаемый результат: если введённая строка является строковым литералом в языке Pascal – 1, если не является – 0.

Таблица 2 – Тесты программы

|  |  |  |  |
| --- | --- | --- | --- |
| Номер и исход теста | Входные  данные | Ожидаемый результат | Результат работы программы |
| 1 (пройден успешно) | ‘’ | 1 | 1 |
| 2 (пройден успешно) | ‘а’ | 1 | 1 |
| 3 (пройден успешно) | ‘’’’ | 1 | 1 |
| 4 (пройден успешно) | ‘ | 0 | 0 |
| 5 (пройден успешно) | ‘’’ | 0 | 0 |
| 6 (пройден успешно) | aaa | 0 | 0 |
| 7 (пройден успешно) | aaa’’ | 0 | 0 |
| 8 (пройден успешно) | ‘aaa’’aaaa’ | 1 | 1 |
| 9 (пройден успешно) | ‘aa’’aa’’aa’’aa’ | 1 | 1 |
| 10 (пройден успешно) | a’ | 0 | 0 |
| 11 (пройден успешно) | ‘a’’ | 0 | 0 |
| 12 (пройден успешно) | stop | - | - |
| 13 (пройден успешно) | ‘aaaaa’ a | 0 | 0 |
| 14 (пройден успешно) | ‘\*aaaaa’ ‘aaa’ | 0 | 0 |
| 15 (пройден успешно) | ‘aaa’ | 0 | 0 |

## 6 Текст программы

#include <iostream>

#include <locale.h>

#include <string>

#include <set>

using namespace std;

enum TCharType {

S,

A

};

enum TState {

State0 = 0,

State1 = 1,

State2 = 2,

State3 = 3,

State4 = 4

};

const TState Transitions[5][2] = {

{State0, State0},

{State0, State2},

{State2, State3},

{State0, State4},

{State2, State3}

};

const bool IsFinalState[5] = {

false, false, false, true, false

};

int GetCharType(char C) {

if (C == '\'') {

return 1;

}

else {

return 0;

}

};

bool CheckString(const std::string Line) {

TState State = State1;

std::size\_t size = Line.size();

for (int i = 0; i <= size-1; ++i) {

State = Transitions[State][GetCharType(Line[i])];

};

return IsFinalState[State];

};

int main()

{

std::string Line;

setlocale(LC\_ALL, "Russian");

do {

std::cout << "Введите строку или stop, если хотите закончить:" << std::endl;

getline(cin, Line);

std::cout << std::endl;

if (Line != "stop") {

if (CheckString(Line)) {

std:cout << "Введённая строка является строковым литералом в языке Pascal" << std::endl << std::endl;

}

else {

std::cout << "Введённая строка НЕ является строковым литералом в Pascal" << std::endl << std::endl;

}

std::cout << "Подстроки-литералы:" << std::endl;

std::size\_t size = Line.size();

//bool flag\_Ap = 0;

bool flag\_N = 0;

std::set <std::string> tempSet;

tempSet.insert(Line);

for (int i = 0; i <= size - 1; ++i) {

for (int k = size - i; k >= 3; --k) {

for (int j = 2; j <= size - i - k + 3; ++j) {

std::string subLine = Line.substr(i, j);

if (CheckString(subLine)) {

if (tempSet.find(subLine) == tempSet.end()) {

std::cout << subLine << std::endl;

tempSet.insert(subLine);

flag\_N = 1;

}

}

}

}

}

if (not flag\_N) {

std::cout << "Среди подстрок литералов не обнаружено" << std::endl << std::endl;

}

else { std::cout << std::endl; };

}

} while (Line != "stop");

return 0;

}