МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Институт информационных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование института (факультета)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_МПО ЭВМ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование кафедры \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Теория автоматов и формальных языков\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование дисциплины в соответствии с учебным планом

ЛАБОРАТОРНАЯ РАБОТА № 2

Регулярные множества и регулярные выражения

Исполнитель

студент \_\_\_ПИб-02-1оп-22\_\_\_

группа

\_\_\_Микуцких Г. А.\_\_\_

Фамилия, имя, отчество

Руководитель \_\_\_Ганичева О. Г.\_\_\_\_

Ф.И.О. преподавателя

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_2023\_\_ год

Ход работы:

1. Формальная модель конечного автомата

Моделью КА называют пятёрку компонентов вида: *δ*, где:

* X – конечное непустое множество входных сигналов (входной алфавит);
* S – конечное непустое множество состояний;
* F – конечное непустое множество выделенных заключительных состояний F⊆S;
* s0 ∈ S – начальное состояние;
* δ: X×S→S – функция переходов.

Стр.183 задание 1 под буквами е), ж); задание 2 под буквами а), б), в), г).

Задание 1.1. Построить конечный автомат с входным алфавитом V = {a, b}, распознающий все цепочки, заканчивающиеся кодом aabba.

, X = {a, b}, S = {0, 1, 2, 3, 4, 5}, F = {5}, δ: X×S→S.

Формальная модель КА 1.1 представлена в виде ор. графа на рис. 1.

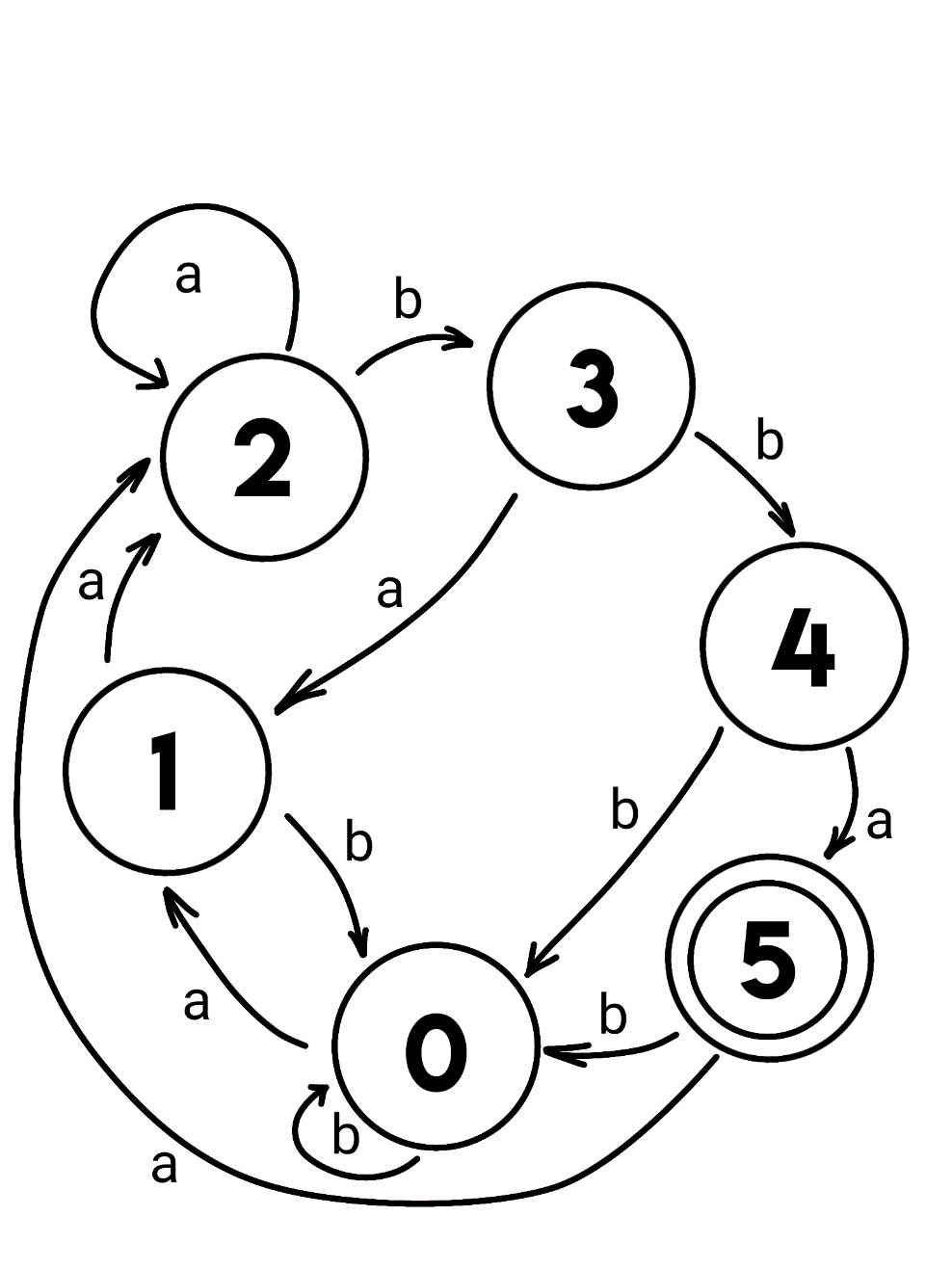


Рис. 1. Граф конечного автомата 1.1

Цепочки-примеры: aabba, aabaabba, aabbaaba, abba, aaba, abaabbaabbb, ba, abbaab, aaa, bbbbb, bbaaabba, abbaaabba, aaabbaabba.

Задание 1.2. Построить конечный автомат с входным алфавитом V = {a, b}, распознающий все цепочки, в которых за каждым a непосредственно следует b.

, X = {a, b}, S = {0, 1, 2, 3}, F = {2}, δ: X×S→S.

Формальная модель КА 1.2 представлена в виде ор. графа на рис. 2.

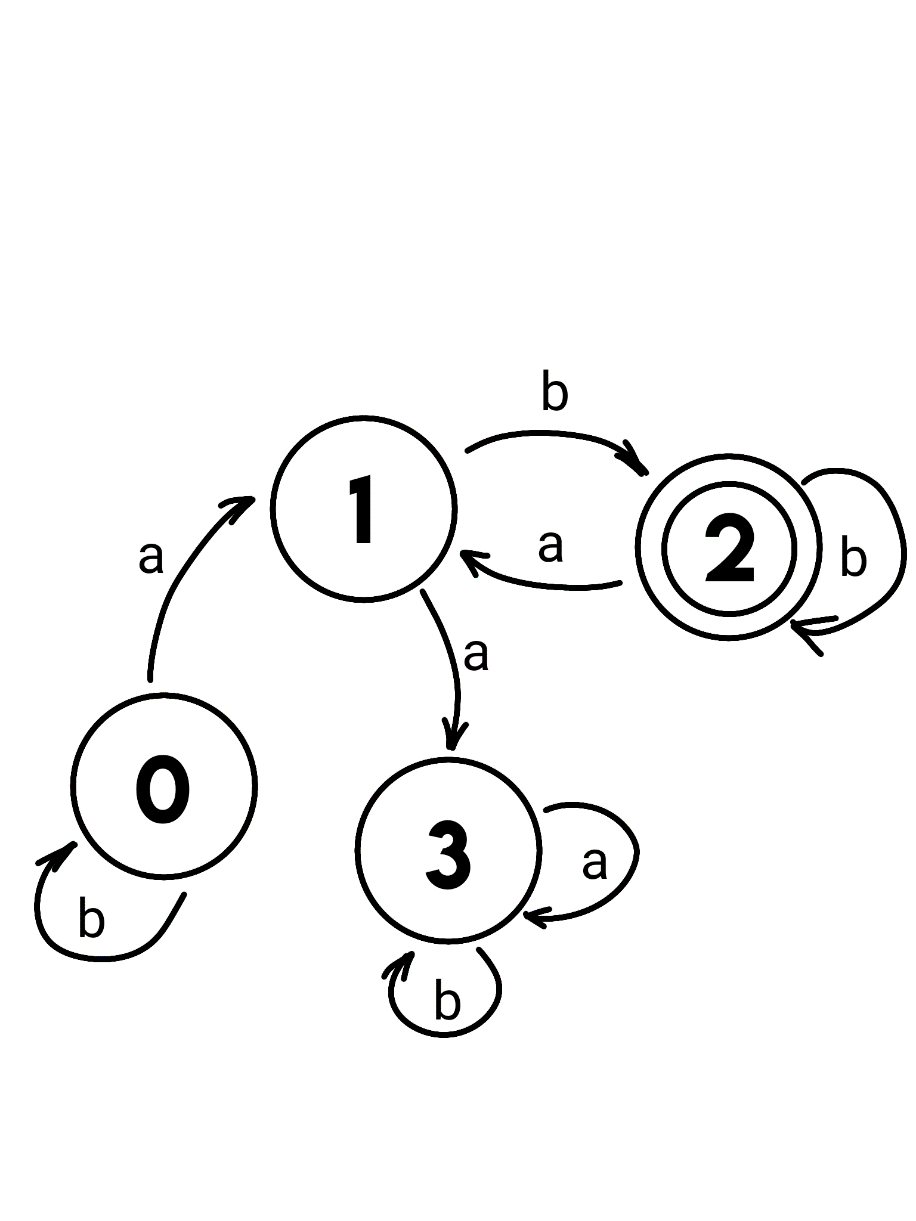


Рис. 2. Граф конечного автомата 1.2

Цепочки-примеры: abaa, ababab, a, ab, bbbab, bbba, baab, abbbab, abba, abbabbabbbb.

Задание 2.1. Построить конечный автомат с входным алфавитом V = {a,b,c}, распознающий все цепочки, в которых за каждым a когда-нибудь в будущем следует b.

, X = {a, b, c}, S = {0, 1, 2, 3}, F = {2}, δ: X×S→S.

Формальная модель КА 2.1 представлена в виде ор. графа на рис. 3.

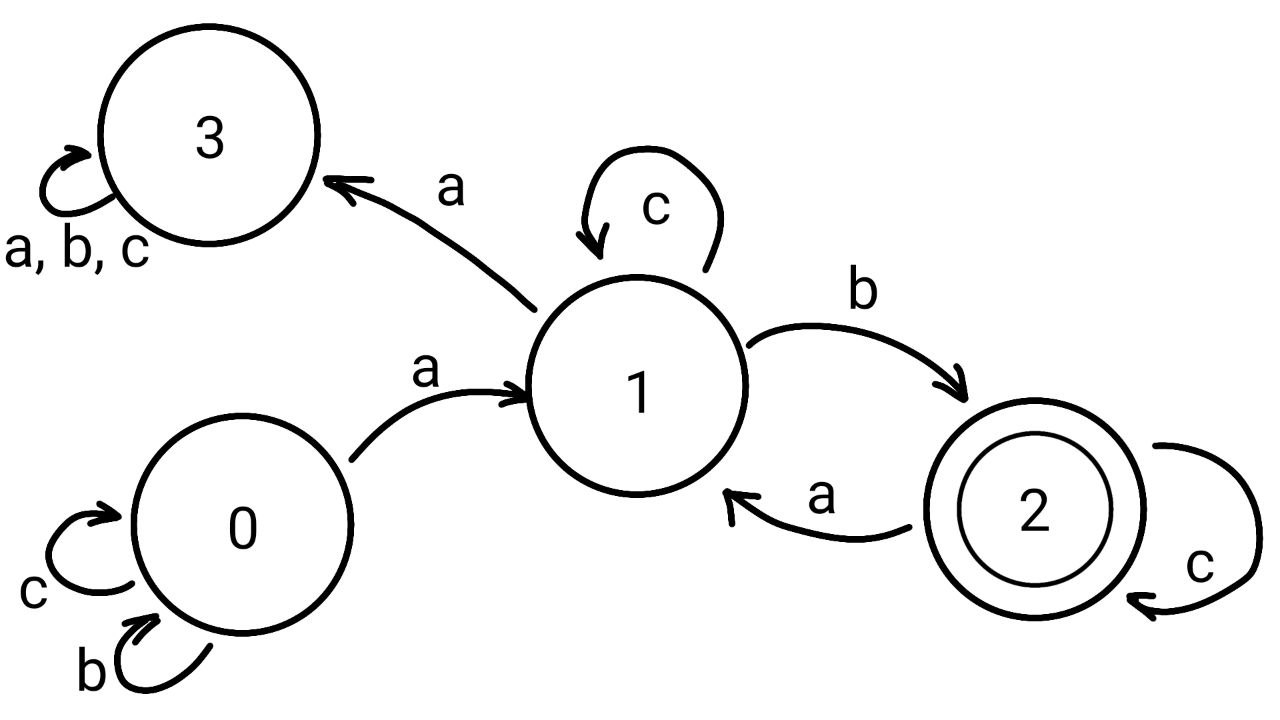


Рис. 3. Граф конечного автомата 2.1

Цепочки-примеры: ab, abac, ababa, acccccbc, cbbbcbcaccb, acbb.

Задание 2.2. Построить конечный автомат с входным алфавитом V = {a,b,c}, распознающий все цепочки, в которых две последние буквы не совпадают.

, X = {a, b, c}, S = {0, 1, 2, 3, 4, 5, 6}, F = {1, 3, 5}, δ: X×S→S. Формальная модель КА 2.2 представлена в виде ор. графа на рис. 4.

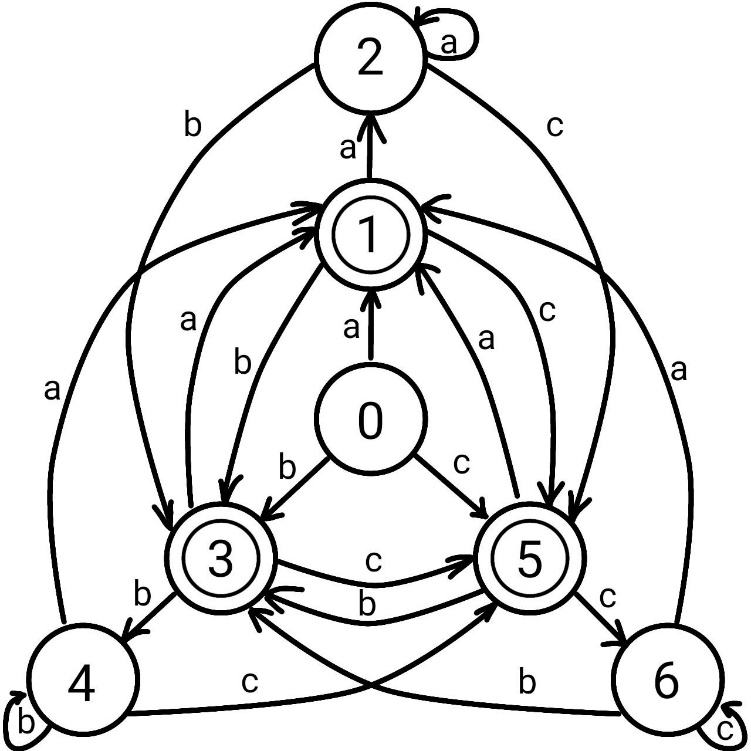


Рис. 4. Граф конечного автомата 2.2

Цепочки-примеры: a, abcc, cca, abcba, aabcbbc, cccccccc, abbb.

Задание 2.3. Построить конечный автомат с входным алфавитом V = {a,b,c}, распознающий все цепочки, начинающиеся и заканчивающиеся различными символами.

, X = {a,b,c}, S = {0, 1, 2, 3, 4, 5, 6}, F = {2, 4, 6}, δ: X×S→S. Формальная модель КА 2.3 представлена в виде ор. графа на рис. 5.

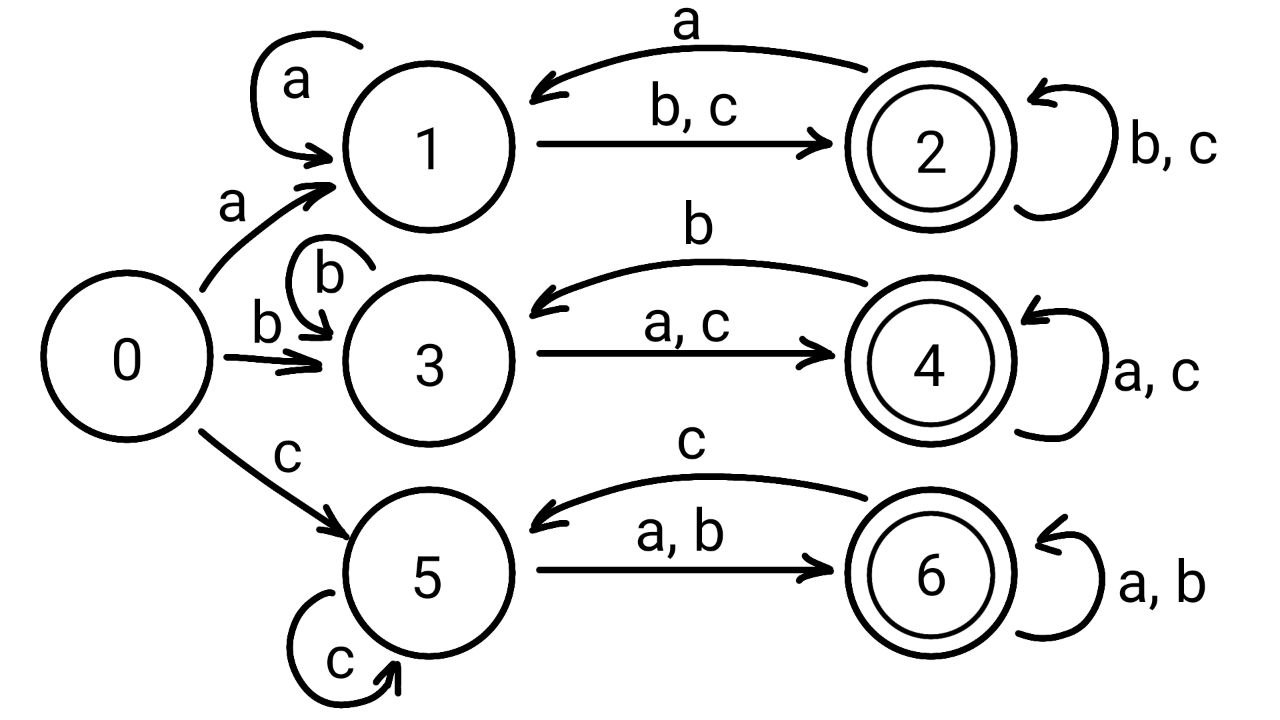


Рис. 5. Граф конечного автомата 2.3

Цепочки-примеры: abcbcaba, bca, abcc, cabc, cbcbcb, abccab.

Задание 2.4. Построить конечный автомат с входным алфавитом V = {a,b,c}, распознающий все цепочки, включающие по крайней мере один символ a и один символ b.

, X = {a, b, c}, S = {0, 1, 2}, F = {2}, δ: X×S→S. Формальная модель КА 2.4 представлена в виде ори. графа на рис. 6.

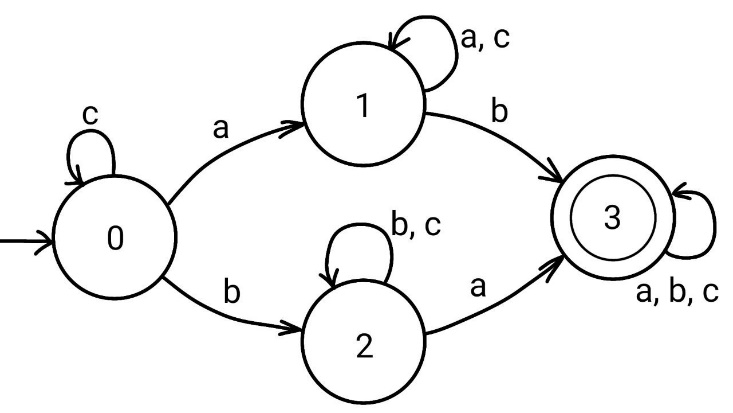


Рис. 6. Граф конечного автомата 2.4

Цепочки-примеры: acaaa, ab, cbbbc, accaacb, abcccbac, a, b.

1. Стр. 185 задания 19, 21

Задание 19. Построить автоматы, распознающие языки, задаваемые РВ: a\*b\*; a\*a\*; a\*+b\*; (a+b)\*; (a\*b\*)\*; (a\*+b\*)\*; a\*b+b\*a.

, X = {a, b}, δ: X×S→S.

Формальные модели КА, распознающих языки, задаваемые регулярными выражениями, представлены в виде ориентированных графов на рис. 7-13.

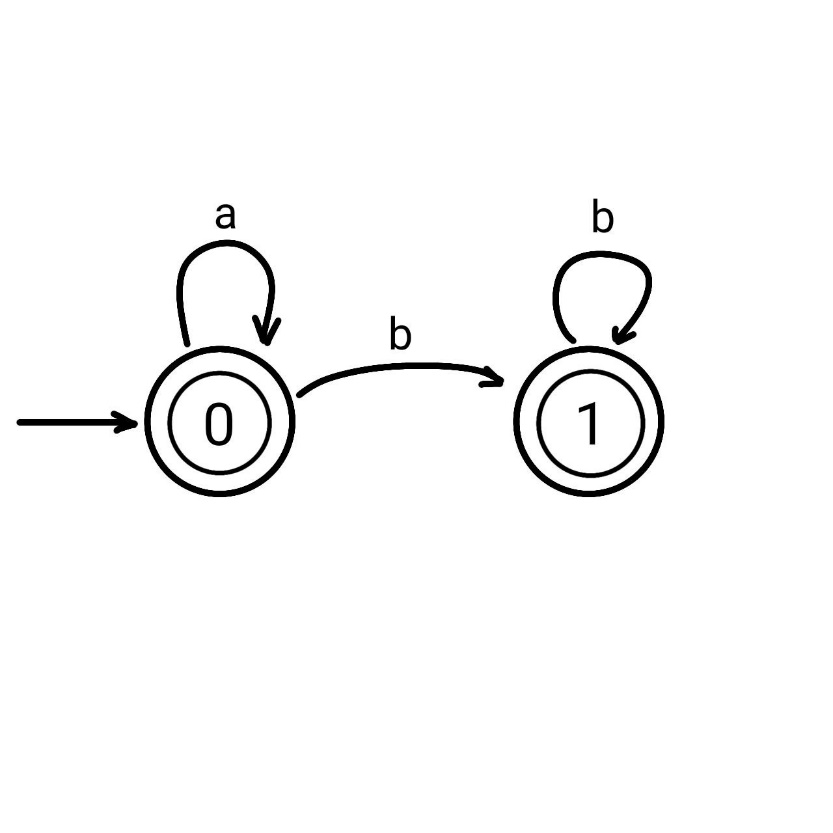


Рис. 7. Граф регулярного выражения «a\*b\*»

S = {0, 1}, F = {1}. Цепочки-примеры: b, ab, aab, bb, aaaaa, abbbbbb.

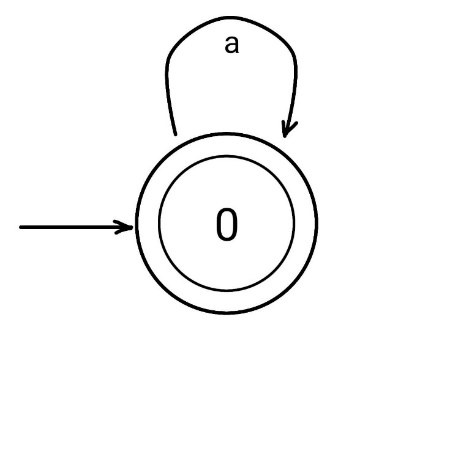


Рис. 8. Граф регулярного выражения «a\*a\*»

S = {0}, F = {0}. Цепочки-примеры: a, aa, aaaaaaaaaa.

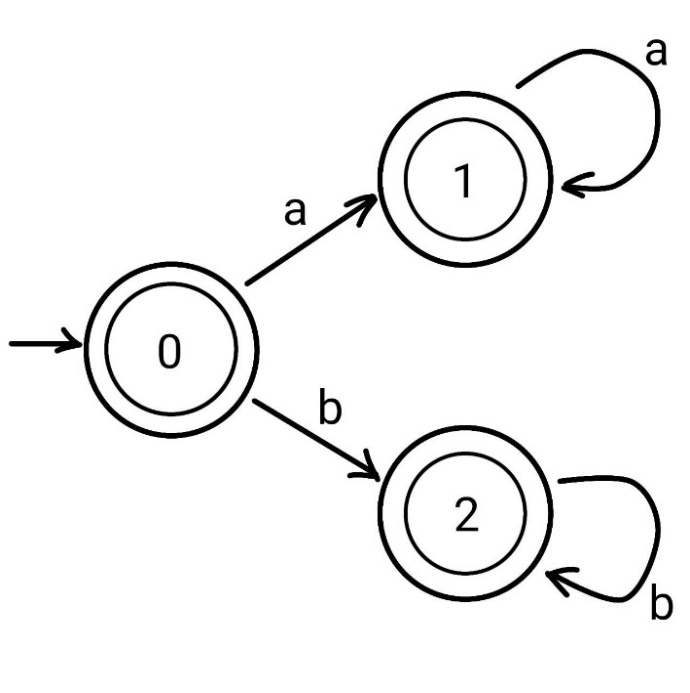


Рис. 9. Граф регулярного выражения «a\*+b\*»

S = {0, 1, 2}, F = {0, 1, 2}. Цепочки-примеры: aaa, bbbb, a, b.

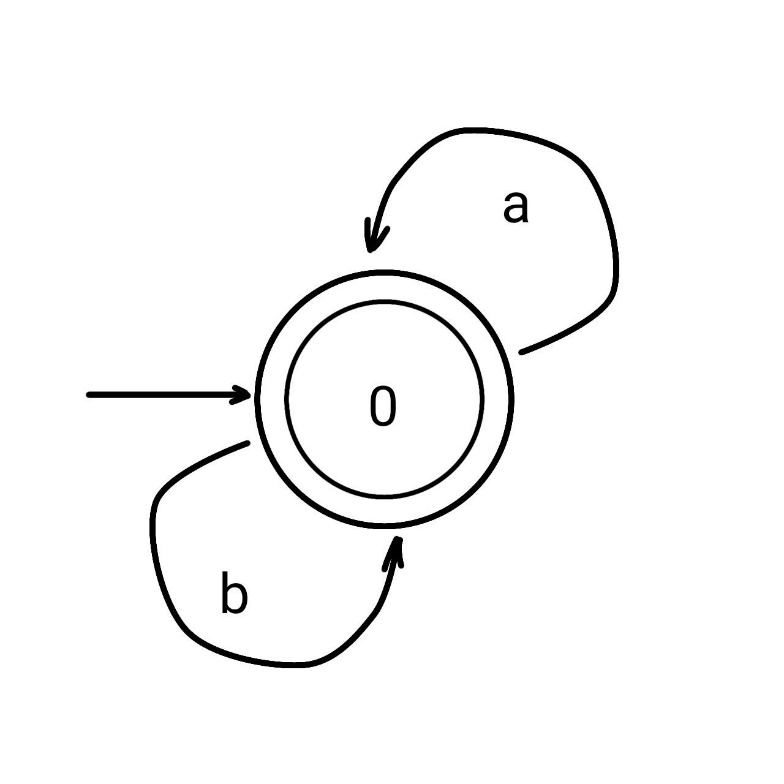


Рис. 10. Граф регулярного выражения «(a+b)\*»

S = {0}, F = {0}. Цепочки-примеры: b, ab, abbbaab.

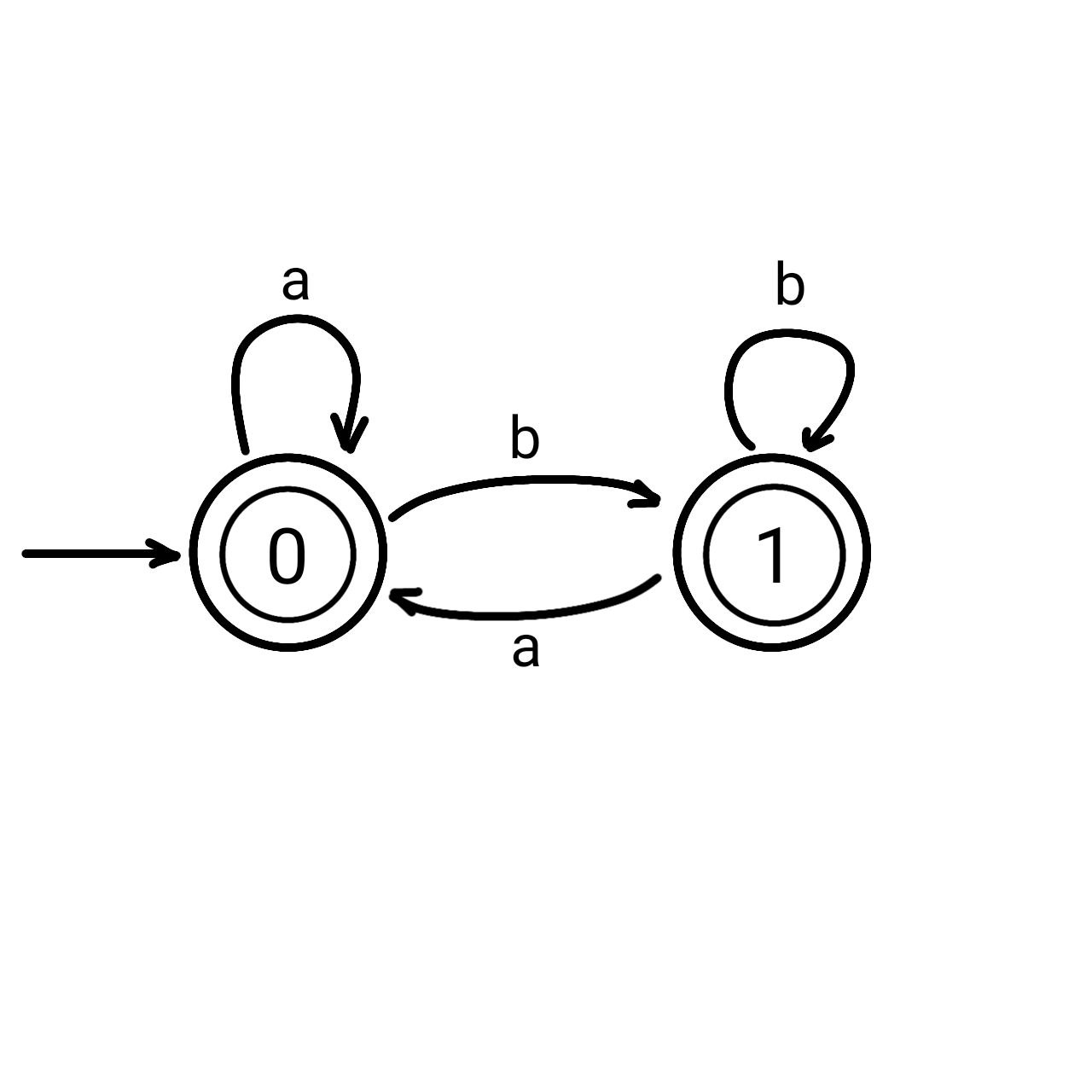


Рис. 11. Граф регулярного выражения «(a\*b\*)\*»

S = {0, 1}, F = {0, 1}. Цепочки-примеры: a, aaab, aabbbab, bbbab.

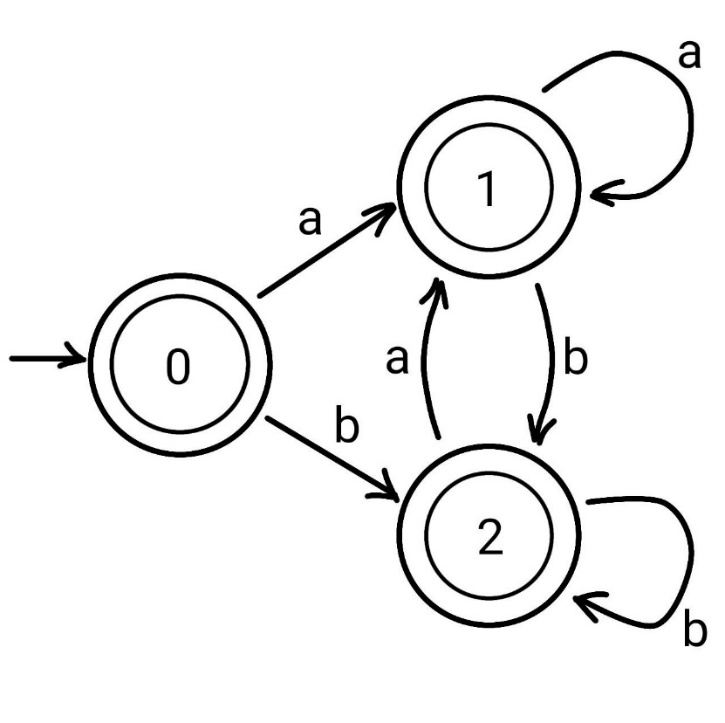


Рис. 12. Граф регулярного выражения «(a\*+b\*)\*»

S = {0, 1, 2}, F = {0, 1, 2}. Цепочки-примеры: a, b, abbaabab, ba.

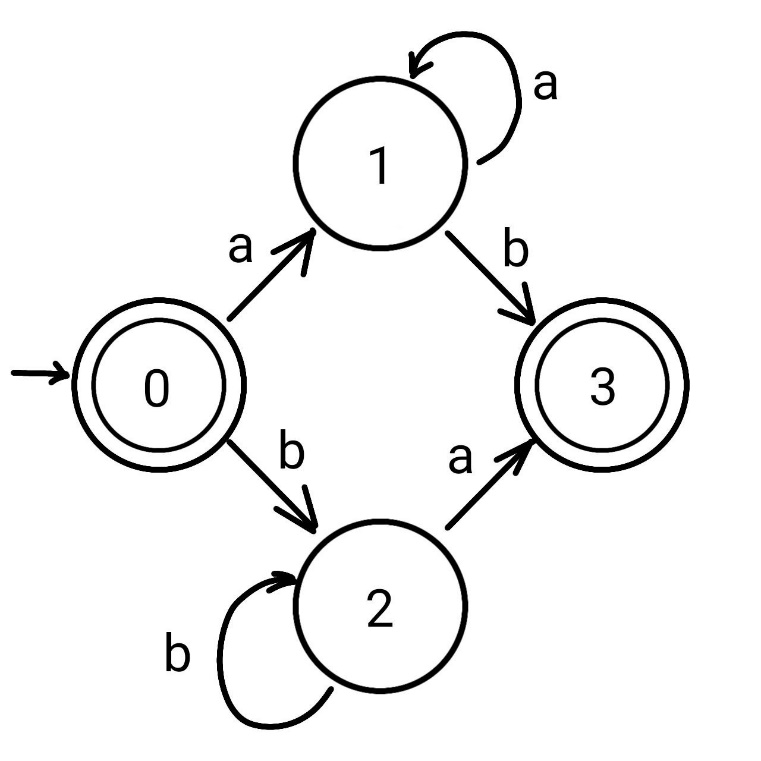


Рис. 13. Граф регулярного выражения «a\*b+b\*a»

S = {0, 1, 2, 3}, F = {0, 3}. Цепочки-примеры: ab, aaaaaaaaab, ba, bbbbbbba.

Задание 21. Построить регулярное выражение, задающее множество всех таких слов над словарём {a, b, c}, в которых за символом b а) обязательно стоит символ c б) не может стоять символ c. Построить конечные автоматы, распознающие соответствующие языки.

Регулярное выражение:

а) (c\*a\*(bc+)\*)\*;

б) (c\*((a+c\*)\*b\*)\*+(b\*a+c\*)\*)\*

Конечный автомат:

а) , X = {a, b, c}, S = {0, 1, 2, 3}, F = {2}, δ: X×S→S.

Формальная модель КА 21.а представлена в виде ориентированного графа на рис. 14.

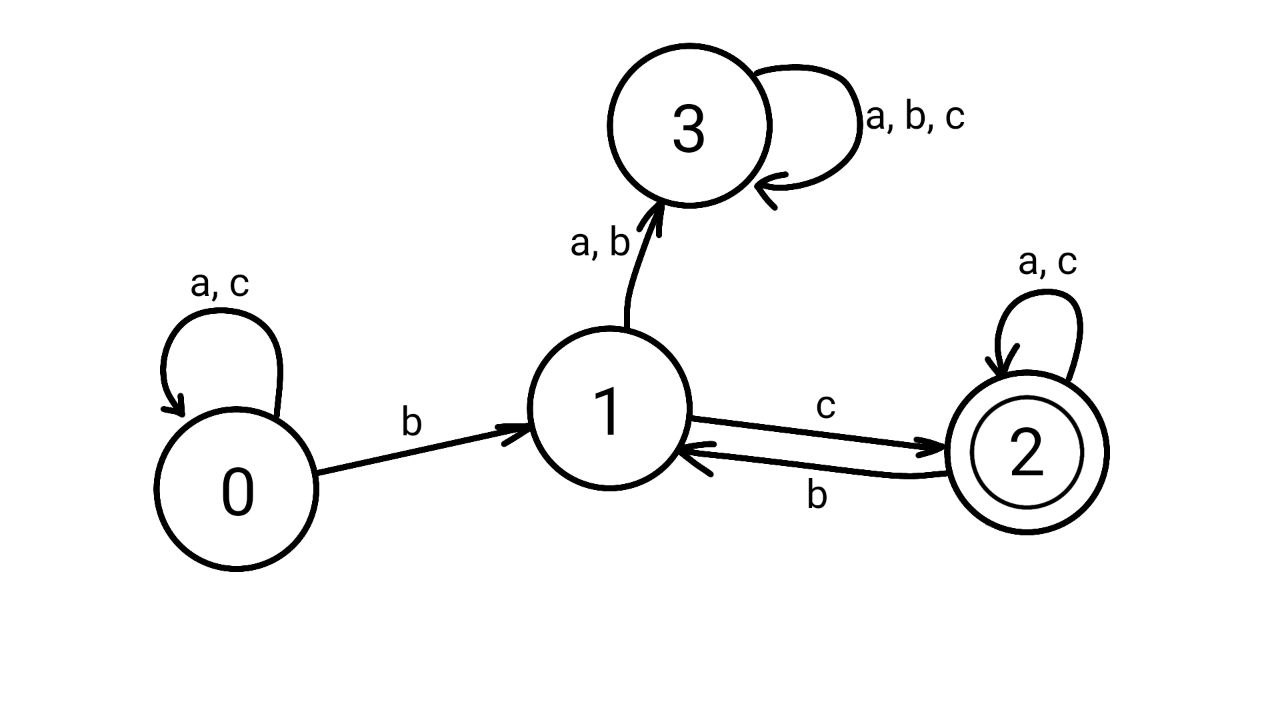


Рис. 14. Граф конечного автомата 21.а

Цепочки-примеры: abcbcaba, bca, abcc, cabc, cbcbcb, abccab.

б) , X = {a, b, c}, S = {0, 1, 2, 3, 4, 5}, F = {2,3,4}, δ: X×S→S.

Формальная модель КА 21.б представлена в виде ориентированного графа на рис. 15.

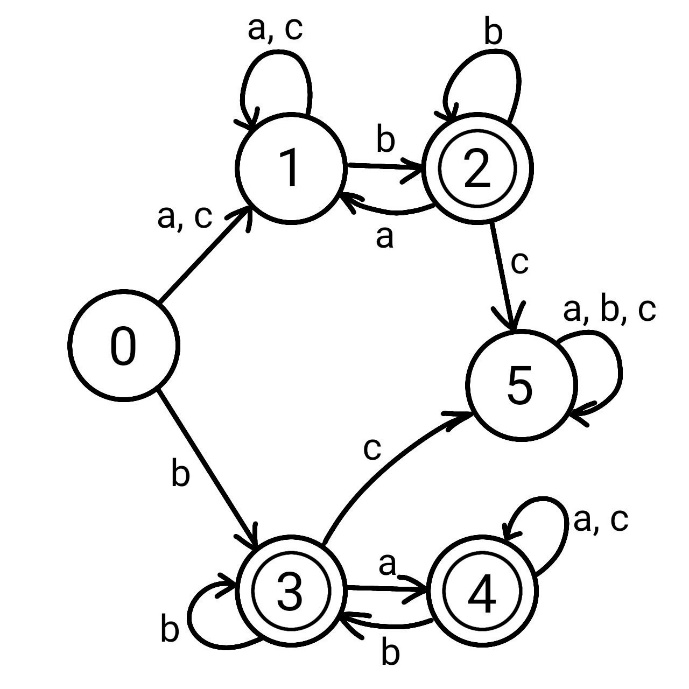


Рис. 15. Граф конечного автомата 21.б

Цепочки-примеры: abcbcaba, bca, abcc, cabc, cbcbcb, abccab.

1. Построить регулярное выражение, которое определяет язык, соответствующий варианту, 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 |
| 14 | 1, 0 | Множество слов, в которых четное число символов 0. |

;

X = {0, 1};

S = {0, 1};

F = {0};

δ: X×S→S.

Регулярное выражение, охватывающее не менее 4-5 вариантов слов: (1\*01\*0)\*+((00)\*1\*)\*

Формальная модель конечного автомата 3 представлена в виде ориентированного графа на рис. 16.

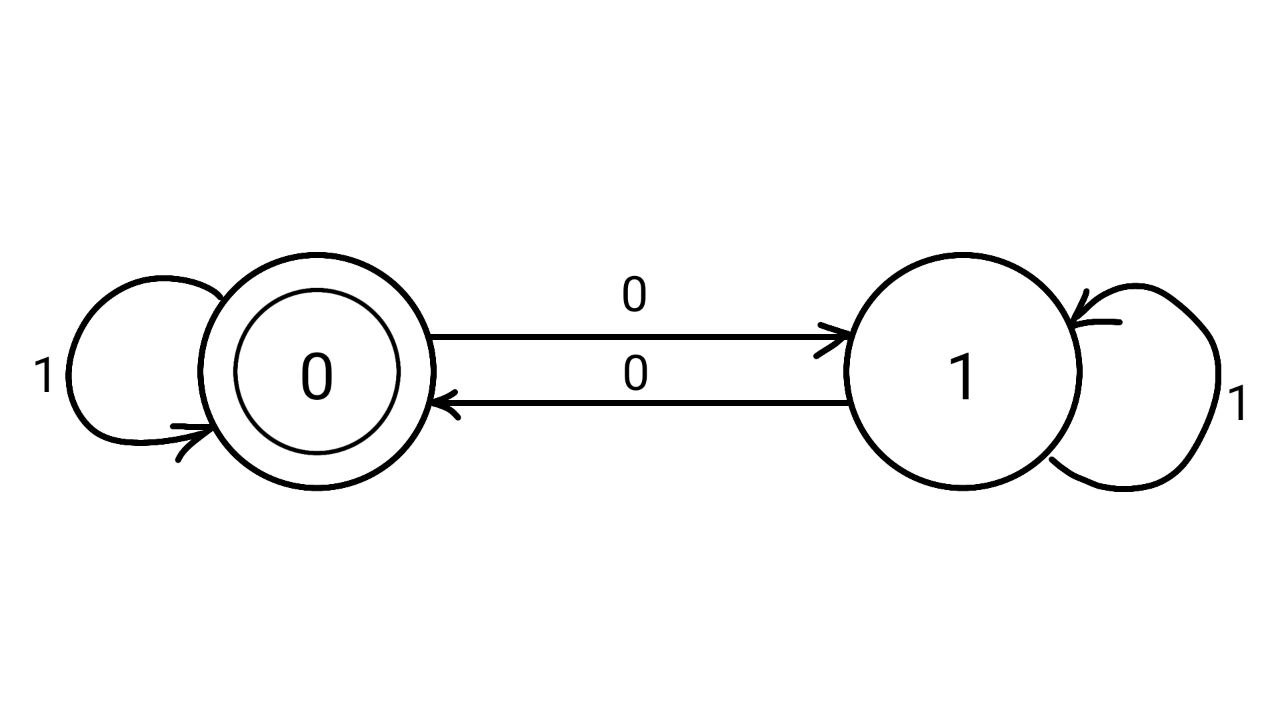


Рис. 16. Граф конечного автомата 3

Цепочки-примеры: 00, 001, 0011100, 1010, 111011000, 1100111100.

Блок-схема функции чтения слова программы представлена на рис. 17.

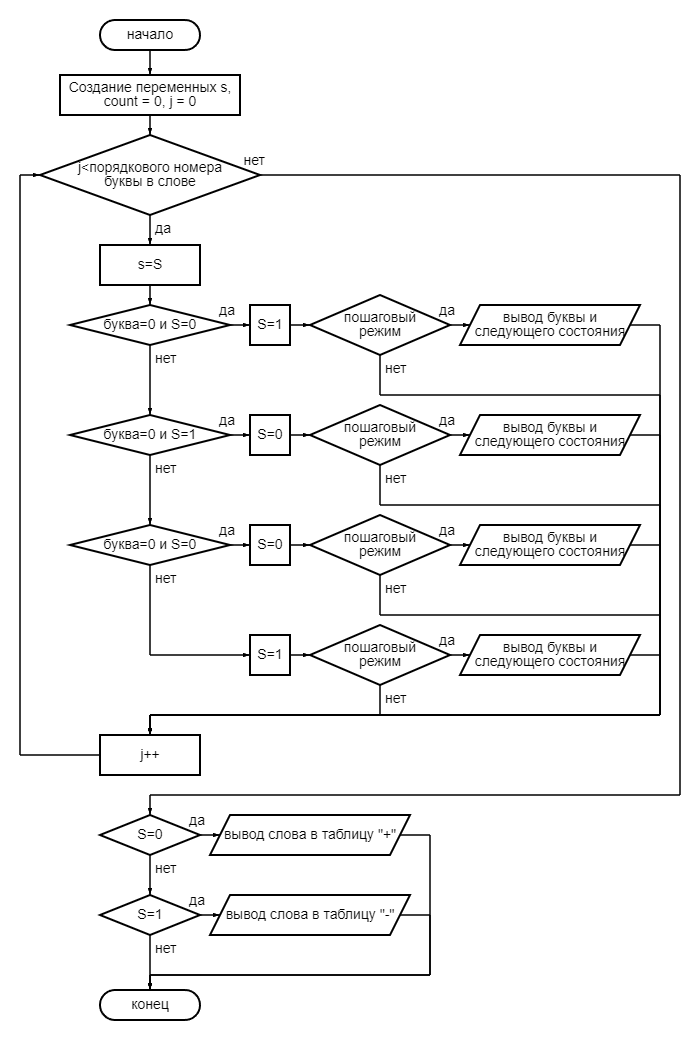


Рис. 17. Блок-схема функции чтения

Примеры корректной работы программы представлены на рис. 18-21.

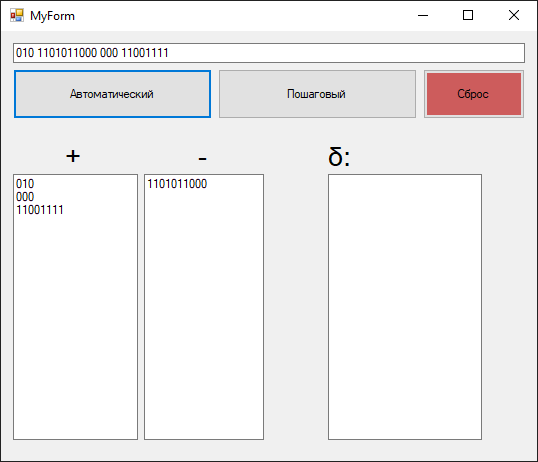


Рис. 18. Пример работы в автоматическом режиме

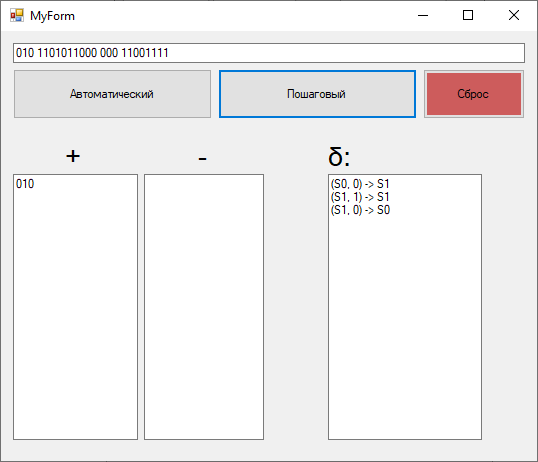


Рис. 19. Пример работы в пошаговом режиме: шаг 1

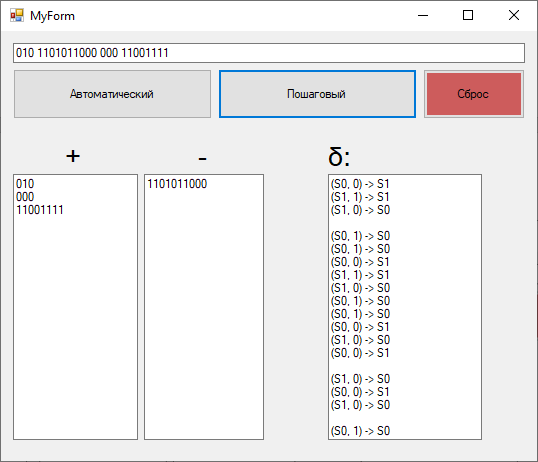


Рис. 20. Пример работы в пошаговом режиме: шаг 4

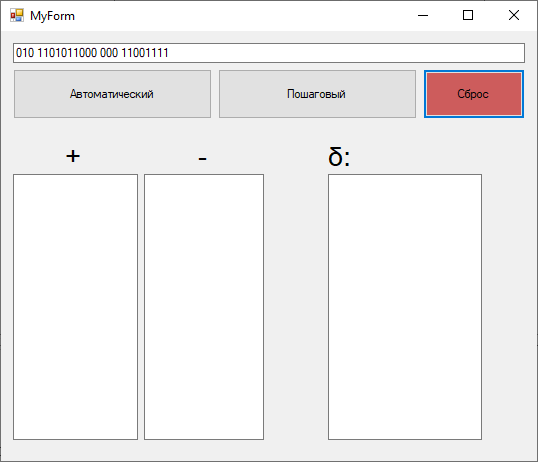


Рис. 21. Кнопка сброса очистила поля

Пример некорректной работы программы представлен на рис. 22.

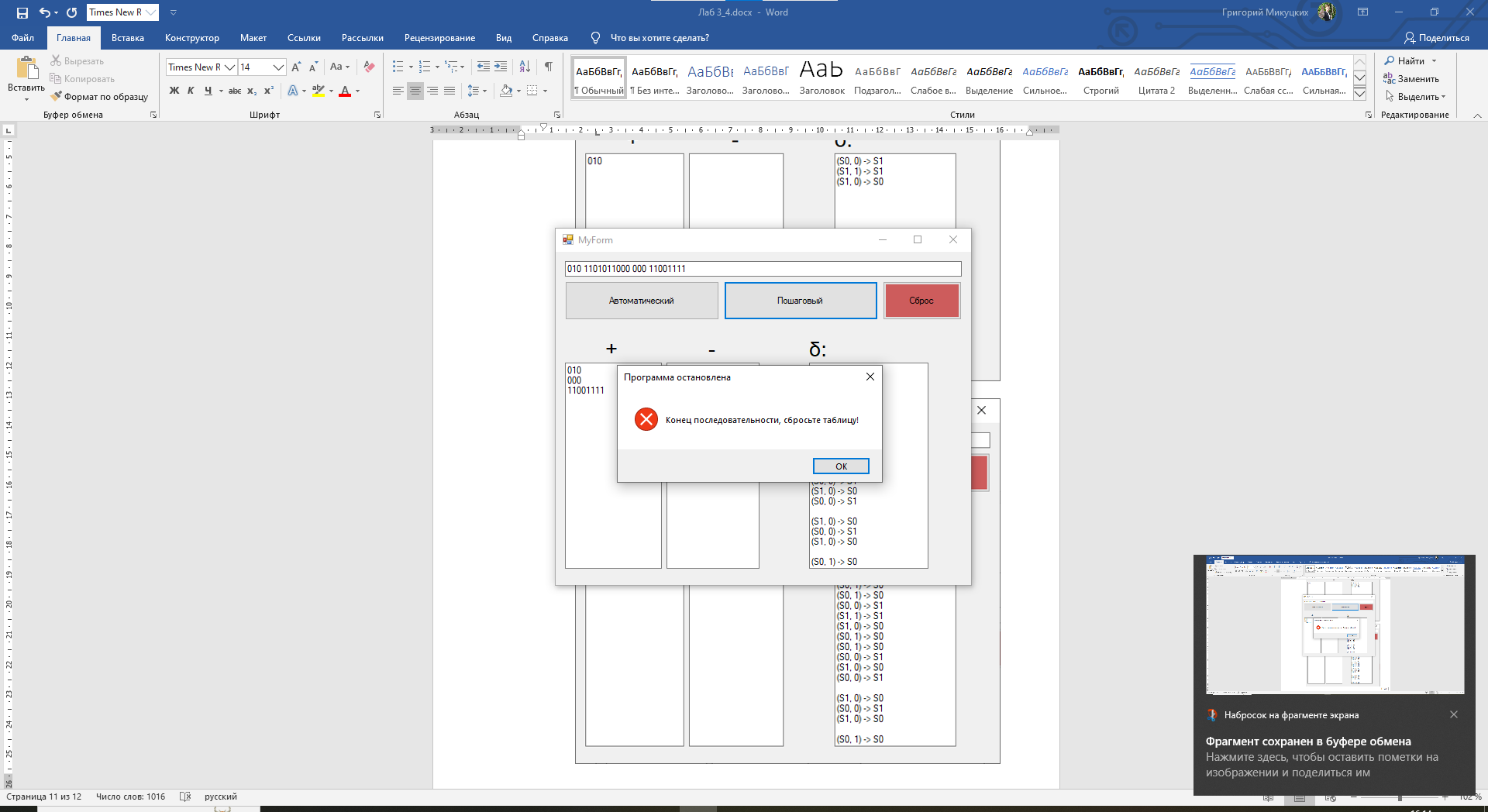


Рис. 22. Продолжить чтение в пошаговом режиме

после последнего слова не получится

Вывод

В ходе работы были построены детерминированные конечные автоматы по их описанию, заданному алфавиту, регулярному выражению. Были составлены регулярные выражения и модель КА по описанию автомата. Была написана программа по модели конечного автомата, реализующего распознавание строк, принадлежащих языку, определяемому регулярным выражением.

Листинг

#pragma once

namespace Project1 {

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>

/// Сводка для MyForm

/// </summary>

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

{

public:

MyForm(void)

{

InitializeComponent();

//

//TODO: добавьте код конструктора

//

}

protected:

/// <summary>

/// Освободить все используемые ресурсы.

/// </summary>

~MyForm()

{

if (components){delete components;}

}

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

protected:

private: System::Windows::Forms::Button^ btn\_auto;

private: System::Windows::Forms::Button^ btn\_step;

private: System::Windows::Forms::Button^ btn\_reset;

private: System::Windows::Forms::TextBox^ textBox\_true;

private: System::Windows::Forms::TextBox^ textBox\_false;

private: System::Windows::Forms::TextBox^ textBox\_delta;

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

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

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

protected:

private:

/// <summary>

/// Обязательная переменная конструктора.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Требуемый метод для поддержки конструктора — не изменяйте

/// содержимое этого метода с помощью редактора кода.

/// </summary>

void InitializeComponent(void)

{

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

this->btn\_auto = (gcnew System::Windows::Forms::Button());

this->btn\_step = (gcnew System::Windows::Forms::Button());

this->btn\_reset = (gcnew System::Windows::Forms::Button());

this->textBox\_true = (gcnew System::Windows::Forms::TextBox());

this->textBox\_false = (gcnew System::Windows::Forms::TextBox());

this->textBox\_delta = (gcnew System::Windows::Forms::TextBox());

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

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

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

this->SuspendLayout();

//

// result

//

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

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

this->result->Size = System::Drawing::Size(512, 20);

this->result->TabIndex = 0;

//

// btn\_auto

//

this->btn\_auto->Location = System::Drawing::Point(12, 38);

this->btn\_auto->Name = L"btn\_auto";

this->btn\_auto->Size = System::Drawing::Size(199, 50);

this->btn\_auto->TabIndex = 1;

this->btn\_auto->Text = L"Автоматический";

this->btn\_auto->UseVisualStyleBackColor = true;

this->btn\_auto->Click += gcnew System::EventHandler(this, &MyForm::btn\_auto\_Click);

//

// btn\_step

//

this->btn\_step->Location = System::Drawing::Point(217, 38);

this->btn\_step->Name = L"btn\_step";

this->btn\_step->Size = System::Drawing::Size(199, 50);

this->btn\_step->TabIndex = 2;

this->btn\_step->Text = L"Пошаговый";

this->btn\_step->UseVisualStyleBackColor = true;

this->btn\_step->Click += gcnew System::EventHandler(this, &MyForm::btn\_step\_Click);

//

// btn\_reset

//

this->btn\_reset->BackColor = System::Drawing::Color::IndianRed;

this->btn\_reset->Location = System::Drawing::Point(422, 38);

this->btn\_reset->Name = L"btn\_reset";

this->btn\_reset->Size = System::Drawing::Size(102, 50);

this->btn\_reset->TabIndex = 3;

this->btn\_reset->Text = L"Сброс";

this->btn\_reset->UseVisualStyleBackColor = false;

this->btn\_reset->Click += gcnew System::EventHandler(this, &MyForm::btn\_reset\_Click);

//

// textBox\_true

//

this->textBox\_true->Location = System::Drawing::Point(12, 143);

this->textBox\_true->Multiline = true;

this->textBox\_true->Name = L"textBox\_true";

this->textBox\_true->Size = System::Drawing::Size(125, 266);

this->textBox\_true->TabIndex = 4;

//

// textBox\_false

//

this->textBox\_false->Location = System::Drawing::Point(143, 143);

this->textBox\_false->Multiline = true;

this->textBox\_false->Name = L"textBox\_false";

this->textBox\_false->Size = System::Drawing::Size(120, 266);

this->textBox\_false->TabIndex = 5;

//

// textBox\_delta

//

this->textBox\_delta->Location = System::Drawing::Point(327, 143);

this->textBox\_delta->Multiline = true;

this->textBox\_delta->Name = L"textBox\_delta";

this->textBox\_delta->Size = System::Drawing::Size(154, 266);

this->textBox\_delta->TabIndex = 6;

//

// label1

//

this->label1->AutoSize = true;

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

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

this->label1->Location = System::Drawing::Point(58, 109);

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

this->label1->Size = System::Drawing::Size(30, 31);

this->label1->TabIndex = 7;

this->label1->Text = L"+";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label2->Location = System::Drawing::Point(191, 109);

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

this->label2->Size = System::Drawing::Size(23, 31);

this->label2->TabIndex = 8;

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

//

// label3

//

this->label3->AutoSize = true;

this->label3->Font = (gcnew System::Drawing::Font(L"Microsoft Sans Serif", 20.25F, System::Drawing::FontStyle::Regular, System::Drawing::GraphicsUnit::Point,

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

this->label3->Location = System::Drawing::Point(321, 109);

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

this->label3->Size = System::Drawing::Size(37, 31);

this->label3->TabIndex = 9;

this->label3->Text = L"δ:";

//

// MyForm

//

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

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

this->ClientSize = System::Drawing::Size(536, 430);

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

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

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

this->Controls->Add(this->textBox\_delta);

this->Controls->Add(this->textBox\_false);

this->Controls->Add(this->textBox\_true);

this->Controls->Add(this->btn\_reset);

this->Controls->Add(this->btn\_step);

this->Controls->Add(this->btn\_auto);

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

this->Name = L"MyForm";

this->Text = L"MyForm";

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

int S = 0;

void reading(int i, cli::array<String^>^ word, bool mode) {

int s, count = 0, j = 0;

while (j < word[i]->Length) {

s = S;

if ((word[i])[j] == '0' && S == 0) {

S = 1;

if (mode) this->textBox\_delta->Text += "(S" + s + ", 0) -> S" + S + Environment::NewLine;

}

else if ((word[i])[j] == '0' && S == 1) {

S = 0;

if (mode) this->textBox\_delta->Text += "(S" + s + ", 0) -> S" + S + Environment::NewLine;

}

else if ((word[i])[j] == '1' && S == 0) {

S = 0;

if (mode) this->textBox\_delta->Text += "(S" + s + ", 1) -> S" + S + Environment::NewLine;

}

else {

S = 1;

if (mode) this->textBox\_delta->Text += "(S" + s + ", 1) -> S" + S + Environment::NewLine;

}

j++;

}

switch (S)

{

case 0: this->textBox\_true->Text += word[i] + Environment::NewLine; break;

case 1: this->textBox\_false->Text += word[i] + Environment::NewLine; break;

}

} int step\_word = 0;

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

String^ str = result->Text;

bool mode = false;

this->textBox\_true->Text = "";

this->textBox\_false->Text = "";

cli::array<String^>^ word = str->Split(' ');

S = 0;

for (int i = 0; i < word->Length; i++) {

if (word[i] != "") reading(i, word, mode);

}

step\_word = word->Length;

}

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

String^ str = result->Text;

cli::array<String^>^ word = str->Split(' ');

bool mode = true;

if (step\_word < word->Length) {

if (word[step\_word] != "") reading(step\_word, word, mode);

step\_word++;

this->textBox\_delta->Text += Environment::NewLine;

}

else MessageBox::Show(L"Конец последовательности, сбросьте таблицу!", L"Программа остановлена", MessageBoxButtons::OK, MessageBoxIcon::Error);

}

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

this->textBox\_true->Text = "";

this->textBox\_false->Text = "";

this->textBox\_delta->Text = "";

step\_word = 0;

}

};

}