МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

*наименование института(факультета)*

Кафедра математического и программного обеспечения ЭВМ

*наименование кафедры*

Теория автоматов и формальных языков

*наименование дисциплины в соответствии с учебным планом*

ЛАБОРАТОРНАЯ РАБОТА №3-4

«**Регулярные множества и регулярные выражения**»

|  |  |
| --- | --- |
| Исполнитель |  |
| Студент | 1ПИб-02-3оп-22 |
|  | *группа* |
|  | Гумаров Д. А. |
|  | *ФИО* |
| Руководитель | Ганичева О. Г. |
|  | *ФИО преподавателя* |
| Оценка |  |
| Подпись |  |

1. Год

Задание

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.

|  |  |  |
| --- | --- | --- |
| 8 | a, b, c | Множество слов, включающих подстроку *aabba* |

Ход работы

**Регулярное выражение:** (a|b|c)\*aabba(a|b|c)\*

**Примеры строк:** aabba, aacbaabbaac, bbacaabbaba и т. д.

**Формальное описание конечного автомата:**

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2, S3, S4, S5}

S0 = {S0}

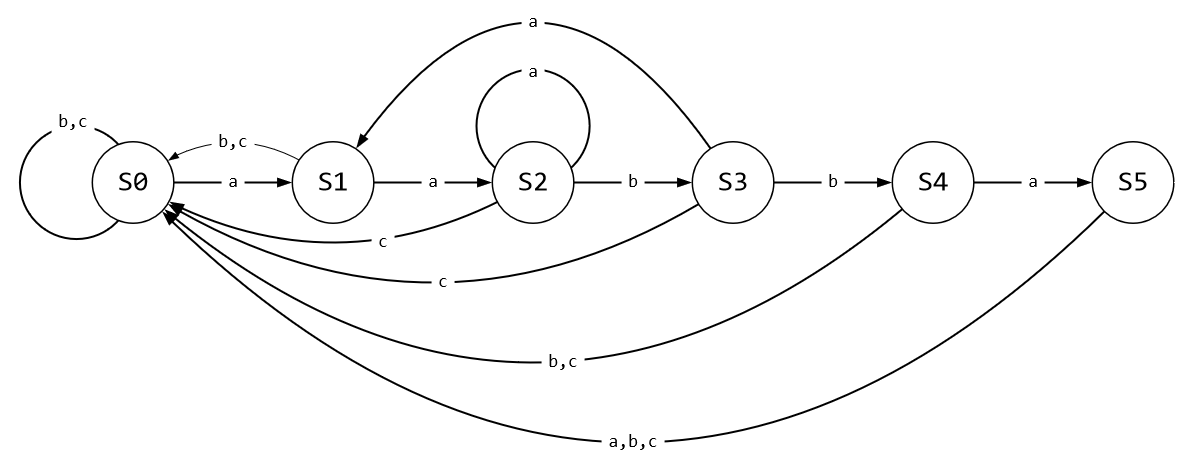
F = {S5}

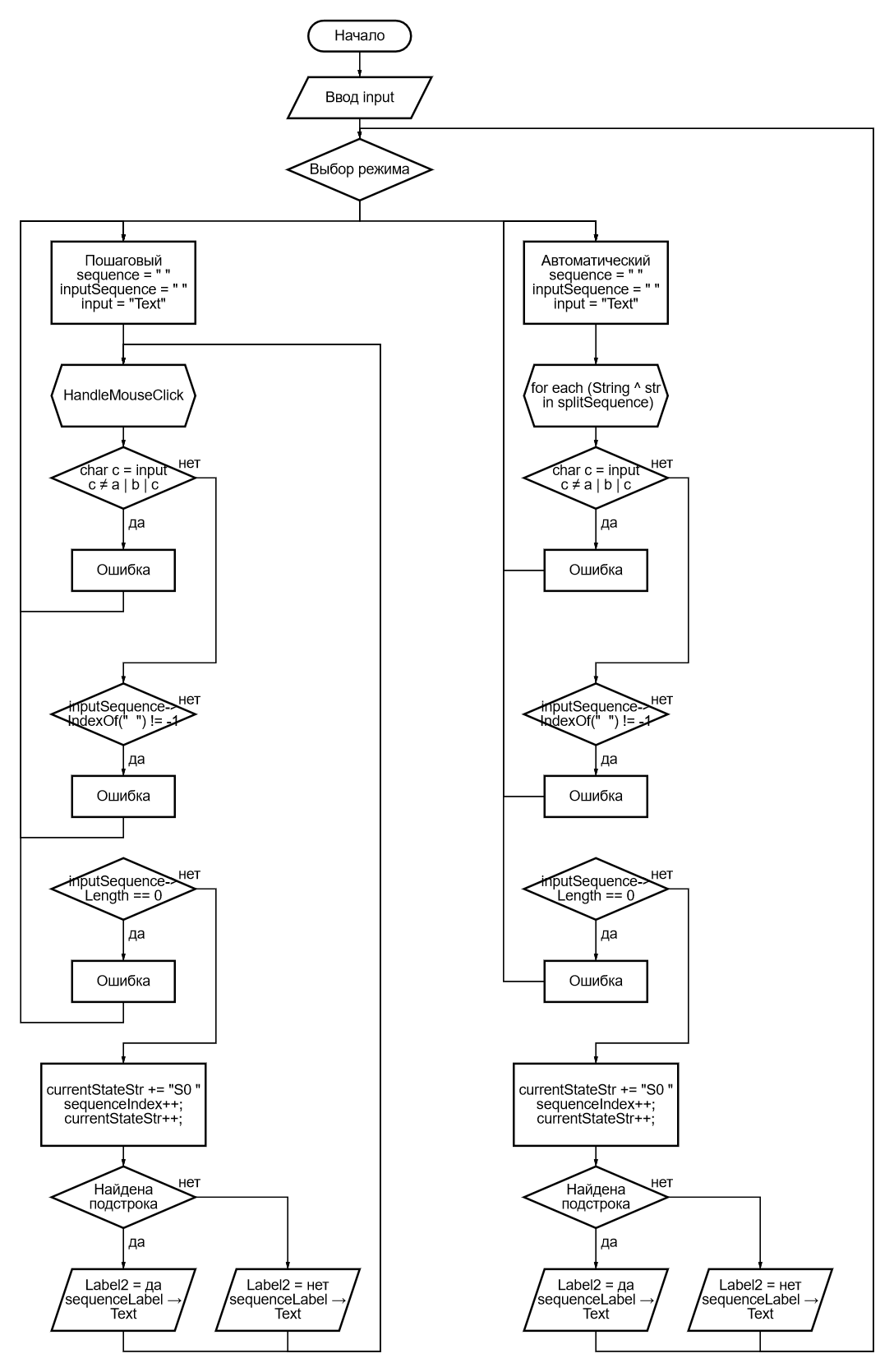
δ:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (S0, a) → S1 | (S1, a) → S2 | (S2, a) → S2 | (S3, a) → S1 | (S4, a) → S5 | (S5, a) → S0 |
| (S0, b) → S0 | (S1, b) → S0 | (S2, b) → S3 | (S3, b) → S4 | (S4, b) → S0 | (S5, b) → S0 |

(S0, c) → S0 (S1, c) → S0  (S2, c) → S0 (S3, c) → S0  (S4, c) → S0 (S5, c) → S0

Граф:



Блок схема:

Текст программы:

#pragma once

#include <iostream>

#include <string>

#include <locale>

#include <msclr/marshal\_cppstd.h>

#include <Windows.h>

namespace laba34 {

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

enum State {

S0, S1, S2, S3, S4, S5

};

State getNextState(State currentState, char input) {

switch (currentState) {

case S0:

switch (input) {

case 'a':

return S1;

case 'b':

case 'c':

return S0;

}

break;

case S1:

switch (input) {

case 'a':

return S2;

case 'b':

case 'c':

return S0;

}

break;

case S2:

switch (input) {

case 'a':

return S2;

case 'b':

return S3;

case 'c':

return S0;

}

break;

case S3:

switch (input) {

case 'a':

return S1;

case 'b':

return S4;

case 'c':

return S0;

}

break;

case S4:

switch (input) {

case 'a':

return S5;

case 'b':

case 'c':

return S0;

}

break;

case S5:

switch (input) {

case 'a':

case 'b':

case 'c':

return S0;

}

break;

}

return currentState;

}

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

{

public:

MyForm(void)

{

InitializeComponent();

}

protected:

~MyForm()

{

if (components)

{

delete components;

}

}

private:

int sequenceIndex = 0;

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

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

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

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

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

System::Windows::Forms::Label^ stateLabel;

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

System::Windows::Forms::Label^ sequenceLabel;

System::Windows::Forms::Button^ clearButton;

System::Windows::Forms::Button^ stepByStepButton;

System::ComponentModel::IContainer^ components;

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

private: System::Windows::Forms::PictureBox^ pictureBox2;

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

private: System::Windows::Forms::PictureBox^ pictureBox3;

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

String^ sequence = "S0 ";

#pragma region Windows Form Designer generated code

void InitializeComponent(void)

{

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

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

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

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

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

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

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

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

this->pictureBox2 = (gcnew System::Windows::Forms::PictureBox());

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

this->pictureBox3 = (gcnew System::Windows::Forms::PictureBox());

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

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

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox2))->BeginInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox3))->BeginInit();

this->SuspendLayout();

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(12, 9);

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

this->label1->Size = System::Drawing::Size(86, 13);

this->label1->TabIndex = 0;

this->label1->Text = L"Введите строку";

//

// textBox1

//

this->textBox1->Location = System::Drawing::Point(104, 6);

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

this->textBox1->Size = System::Drawing::Size(353, 20);

this->textBox1->TabIndex = 1;

//

// button1

//

this->button1->Location = System::Drawing::Point(104, 32);

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

this->button1->Size = System::Drawing::Size(167, 23);

this->button1->TabIndex = 2;

this->button1->Text = L"Проверить все строки";

this->button1->UseVisualStyleBackColor = true;

this->button1->Click += gcnew System::EventHandler(this, &MyForm::CheckString);

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(12, 114);

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

this->label2->Size = System::Drawing::Size(111, 13);

this->label2->TabIndex = 3;

this->label2->Text = L"Текущее состояние:";

//

// label3

//

this->label3->AutoSize = true;

this->label3->Location = System::Drawing::Point(129, 114);

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

this->label3->Size = System::Drawing::Size(0, 13);

this->label3->TabIndex = 4;

//

// stateLabel

//

this->stateLabel->AutoSize = true;

this->stateLabel->Location = System::Drawing::Point(322, 247);

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

this->stateLabel->Size = System::Drawing::Size(0, 13);

this->stateLabel->TabIndex = 5;

//

// label4

//

this->label4->AutoSize = true;

this->label4->Location = System::Drawing::Point(12, 257);

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

this->label4->Size = System::Drawing::Size(174, 13);

this->label4->TabIndex = 6;

this->label4->Text = L"Последовательность состояний:";

//

// sequenceLabel

//

this->sequenceLabel->AutoSize = true;

this->sequenceLabel->Location = System::Drawing::Point(12, 281);

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

this->sequenceLabel->Size = System::Drawing::Size(0, 13);

this->sequenceLabel->TabIndex = 7;

//

// clearButton

//

this->clearButton->Location = System::Drawing::Point(104, 61);

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

this->clearButton->Size = System::Drawing::Size(353, 23);

this->clearButton->TabIndex = 8;

this->clearButton->Text = L"Очистить";

this->clearButton->UseVisualStyleBackColor = true;

this->clearButton->Click += gcnew System::EventHandler(this, &MyForm::ClearSequence);

//

// stepByStepButton

//

this->stepByStepButton->Location = System::Drawing::Point(277, 32);

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

this->stepByStepButton->Size = System::Drawing::Size(180, 23);

this->stepByStepButton->TabIndex = 9;

this->stepByStepButton->Text = L"Проверить пошагово";

this->stepByStepButton->UseVisualStyleBackColor = true;

this->stepByStepButton->Click += gcnew System::EventHandler(this, &MyForm::StepByStepProcess);

//

// pictureBox1

//

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

this->pictureBox1->Location = System::Drawing::Point(12, 384);

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

this->pictureBox1->Size = System::Drawing::Size(622, 297);

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

this->pictureBox1->TabIndex = 10;

this->pictureBox1->TabStop = false;

//

// pictureBox2

//

this->pictureBox2->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox2.Image")));

this->pictureBox2->Location = System::Drawing::Point(640, 512);

this->pictureBox2->Name = L"pictureBox2";

this->pictureBox2->Size = System::Drawing::Size(622, 169);

this->pictureBox2->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->pictureBox2->TabIndex = 11;

this->pictureBox2->TabStop = false;

//

// label5

//

this->label5->Anchor = static\_cast<System::Windows::Forms::AnchorStyles>((((System::Windows::Forms::AnchorStyles::Top | System::Windows::Forms::AnchorStyles::Bottom)

| System::Windows::Forms::AnchorStyles::Left)

| System::Windows::Forms::AnchorStyles::Right));

this->label5->AutoSize = true;

this->label5->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 25, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label5->Location = System::Drawing::Point(934, 427);

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

this->label5->Size = System::Drawing::Size(328, 39);

this->label5->TabIndex = 12;

this->label5->Text = L"(a|b|c)\*aabba(a|b|c)\*";

//

// pictureBox3

//

this->pictureBox3->Image = (cli::safe\_cast<System::Drawing::Image^>(resources->GetObject(L"pictureBox3.Image")));

this->pictureBox3->Location = System::Drawing::Point(640, 384);

this->pictureBox3->Name = L"pictureBox3";

this->pictureBox3->Size = System::Drawing::Size(288, 122);

this->pictureBox3->SizeMode = System::Windows::Forms::PictureBoxSizeMode::StretchImage;

this->pictureBox3->TabIndex = 13;

this->pictureBox3->TabStop = false;

//

// label6

//

this->label6->AutoSize = true;

this->label6->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 25, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label6->Location = System::Drawing::Point(463, 9);

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

this->label6->Size = System::Drawing::Size(814, 39);

this->label6->TabIndex = 14;

this->label6->Text = L"Введите строки через ОДИН пробел, без запятых";

//

// MyForm

//

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

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

this->ClientSize = System::Drawing::Size(1274, 693);

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

this->Controls->Add(this->pictureBox3);

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

this->Controls->Add(this->pictureBox2);

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

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

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

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

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

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

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

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

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

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

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

this->Name = L"MyForm";

this->Text = L"Лабораторная работа №3-4";

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

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox2))->EndInit();

(cli::safe\_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox3))->EndInit();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

void ProcessSequence(String^ sequence)

{

// Очистка метки перед началом обработки новой последовательности

label2->Text = "";

array<String^>^ splitSequence = sequence->Split(' ');

for each (String ^ str in splitSequence)

{

std::string input = msclr::interop::marshal\_as<std::string>(str);

State currentState = S0;

String^ currentStateStr = "";

// Добавляем начальное состояние S0 в начало последовательности для текущей строки

currentStateStr += "S0 ";

// Добавляем заголовок для текущей строки

sequenceLabel->Text += "Строка " + str + " - ";

// Проверка наличия подстроки "aabba" в текущей строке

if (input.find("aabba") != std::string::npos) {

label2->Text += "Строка " + str + " содержит подстроку 'aabba'.\n";

}

else {

label2->Text += "Строка " + str + " не содержит подстроку 'aabba'.\n";

}

for (char& c : input) {

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

// Добавляем сообщение об ошибке в последовательность состояний для текущей строки

currentStateStr = "Ошибка: Символ '" + c + "' не принадлежит алфавиту.";

sequenceLabel->Text += currentStateStr + Environment::NewLine;

return;

}

currentState = getNextState(currentState, c);

// Обновление последовательности состояний для текущей строки

currentStateStr += "S" + static\_cast<int>(currentState).ToString() + " ";

}

// Добавляем последовательность состояний для текущей строки

sequenceLabel->Text += currentStateStr + Environment::NewLine;

}

}

void CheckString(Object^ sender, EventArgs^ e)

{

String^ inputSequence = textBox1->Text->Trim();

// Проверка на пустую строку

if (inputSequence->Length == 0)

{

MessageBox::Show("Введите последовательность строк.", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

// Проверка наличия двух пробелов между строками

if (inputSequence->IndexOf(" ") != -1)

{

MessageBox::Show("Пожалуйста, введите строки без лишних пробелов между ними.", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

// Обработка введенной последовательности строк

ProcessSequence(inputSequence);

}

void StepByStepProcess(Object^ sender, EventArgs^ e)

{

// Получение введенной последовательности строк из TextBox

String^ inputSequence = textBox1->Text->Trim();

// Проверка на пустую строку или строки, содержащие только пробелы

if (String::IsNullOrWhiteSpace(inputSequence))

{

MessageBox::Show("Введите последовательность строк.", "Ошибка", MessageBoxButtons::OK, MessageBoxIcon::Error);

return;

}

// Очистка Label с последовательностью состояний перед началом новой последовательности

sequenceLabel->Text = "";

// Разделение последовательности на отдельные строки

array<String^>^ splitSequence = inputSequence->Split(' ');

// Обработка каждой строки пошагово

if (sequenceIndex < splitSequence->Length)

{

// Очистка текущего состояния

stateLabel->Text = "";

// Обработка текущей строки

ProcessSequence(splitSequence[sequenceIndex]);

// Увеличение индекса строки для следующего шага

sequenceIndex++;

}

else

{

// Все строки обработаны, сброс индекса строки

sequenceIndex = 0;

}

// Включение кнопки для следующего шага

stepByStepButton->Enabled = true;

}

void HandleMouseClick(Object^ sender, MouseEventArgs^ e)

{

// Если происходит клик мыши, разрешаем продолжение пошаговой обработки

stepByStepButton->Enabled = true;

}

void ClearSequence(Object^ sender, EventArgs^ e)

{

// Очистка TextBox

textBox1->Text = "";

// Очистка Label

label2->Text = "";

// Очистка последовательности и обновление соответствующего Label

sequence = "";

sequenceLabel->Text = sequence;

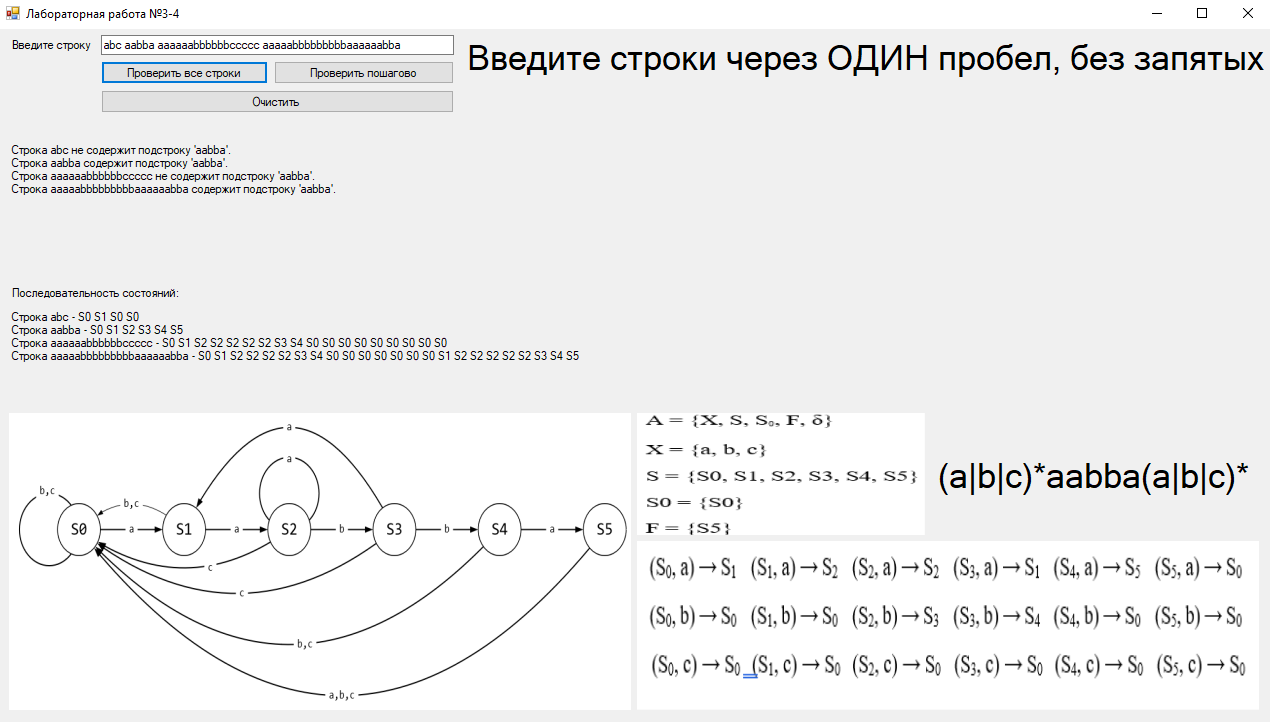
// Очистка текущего состояния и обновление соответствующего Label

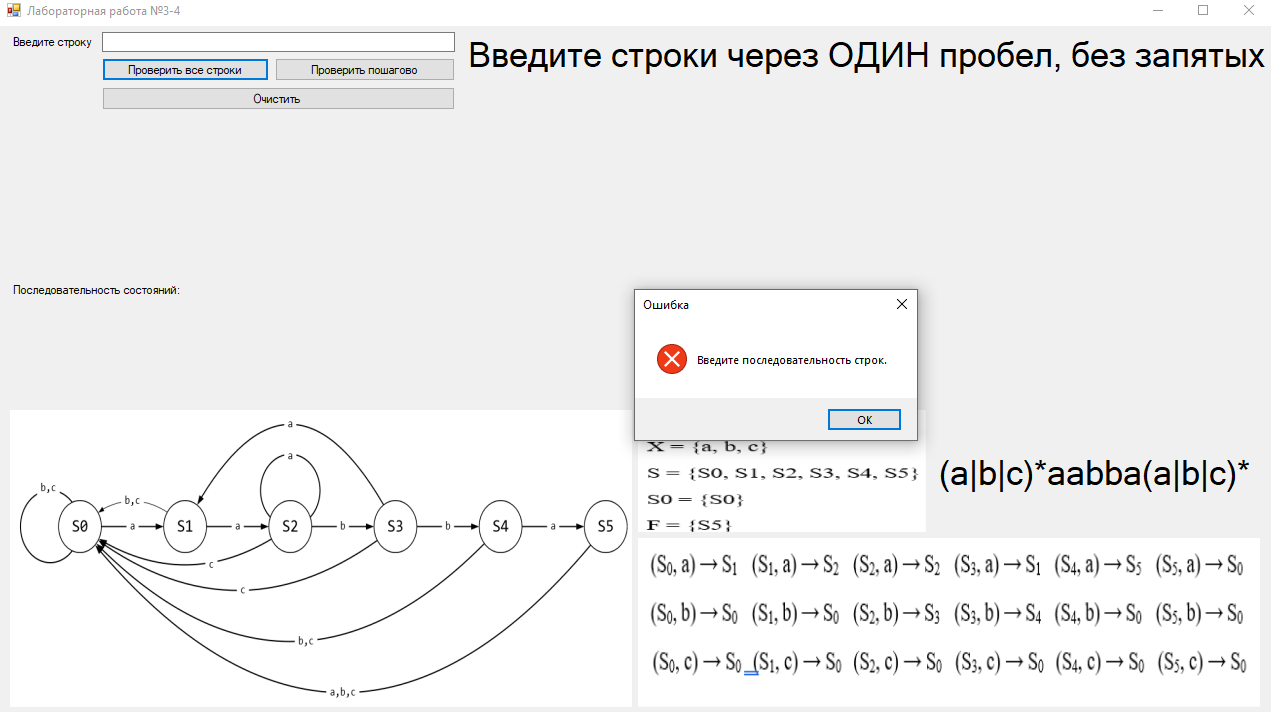
stateLabel->Text = "";

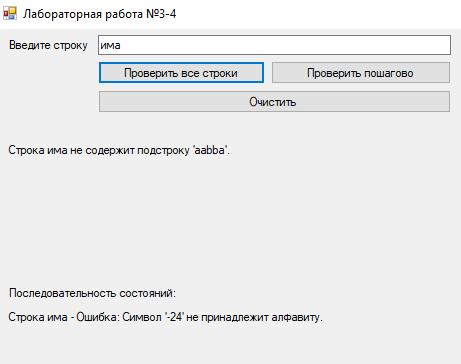
}

};

}







Вывод:

В ходе лабораторной работы смоделирована работа конечного автомата. Была разработана программа, которая имитирует работу конечного автомата, используя оператор "switch".

Характеристика построенного автомата:

1. Абстрактный

2. Конечный

3. Асинхронный

4. Распозаватель