МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт информационных технологий |
| Кафедра математического и программного обеспечения |
| Теория автоматов и формальных языков |
|  |

ЛАБОРАТОРНАЯ РАБОТА №3-4

«Регулярные множества и регулярные выражения»

|  |  |
| --- | --- |
| Исполнитель: | студент  группы 1ПИб-02-3оп-23  Богданов  Ренат Алексеевич |
| Руководитель: | Ганичева  Оксана Георгиевна |
| Оценка: |  |
| Подпись: |  |

2024 год

Задание

1. Выполнить задания из учебника Карпов Ю.Г. «Теория автоматов»
2. Стр.183 задание 1. Под буквами е, ж,

Задание 2. Под буквами а, б, в, г.

1. Стр. 185 задания 19, 21
2. Выполнить задание по вариантам.
3. Построить регулярное выражение, которое определяет язык, соответствующий вашему варианту (см таблицу, вариант по списку группы). Для заданного регулярного выражения построить детерминированный конечный автомат. На основе конечного автомата написать программу для распознавания строк, принадлежащих языку, определяемому регулярным выражением.

|  |  |  |
| --- | --- | --- |
| **Вариант** | **Алфавит языка** | **Описание языка** |
| 19 | a, b, с | Множество слов, в которых за подстрокой аbb следует подстрока сс |

**Часть 1**

Задание 1

Построить конечный автомат с входным алфавитом V = {a, b}, распознающий:

Е) все цепочки, заканчивающиеся кодом aabba (см. рисунок 1);

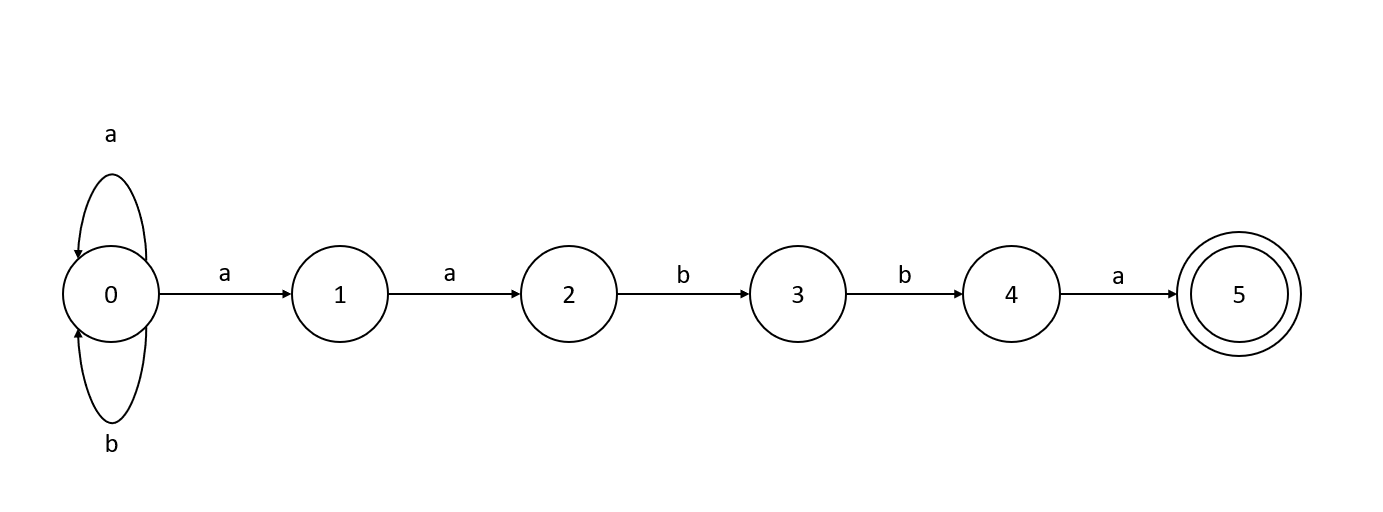


Рис. 1. Автомат 1-Е

Ж) все цепочки, в которых за каждым a непосредственно следует b (см. рисунок 2).

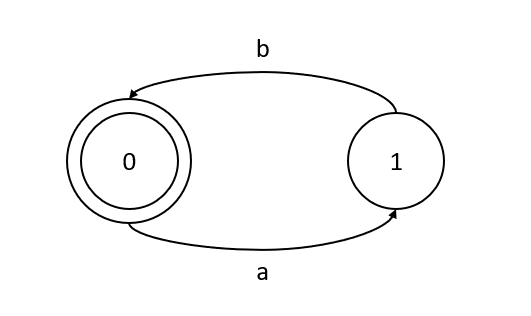


Рис. 2. Автомат 1-Ж

Задание 2

Построить конечный автомат с входным алфавитом V = {a, b, c}, распознающий:

А) все цепочки, в которых за каждым a когда-нибудь в будущем следует b (см. рис. 3);

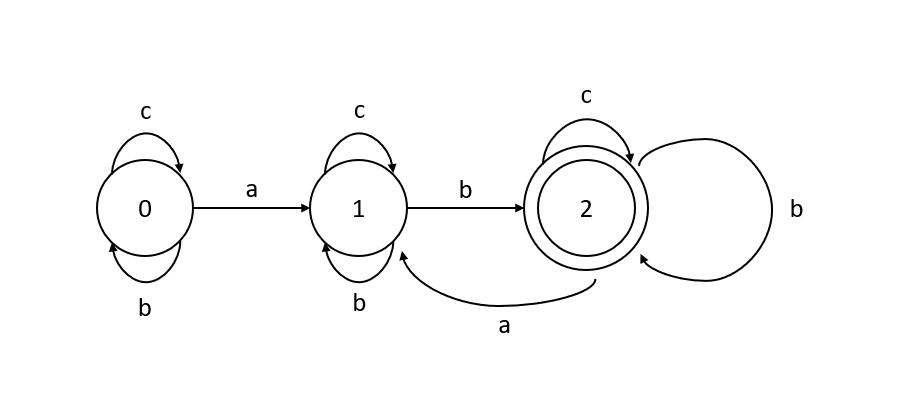


Рис. 3. Автомат 2-А

Б) все цепочки, в которых две последние буквы не совпадают (см. рис. 4);

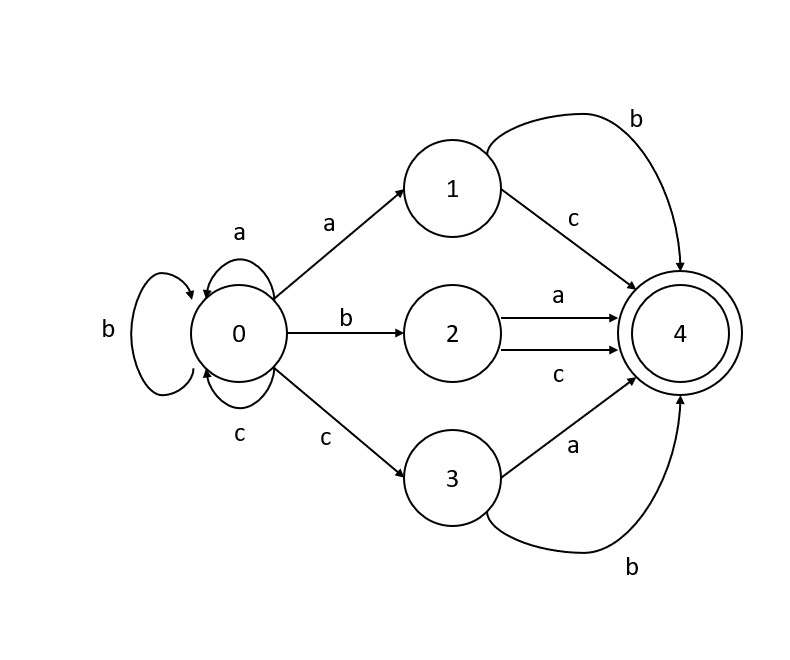


Рис. 4. Автомат 2-Б

В) все цепочки, начинающиеся и заканчивающиеся различными символами (см. рис. 5);

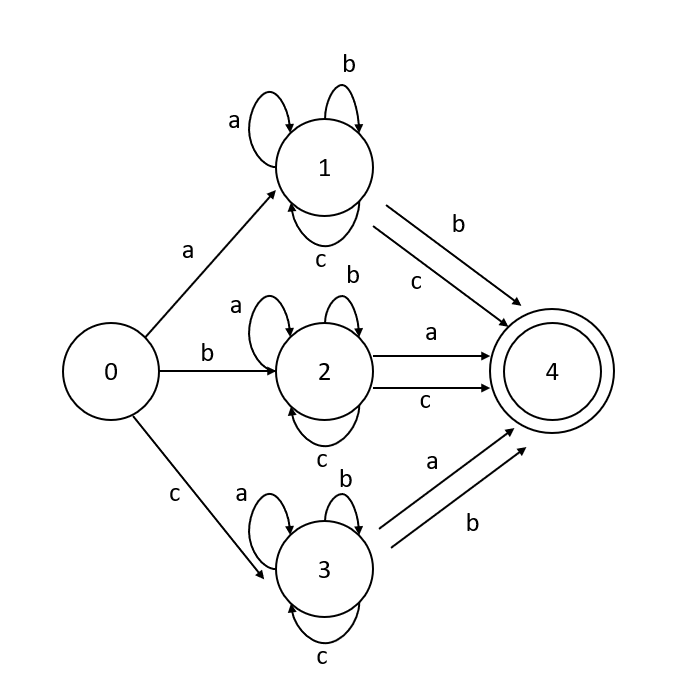


Рис. 5. Автомат 2-В

Г) все цепочки, включающие по крайней мере один символ a и один символ b (см. рис. 6).

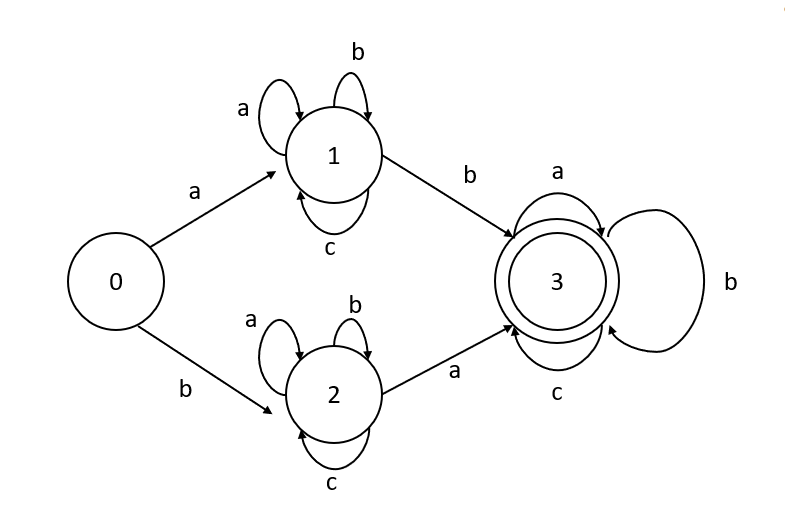


Рис. 6. Автомат 2-Г

Задание 19

Построить автоматы, распознающие языки, задаваемые регулярным выражениям:

1. a\*b\* (см. рис. 7)

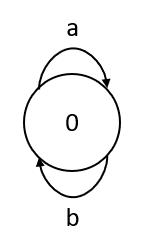


Рис. 7. Автомат 19-1

1. a\*a\* (см. рис. 8)

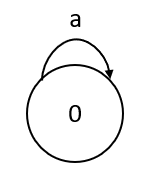


Рис. 8. Автомат 19-2

1. a\*+b\* (см. рис. 9)

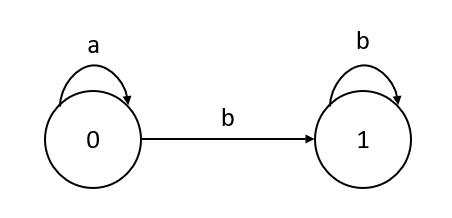


Рис. 9. Автомат 19-3

1. (a+b)\* (см. рис. 10)

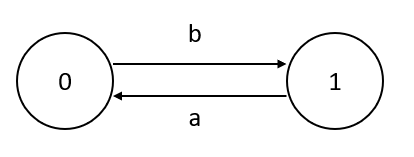


Рис. 10. Автомат 19-4

1. (a\*b\*)\* (см. рис. 11)

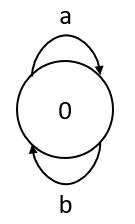


Рис. 11. Автомат 19-5

1. (a\*+b\*)\* (см. рис. 12)

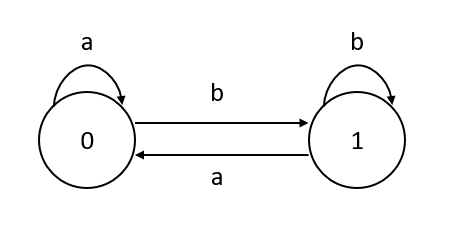


Рис. 12. Автомат 19-6

1. a\*b+b\*a (см. рис. 13)

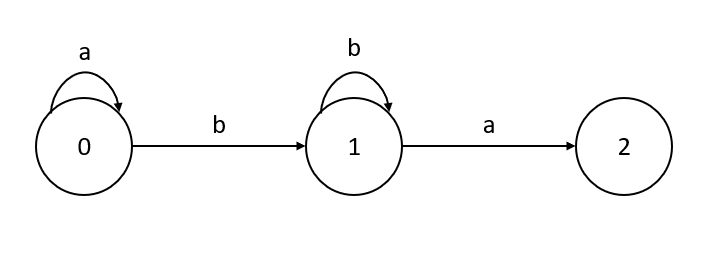


Рис. 13. Автомат 19-7

Задание 21

Построить регулярное выражение, задающее множество всех таких слов над словарем {a,b,c}, в которых за символом b:

а) обязательно стоит символ с

б) не может стоять символ с

Построить конечные автоматы, распознающие соответствующие языки.

Пункт А

Регулярное выражение – (a\*c\*(bc)\*a\*c\*)\* (см. рис. 12)

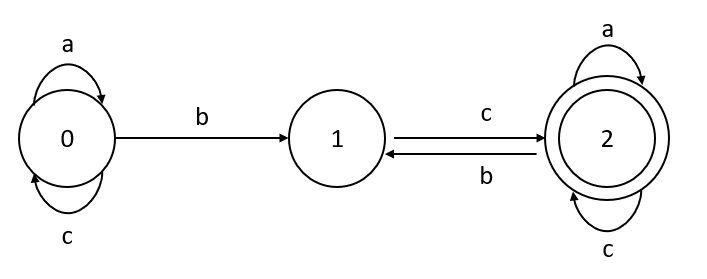


Рис. 12. Автомат 21-а

Регулярное выражение – a\*c\*ba\*b\* (см. рис. 13)

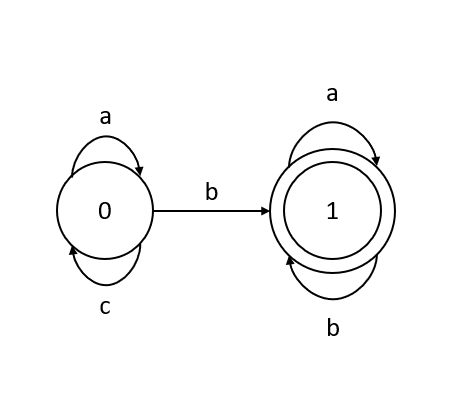


Рис. 13. Автомат 21-б

**Часть 2**

Алфавит языка: V = {a,b,c}

Описание языка: множество слов, в которых за подстрокой “аbb” следует подстрока “сс”.

Регулярное выражение: (a+b+c)\*abb(a+b+c)\*cc(a+b+c)\*

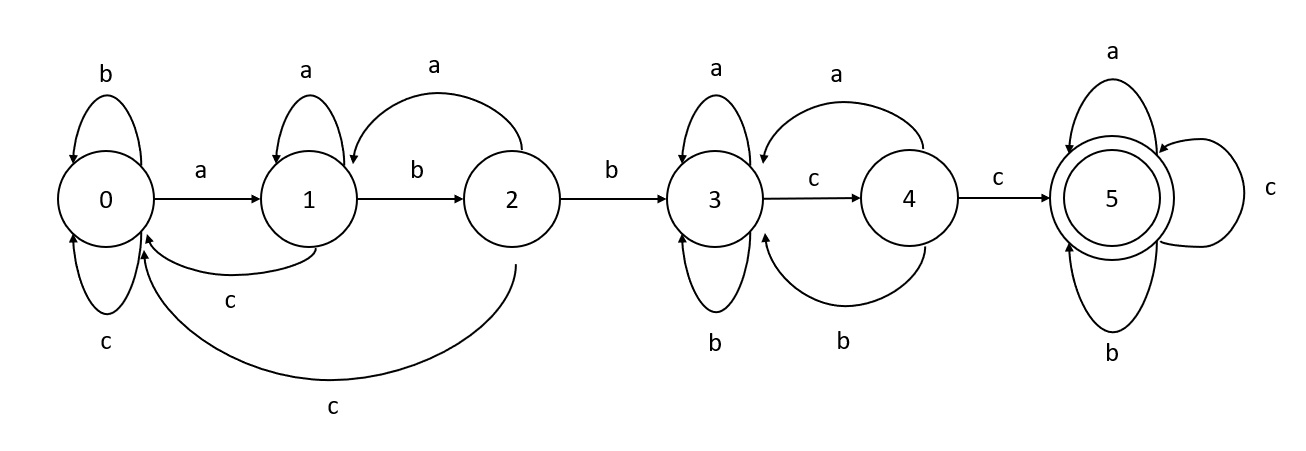


Рис. 14. Конечный автомат

Формальное описание модели: Автомат моделирует последовательность символов из алфавита {a, b, c} и распознает слова, в которых за подстрокой "abb" обязательно следует подстрока "сс". Допустимо наличие произвольного слова между подстроками "abb" и "сс". Внутреннее состояние автомата меняется в зависимости от входных символов, а выходные сигналы отражают принадлежность слова к языку. Автомат поддерживает состояния для принятия символов и проверки наличия необходимых подстрок. В программе предусмотрены как автоматический режим, где проверяется множество строк, так и ручной режим, где проверяется одно слово шаг за шагом. Состояния автомата включают начальное состояние S0 и конечное состояние S5, через которые проходит автомат при правильном распознавании слова. Входные символы 'a', 'b', 'c' обрабатываются по определенным переходам, которые обеспечивают корректное распознавание слов согласно заданным условиям.

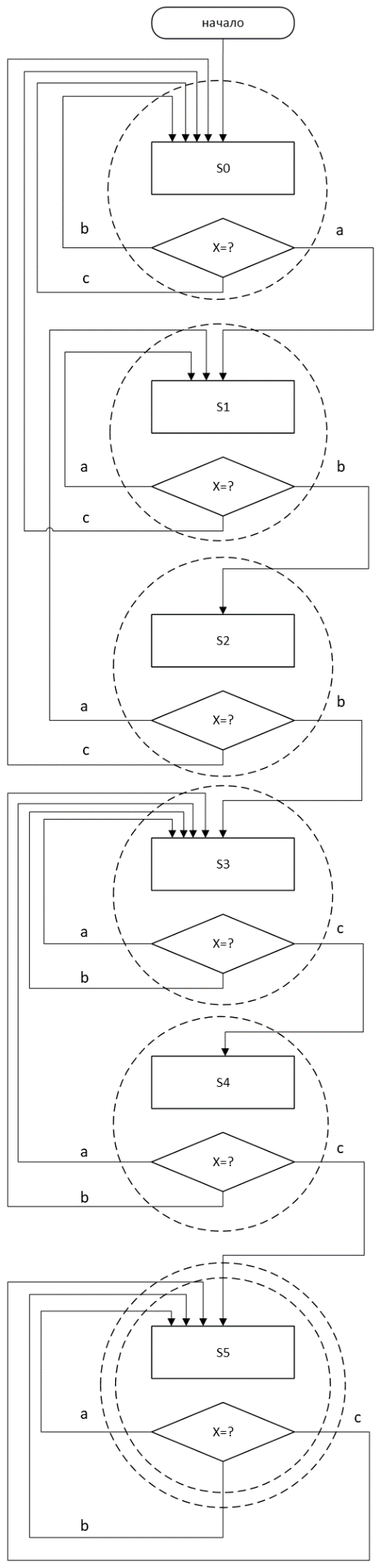


Рис. 15. Структурная схема автомата

Программное моделирование

Код программы

CppCliWinForm.cpp

#include "pch.h"

#include "Form1.h"

using namespace CppCliWinForm;

[System::STAThreadAttribute]

int main(array<System::String^>^ args)

{

// Enabling Windows XP visual effects before any controls are created

System::Windows::Forms::Application::EnableVisualStyles();

System::Windows::Forms::Application::SetCompatibleTextRenderingDefault(false);

// Create the main window and run it

System::Windows::Forms::Application::Run(gcnew Form1());

return 0;

}

Form1.h

#pragma once

namespace CppCliWinForm {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Summary for Form1

/// </summary>

public ref class Form1 : public System::Windows::Forms::Form

{

public:

enum class State { S0, S1, S2, S3, S4, S5 };

State transition(State currentState, char input) {

switch (currentState) {

case State::S0:

switch (input) {

case 'a': return State::S1;

case 'b': return State::S2;

case 'c': return State::S3;

default: return currentState;

}

case State::S1:

switch (input) {

case 'a': return State::S1;

case 'b': return State::S2;

case 'c': return State::S0;

default: return currentState;

}

case State::S2:

switch (input) {

case 'a': return State::S1;

case 'b': return State::S3;

case 'c': return State::S0;

default: return currentState;

}

case State::S3:

switch (input) {

case 'a': return State::S3;

case 'b': return State::S3;

case 'c': return State::S4;

default: return currentState;

}

case State::S4:

switch (input) {

case 'a': return State::S3;

case 'b': return State::S3;

case 'c': return State::S5;

default: return currentState;

}

case State::S5:

switch (input) {

case 'a': return State::S5;

case 'b': return State::S5;

case 'c': return State::S5;

default: return currentState;

}

default:

return currentState;

}

}

Form1(void)

{

InitializeComponent();

CurrentState = State::S0;

}

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

}

private: System::Windows::Forms::Label^ label1;

private: System::Windows::Forms::RichTextBox^ richTextBox1;

private: System::Windows::Forms::PictureBox^ pictureBox1;

private: System::Windows::Forms::TextBox^ textBox1;

private: System::Windows::Forms::Button^ button1;

private: System::Windows::Forms::Label^ label2;

private: System::Windows::Forms::Label^ label3;

private: System::Windows::Forms::Label^ label4;

private: System::Windows::Forms::DataGridView^ dataGridView1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Номер;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ Строка;

private: System::Windows::Forms::DataGridView^ dataGridView2;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ dataGridViewTextBoxColumn1;

private: System::Windows::Forms::DataGridViewTextBoxColumn^ dataGridViewTextBoxColumn2;

private: System::Windows::Forms::Label^ label5;

private: System::Windows::Forms::Label^ label6;

private: System::Windows::Forms::TextBox^ textBox2;

private: System::Windows::Forms::Button^ button2;

private: System::ComponentModel::Container^ components;

private: State CurrentState;

#pragma region Windows Form Designer generated code

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(Form1::typeid));

this->label1 = (gcnew System::Windows::Forms::Label());

this->richTextBox1 = (gcnew System::Windows::Forms::RichTextBox());

this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());

this->textBox1 = (gcnew System::Windows::Forms::TextBox());

this->button1 = (gcnew System::Windows::Forms::Button());

this->label2 = (gcnew System::Windows::Forms::Label());

this->label3 = (gcnew System::Windows::Forms::Label());

this->label4 = (gcnew System::Windows::Forms::Label());

this->dataGridView1 = (gcnew System::Windows::Forms::DataGridView());

this->Номер = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->Строка = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->dataGridView2 = (gcnew System::Windows::Forms::DataGridView());

this->dataGridViewTextBoxColumn1 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->dataGridViewTextBoxColumn2 = (gcnew System::Windows::Forms::DataGridViewTextBoxColumn());

this->label5 = (gcnew System::Windows::Forms::Label());

this->label6 = (gcnew System::Windows::Forms::Label());

this->textBox2 = (gcnew System::Windows::Forms::TextBox());

this->button2 = (gcnew System::Windows::Forms::Button());

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView2))->BeginInit();

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20.25F, System::Drawing::FontStyle::Bold, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label1->Location = System::Drawing::Point(13, 13);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(265, 31);

this->label1->TabIndex = 0;

this->label1->Text = L"Конечный автомат";

//

// richTextBox1

//

this->richTextBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 12, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->richTextBox1->Location = System::Drawing::Point(348, 342);

this->richTextBox1->Name = L"richTextBox1";

this->richTextBox1->ReadOnly = true;

this->richTextBox1->Size = System::Drawing::Size(380, 253);

this->richTextBox1->TabIndex = 1;

this->richTextBox1->Text = L"";

//

// pictureBox1

//

this->pictureBox1->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.Image")));

this->pictureBox1->InitialImage = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox1.InitialImage")));

this->pictureBox1->Location = System::Drawing::Point(19, 48);

this->pictureBox1->Name = L"pictureBox1";

this->pictureBox1->Size = System::Drawing::Size(709, 255);

this->pictureBox1->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->pictureBox1->TabIndex = 2;

this->pictureBox1->TabStop = false;

//

// textBox1

//

this->textBox1->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox1->Location = System::Drawing::Point(19, 342);

this->textBox1->Name = L"textBox1";

this->textBox1->Size = System::Drawing::Size(316, 31);

this->textBox1->TabIndex = 3;

//

// button1

//

this->button1->AutoSizeMode = System::Windows::Forms::AutoSizeMode::GrowAndShrink;

this->button1->Location = System::Drawing::Point(19, 379);

this->button1->Name = L"button1";

this->button1->Size = System::Drawing::Size(316, 32);

this->button1->TabIndex = 4;

this->button1->Text = L"Автоматический режим";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &Form1::button1\_Click);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label2->Location = System::Drawing::Point(18, 308);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(127, 25);

this->label2->TabIndex = 6;

this->label2->Text = L"Поле ввода";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 14.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label3->Location = System::Drawing::Point(19, 461);

this->label3->Name = L"label3";

this->label3->Size = System::Drawing::Size(139, 24);

this->label3->TabIndex = 7;

this->label3->Text = L"Ручной режим";

//

// label4

//

this->label4->AutoSize = true;

this->label4->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label4->Location = System::Drawing::Point(343, 308);

this->label4->Name = L"label4";

this->label4->Size = System::Drawing::Size(142, 25);

this->label4->TabIndex = 8;

this->label4->Text = L"Информация";

//

// dataGridView1

//

this->dataGridView1->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView1->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(2) {

this->Номер,

this->Строка

});

this->dataGridView1->Location = System::Drawing::Point(735, 48);

this->dataGridView1->Name = L"dataGridView1";

this->dataGridView1->ReadOnly = true;

this->dataGridView1->Size = System::Drawing::Size(532, 255);

this->dataGridView1->TabIndex = 9;

//

// Номер

//

this->Номер->HeaderText = L"Номер";

this->Номер->Name = L"Номер";

//

// Строка

//

this->Строка->HeaderText = L"Строка";

this->Строка->Name = L"Строка";

//

// dataGridView2

//

this->dataGridView2->ColumnHeadersHeightSizeMode = System::Windows::Forms::DataGridViewColumnHeadersHeightSizeMode::AutoSize;

this->dataGridView2->Columns->AddRange(gcnew cli::array< System::Windows::Forms::DataGridViewColumn^ >(2) {

this->dataGridViewTextBoxColumn1,

this->dataGridViewTextBoxColumn2

});

this->dataGridView2->Location = System::Drawing::Point(735, 342);

this->dataGridView2->Name = L"dataGridView2";

this->dataGridView2->ReadOnly = true;

this->dataGridView2->Size = System::Drawing::Size(532, 255);

this->dataGridView2->TabIndex = 10;

//

// dataGridViewTextBoxColumn1

//

this->dataGridViewTextBoxColumn1->HeaderText = L"Номер";

this->dataGridViewTextBoxColumn1->Name = L"dataGridViewTextBoxColumn1";

//

// dataGridViewTextBoxColumn2

//

this->dataGridViewTextBoxColumn2;

this->dataGridViewTextBoxColumn2->Name = L"dataGridViewTextBoxColumn2";

//

// label5

//

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label5->Location = System::Drawing::Point(730, 20);

this->label5->Name = L"label5";

this->label5->Size = System::Drawing::Size(210, 25);

this->label5->TabIndex = 11;

this->label5->Text = L"Правильные строки";

//

// label6

//

this->label6->AutoSize = true;

this->label6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->label6->Location = System::Drawing::Point(730, 308);

this->label6->Name = L"label6";

this->label6->Size = System::Drawing::Size(228, 25);

this->label6->TabIndex = 12;

this->label6->Text = L"Неправильные строки";

//

// textBox2

//

this->textBox2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 15.75F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

static\_cast<System::Byte>(204)));

this->textBox2->Location = System::Drawing::Point(19, 488);

this->textBox2->Name = L"textBox2";

this->textBox2->Size = System::Drawing::Size(316, 31);

this->textBox2->TabIndex = 13;

//

// button2

//

this->button2->Location = System::Drawing::Point(19, 526);

this->button2->Name = L"button2";

this->button2->Size = System::Drawing::Size(316, 32);

this->button2->TabIndex = 14;

this->button2->Text = L"Начать разбор слова";

this->button2->UseVisualStyleBackColor = true;

this->button2->Click += gcnew System::EventHandler(this, &Form1::button2\_Click);

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(1296, 607);

this->Controls->Add(this->button2);

this->Controls->Add(this->textBox2);

this->Controls->Add(this->label6);

this->Controls->Add(this->label5);

this->Controls->Add(this->dataGridView2);

this->Controls->Add(this->dataGridView1);

this->Controls->Add(this->label4);

this->Controls->Add(this->label3);

this->Controls->Add(this->label2);

this->Controls->Add(this->button1);

this->Controls->Add(this->textBox1);

this->Controls->Add(this->pictureBox1);

this->Controls->Add(this->richTextBox1);

this->Controls->Add(this->label1);

this->Name = L"Form1";

this->Text = L"Конечный автомат";

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView1))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->dataGridView2))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private: System::Void button1\_Click(System::Object^ sender, System::EventArgs^ e) {

dataGridView1->Rows->Clear();

dataGridView2->Rows->Clear();

array<Char>^ delimiters = { ' ', '\n' };

array<String^>^ lines = textBox1->Text->Split(delimiters, StringSplitOptions::RemoveEmptyEntries);

int correctCount = 1;

int incorrectCount = 1;

for each (String ^ line in lines) {

State currentState = State::S0;

bool isValid = true;

for each (char c in line) {

if (c != 'a' && c != 'b' && c != 'c') {

isValid = false;

break;

}

currentState = transition(currentState, c);

}

if (currentState == State::S5 && isValid) {

dataGridView1->Rows->Add(correctCount++, line);

}

else {

dataGridView2->Rows->Add(incorrectCount++, line);

}

}

}

private: System::Void button2\_Click(System::Object^ sender, System::EventArgs^ e) {

richTextBox1->Clear();

State currentState = State::S0;

String^ input = textBox2->Text;

for each (char c in input) {

String^ currentCharString = gcnew String(c, 1); // Создаем строку из одного символа

richTextBox1->AppendText("Текущее состояние: " + currentState.ToString() + "\n");

richTextBox1->AppendText("Текущий символ: '" + currentCharString + "'\n");

if (c != 'a' && c != 'b' && c != 'c') {

richTextBox1->AppendText("Недопустимый символ '" + currentCharString + "'\n");

return;

}

State nextState = transition(currentState, c);

richTextBox1->AppendText("Состояние " + currentState.ToString() + " -> Ввод '" + currentCharString + "' -> Состояние " + nextState.ToString() + "\n");

currentState = nextState;

}

if (currentState == State::S5) {

richTextBox1->AppendText("Слово принято.\n");

}

else {

richTextBox1->AppendText("Слово отвергнуто.\n");

}

}

};

}

Примеры работы программы

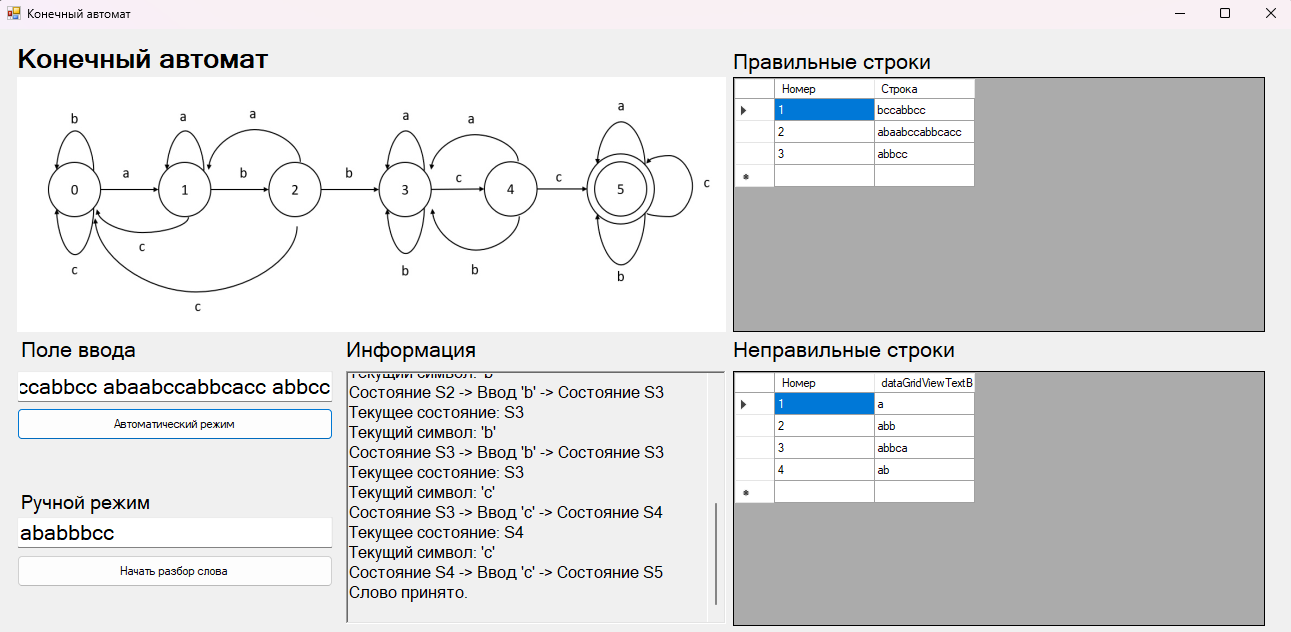


Рис. 16. Интерфейс программы, пример работы в автоматическом и ручном режимах

Вывод

В ходе выполнения лабораторной работы была успешно реализована программа для распознавания строк, принадлежащих языку, определяемому регулярным выражением (a+b+c)\*abb(a+b+c)cc(a+b+c). Построены и проанализированы конечные автоматы для различных языков, заданных в учебнике, что позволило глубже понять принципы работы конечных автоматов и регулярных выражений. Реализованный интерфейс программы предоставляет возможность проверять строки как в автоматическом, так и в ручном режимах, демонстрируя текущее состояние автомата и принимаемые решения на каждом шаге. Полученные результаты подтверждают корректность работы программы и соответствуют теоретическим ожиданиям.