МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт Информационных Технологий

Кафедра МПО ЭВМ

Дисциплина «Теория автоматов и формальных языков»

Лабораторная работа №3-4

«Регулярные множества и регулярные выражения»

Выполнил:

студент группы 1ПИб-02-3оп-22

Маркелов Сергей Александрович

Проверил:

доцент, к.т.н.

Ганичева Оксана Георгиевна

Череповец, 2024 год

**Задание 1.**

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

е) Все цепочки, заканчивающиеся кодом aabba

**Регулярное выражение:** (a+b)\*aabba

**Примеры строк:** aabba, aaabba, baabba, abababababaababbaabba и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

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 |

**Граф:**

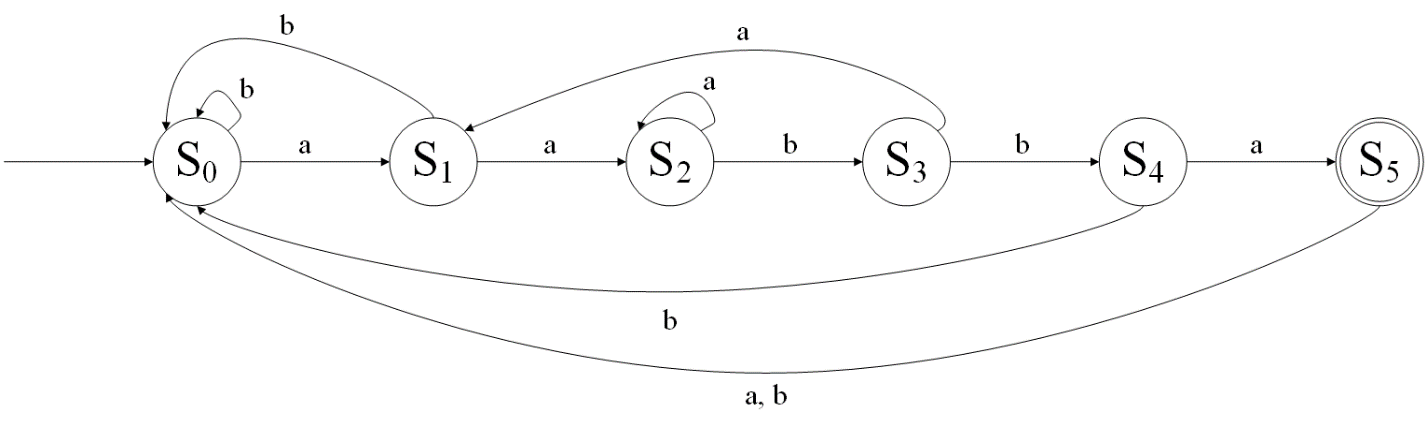


Рис. 1. Граф конечного автомата, соответвующего выражению (a+b)\*aabba

ж) Все цепочки, в которых за каждым a непосредственно следует b

**Регулярное выражение:** (b\*(ab+)\*)\*

**Примеры строк:** ε, b, bb, bbb, ab, bab, babb, babab, abab, bbbbbbabbbbbabbbbbabb и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S2}

δ:

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

**Граф:**

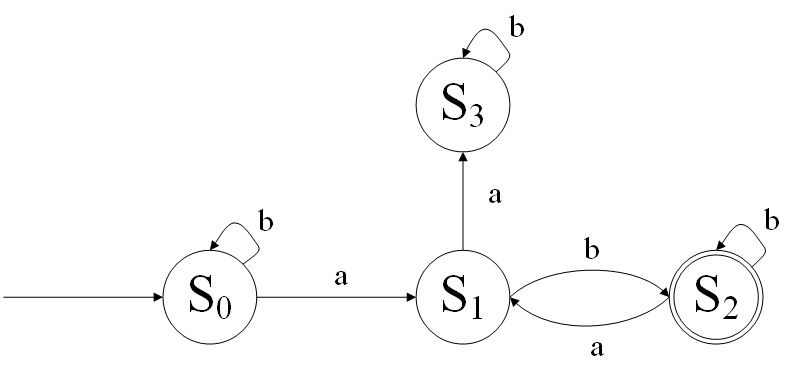


Рис. 2. Граф конечного автомата, соответвующего выражению (b\*(ab+)\*)\*

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

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

**Примеры строк:** ε, b, c, bc, cb, bcbcbbcb,ab, acb, cacbc, bccbbcccbbcacccccccbacbc и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2}

S0 = {S0}

F = {S2}

δ:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (S0, a) → S1 | (S1, a) → S1 | (S2, a) → S1 |  |  |  |
| (S0, b) → S0 | (S1, b) → S2 | (S2, b) → S2 |  |  |  |
| (S0, c) → S0 | (S1, c) → S1 | (S2, c) → S2 |  |  |  |

**Граф:**

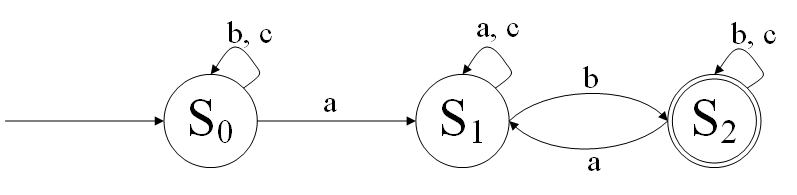


Рис. 3. Граф конечного автомата, соответвующего выражению (b+c)\*(a(a+c)\*b(b+c)\*)\*

б) Все цепочки, в которых 2 последние буквы не совпадают

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

**Примеры строк:** ab, ba, ac, ca, bc, cb, abc, abcbc, abbabcbccbabacbcbcb и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2, S3, S4, S5, S6}

S0 = {S0}

F = {S4, S5, S6}

δ:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| (S0, a)→ S1 | (S1, a)→ S1 | (S2, a)→ S4 | (S3, a)→ S4 | (S4, a)→ S1 | (S5, a)→ S4 | (S6, a)→ S4 |
| (S0, b)→ S2 | (S1, b)→ S5 | (S2, b)→ S2 | (S3, b)→ S5 | (S4, b)→ S5 | (S5, b)→ S2 | (S6, b)→ S5 |
| (S0, c)→ S3 | (S1, c)→ S6 | (S2, c)→ S6 | (S3, c)→ S3 | (S4, c)→ S6 | (S5, c)→ S6 | (S6, c)→ S3 |

**Граф:**

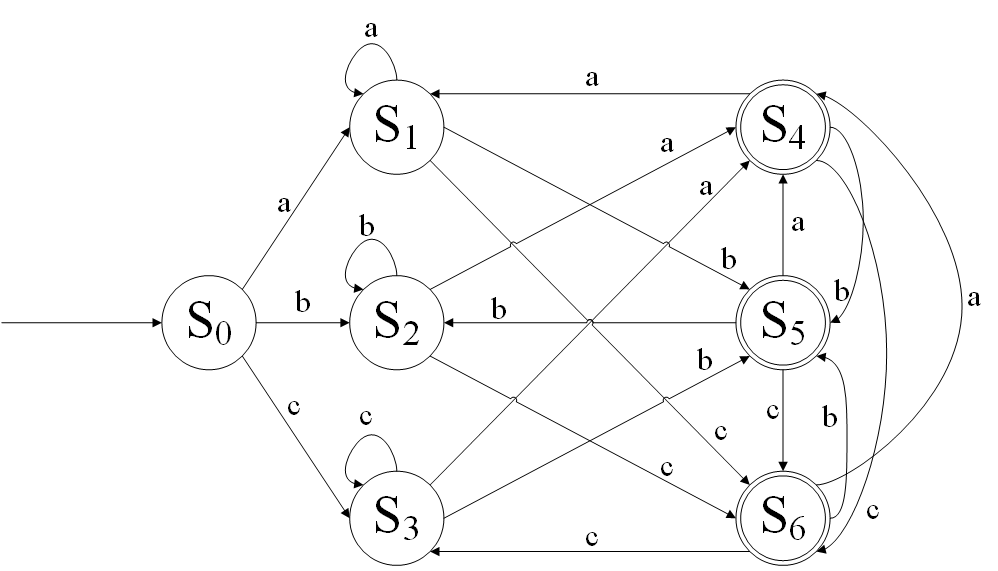


Рис. 4. Граф конечного автомата, соответвующего выражению (a+b+c)\*((a(b+c))+(b(a+c))+(c(a+b)))

в) Все цепочки, начинающиеся и заканчивающиеся различными символами

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

**Примеры строк:** ab, ba, ac, ca, bc, cb, abc, abcbc, abbabcbccbabacbcbcb и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2, S3, S4, S5, S6}

S0 = {S0}

F = {S4, S5, S6}

δ:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| (S0, a)→ S1 | (S1, a)→ S1 | (S2, a)→ S5 | (S3, a)→ S6 | (S4, a)→ S1 | (S5, a)→ S5 | (S6, a)→ S6 |
| (S0, b)→ S2 | (S1, b)→ S4 | (S2, b)→ S2 | (S3, b)→ S6 | (S4, b)→ S4 | (S5, b)→ S2 | (S6, b)→ S6 |
| (S0, c)→ S3 | (S1, c)→ S4 | (S2, c)→ S5 | (S3, c)→ S3 | (S4, c)→ S4 | (S5, c)→ S5 | (S6, c)→ S3 |

**Граф:**

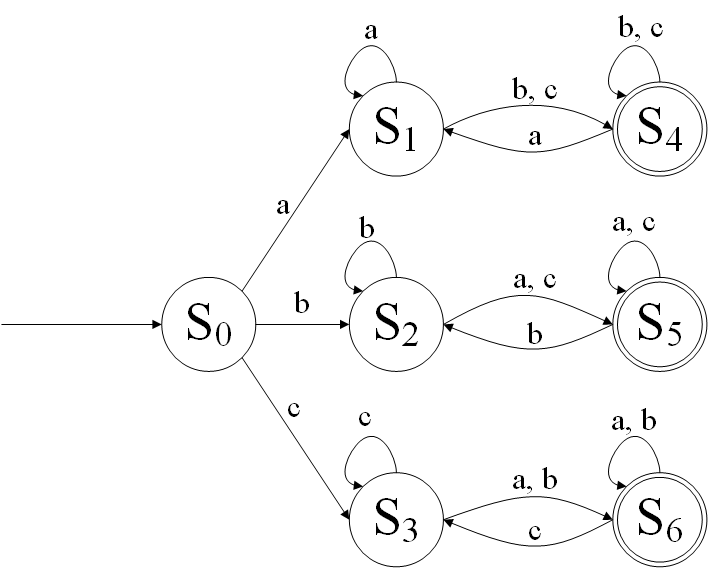


Рис. 5. Граф конечного автомата, соответвующего выражению (a(a+b+c)\*(b+c))+(b(a+b+c)\*(a+c))+(c(a+b+c)\*(a+b))

г) Все цепочки, включающие по крайней мере 1 символ a и 1 символ b

**Регулярное выражение:**

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

**Примеры строк:** ab, ba, acb, bca, cabc, aabb, cccccccccccbcccccccacссссс и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S3}

δ:

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

**Граф:**

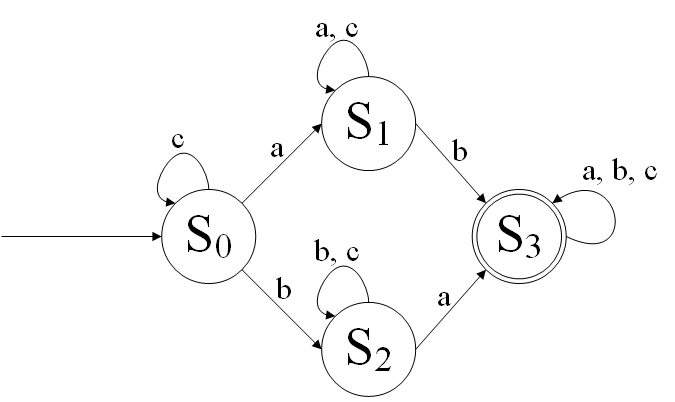


Рис. 6. Граф конечного автомата, соответвующего выражению (a+b+с)\*((a(a+b+c)\*b(a+b+c)\*)+(b(a+b+c)\*a(a+b+c)\*))(a+b+с)\*

**Задание 2.**

19. Построить автоматы, распознающие языки, задаваемые регулярными выражениями

**Регулярное выражение:** a\*b\*

**Описание:** все цепочки, в которых сначала идет подряд любое количество символов a, а затем подряд любое количество символов b

**Примеры строк:** ε, a, b, ab, aab, abb, aabb, aaaaaaaaaaaabbbbbbbbbbbbb и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0, S1, S2}

S0 = {S0}

F = {S0, S1}

δ:

|  |  |  |  |
| --- | --- | --- | --- |
| (S0, a) → S0 | (S1, a) → S2 | (S2, a) → S2 |  |
| (S0, b) → S1 | (S1, b) → S1 | (S2, b) → S2 |  |

**Граф:**

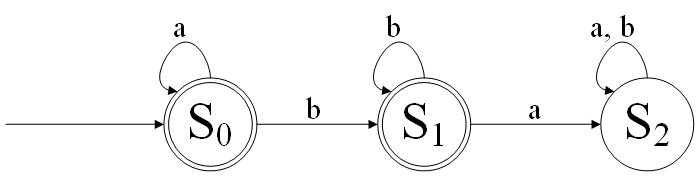


Рис. 7. Граф конечного автомата, соответвующего выражению a\*b\*

**Регулярное выражение:** a\*a\*

**Описание:** все цепочки, состоящие из любого количества символов a

**Примеры строк:** ε, a, aa, aaa, aaaa, aaaaa, aaaaaaaaaaaaaaaaaaa и т. д.

А = {X, S, S0, F, δ}

X = {a}

S = {S0}

S0 = {S0}

F = {S0}

δ: (S0, a) → S0

**Граф:**

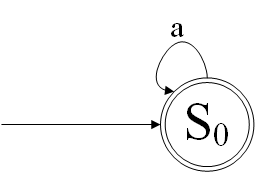


Рис. 8. Граф конечного автомата, соответвующего выражению a\*a\*

**Регулярное выражение:** a\*+b\*

**Описание:** все цепочки, состоящие из любого количества символов a или из любого количества символов b

**Примеры строк:** ε, a, b, aa, bb, aaa, bbb, aaaaaaaaaaaaa, bbbbbbbbbbb и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S1, S2}

δ:

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

**Граф:**

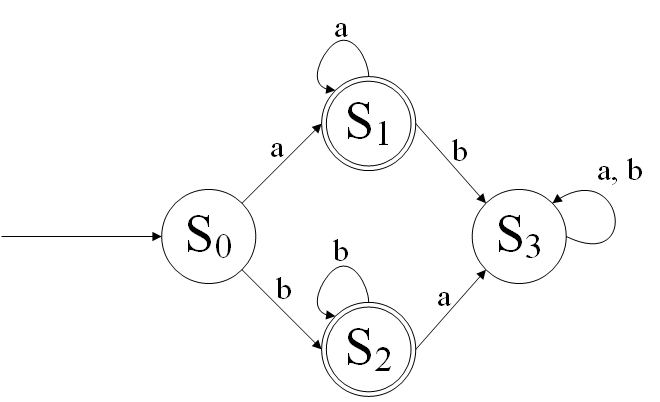


Рис. 9. Граф конечного автомата, соответвующего выражению a\*+b\*

**Регулярное выражение:** (a+b)\*

**Описание:** все цепочки, состоящие из любого количества символов a и/или b

**Примеры строк:** ε, a, b, aa, bb, ab, ba, aab, bab, aabb, bbaa, abab, baba, abaabbabababaabbaba и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0}

S0 = {S0}

F = {S0}

δ: (S0, a) → S0 (S0, b) → S0

**Граф:**

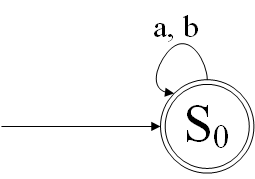


Рис. 10. Граф конечного автомата, соответвующего выражению (a+b)\*

**Регулярное выражение:** (a\*b\*)\*

**Описание:** все цепочки, состоящие из любого количества символов a и/или b

**Примеры строк:** ε, a, b, aa, bb, aaa, bbb, aaaaaaaaaaaaa, bbbbbbbbbbb и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0}

S0 = {S0}

F = {S0}

δ: (S0, a) → S0 (S0, b) → S0

**Граф:**

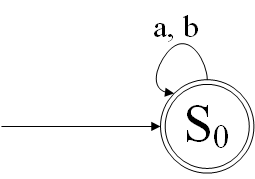


Рис. 11. Граф конечного автомата, соответвующего выражению (a\*b\*)\*

**Регулярное выражение:** (a\*+b\*)\*

**Описание:** все цепочки, состоящие из любого количества символов a и/или b

**Примеры строк:** ε, a, b, aa, bb, aaa, bbb, aaaaaaaaaaaaa, bbbbbbbbbbb и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0}

S0 = {S0}

F = {S0}

δ: (S0, a) → S0 (S0, b) → S0

**Граф:**

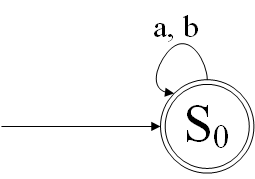


Рис. 12. Граф конечного автомата, соответвующего выражению (a\*+b\*)\*

**Регулярное выражение:** a\*b+b\*a

**Описание:** все цепочки, состоящие из любого количества символов a и затем символа b или состоящие из любого количества символов b и затем символа a

**Примеры строк:** ε, a, b, ab, ba, aab, bba, aaaaaaaaaaab, bbbbbbbbbbbba и т. д.

А = {X, S, S0, F, δ}

X = {a, b}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S3}

δ:

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

**Граф:**

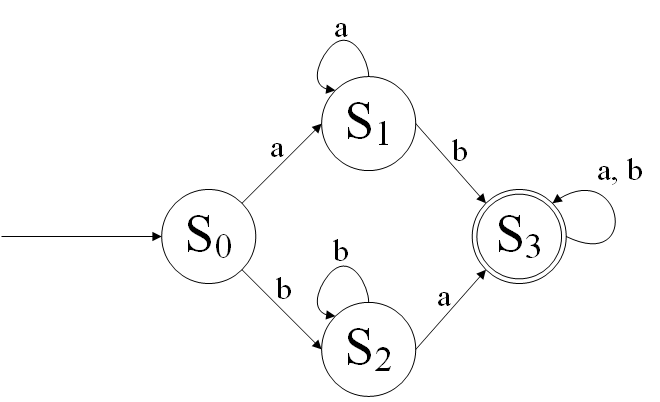


Рис. 13. Граф конечного автомата, соответвующего выражению a\*b+b\*a

21. Построить регулярное выражение, задающее множество всех таких слов над словарем {a, b, c}, и конечные автоматы, распознающие соответствующие языки, в которых за символом b:

а) обязательно стоит символ c

**Регулярное выражение:** (a+c)\*(bc(a+c)\*)\*

**Примеры строк:** ε, a, c, ac, acca, abc, bcbc, cbcbc, acbcbcabccacaaccabca и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S2}

δ:

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

**Граф:**

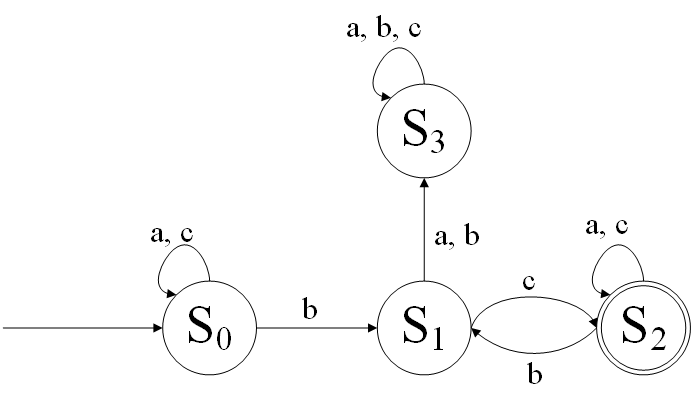


Рис. 14. Граф конечного автомата, соответвующего выражению (a+c)\*(bc(a+c)\*)\*

б) не может стоять символ c

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

**Примеры строк:** ε, a, c, ac, acca, b, ba, bb, bbbbbb, bababbabbaba, cccccccccbbaccccccbaccccc и т. д.

А = {X, S, S0, F, δ}

X = {a, b, c}

S = {S0, S1, S2}

S0 = {S0}

F = {S1}

δ:

|  |  |  |
| --- | --- | --- |
| (S0, a) → S0 | (S1, a) → S1 | (S2, a) → S2 |
| (S0, b) → S1 | (S1, b) → S1 | (S2, b) → S2 |
| (S0, c) → S0 | (S1, c) → S2 | (S2, c) → S2 |

**Граф:**

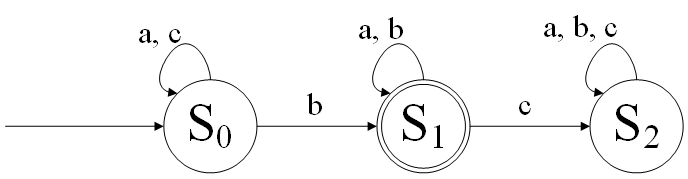


Рис. 15. Граф конечного автомата, соответвующего выражению (a+c)\*((b\*a\*)+(b\*a(a+c)\*))\*

**Задание 3.**

1. Регулярное выражение

Построить регулярное выражение, которое определяет язык, соответствующий варианту:

|  |  |  |
| --- | --- | --- |
| Вариант | Алфавит языка | Описание языка |
| 14 | 1, 0 | Множество слов, в которых четное число символов 0 |

*Примечание: в данном задании решение для общего случая с использованием регулярного выражения затруднительно, поэтому решение должно быть частичным, т. е. предусматривать только некоторые случаи. Решение должно охватывать не менее 4-5 вариантов таких слов.*

**Регулярное выражение:** (1\*01\*0)\*

**Примеры строк:** 100, 10101, 1010, 1001, 11011011, 101101, 101010101 и т. д.

1. Формальное описание конечного автомата

Для заданного регулярного выражения построить детерминированный конечный автомат.

А = {X, S, S0, F, δ}

X = {0, 1}

S = {S0, S1, S2, S3}

S0 = {S0}

F = {S3}

δ:

|  |  |  |  |
| --- | --- | --- | --- |
| (S0, 0) → S1 | (S1, 0) → S3 | (S2, 0) → S3 | (S3, 0) → S1 |
| (S0, 1) → S0 | (S1, 1) → S2 | (S2, 1) → S2 | (S3, 1) → S0 |

1. Граф конечного автомата

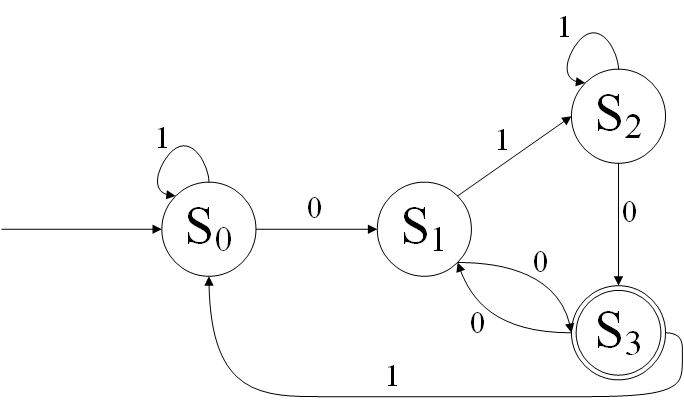


Рис. 16. Граф конечного автомата, соответвующего выражению (1\*01\*0)\*

1. Описание и блок-схема программы

На основе конечного автомата написать программу для распознавания строк, принадлежащих языку, определяемому регулярным выражением.

Представление результатов работы программы

1. В автоматическом режиме:
   1. Входная строка – содержит произвольное количество строк в заданном алфавите
   2. Две таблицы – в которые записываются правильные и неправильные строки из входного множества
2. В пошаговом режиме для заданного слова осуществляется разбор по шагам:
   1. Во входной строке – одно слово
   2. В результате – отображается последовательность команд конечного автомата

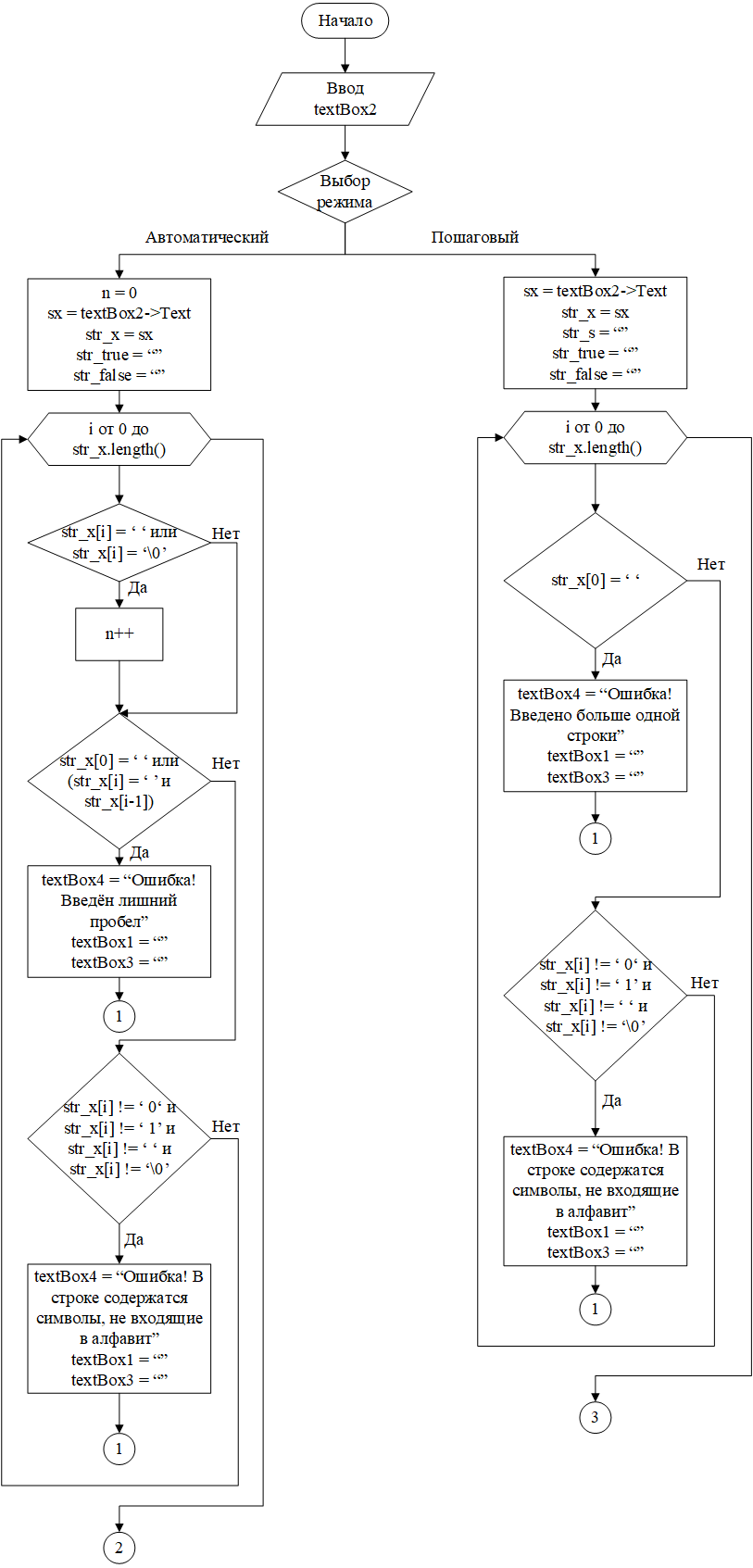
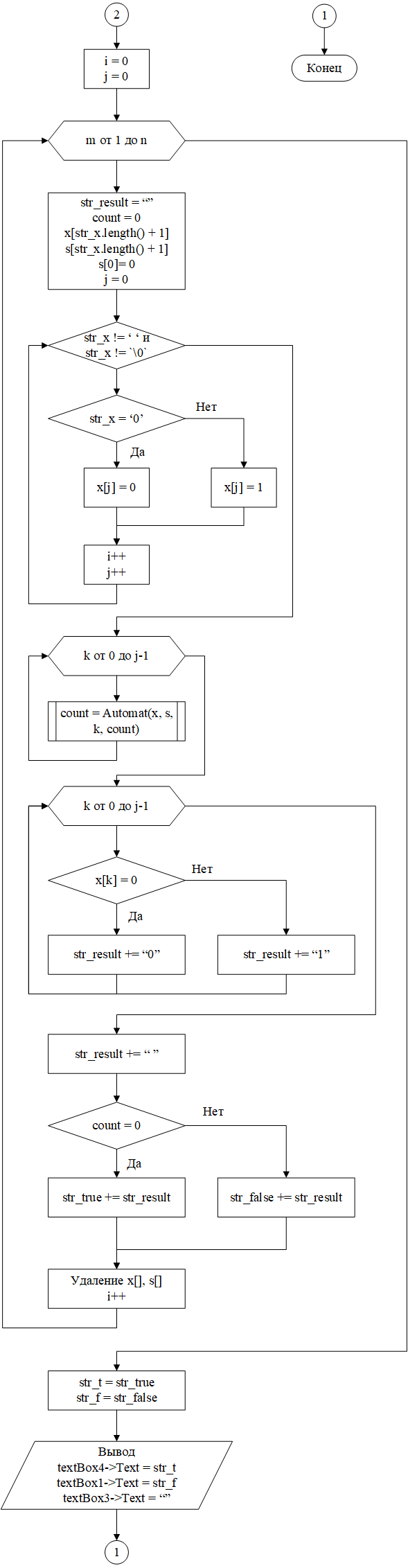
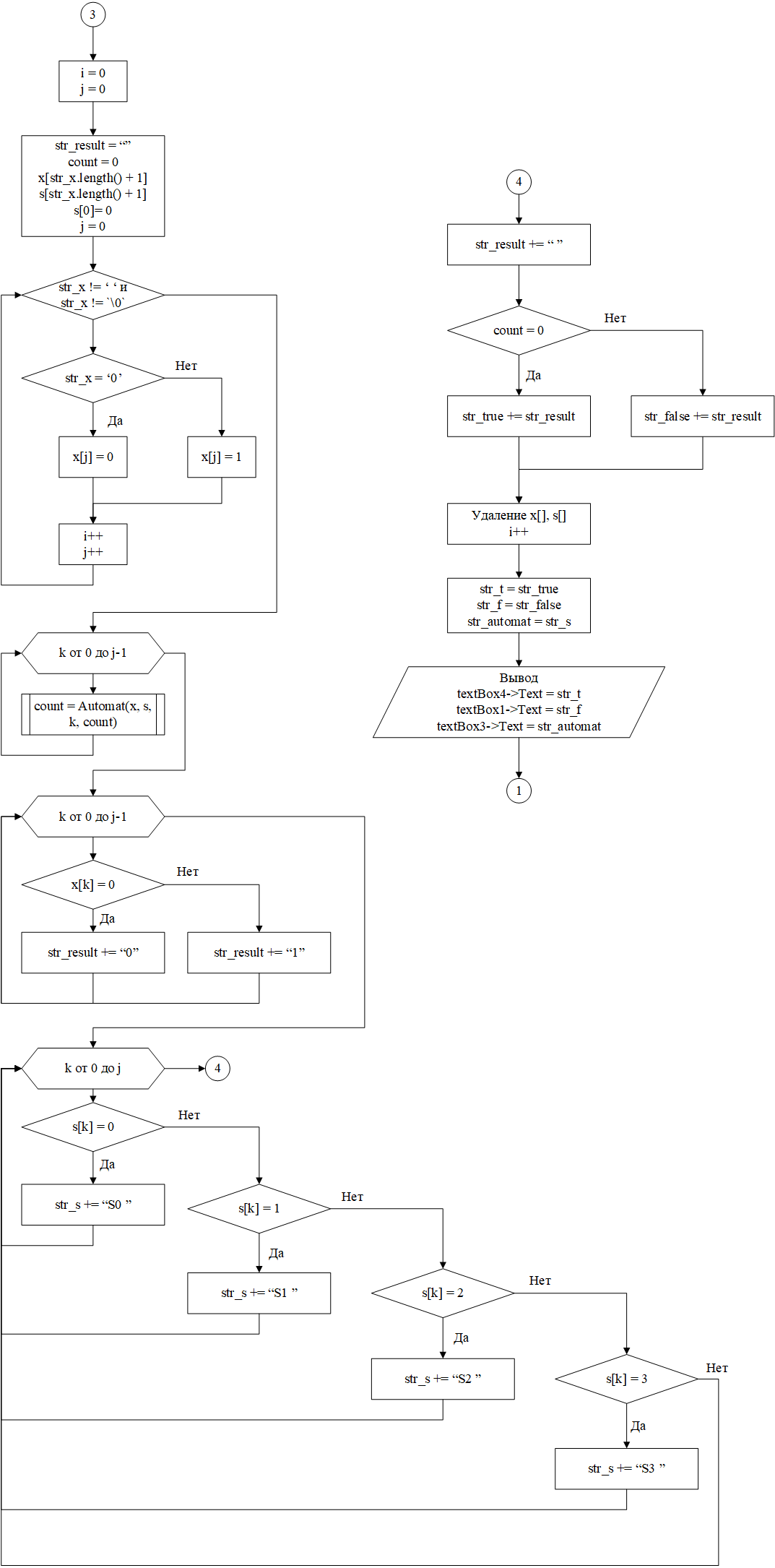


Рис. 17. Блок-схема программы

Продолжение рис. 17



Продолжение рис. 17



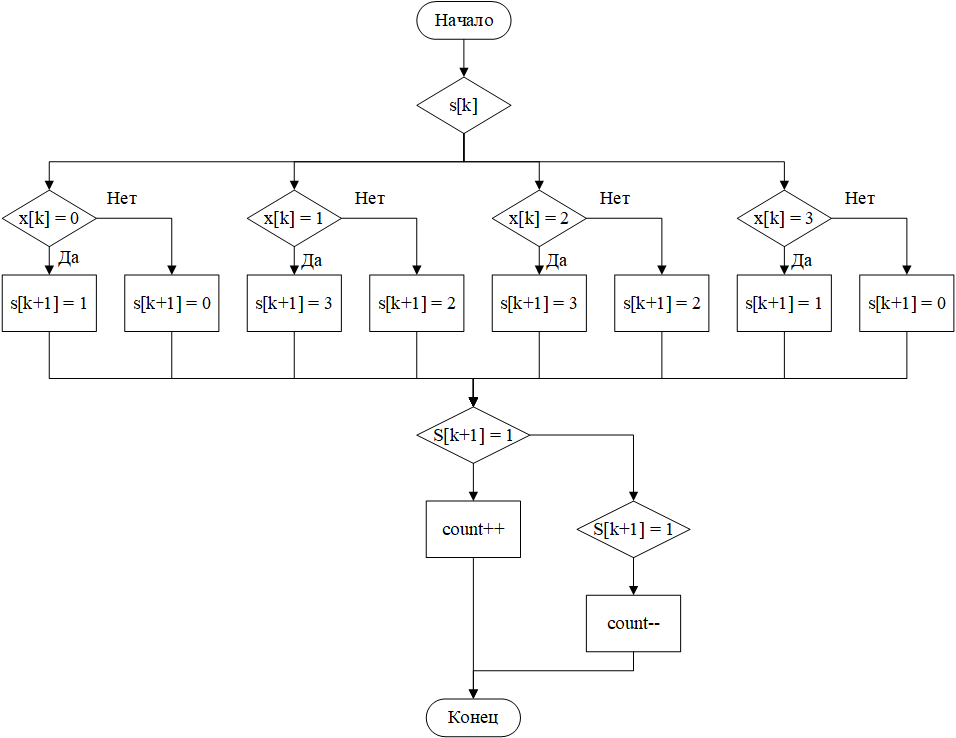


Рис. 18. Блок-схема автомата

**Описание программы:**

Функция **Automat** типа **int** (автомат, распознающий строки):

1. Оператор **switch** (на вход подается элемент массива **s[k]**)
   * **s[k] = 0**:

если **x[k] = 0**, то **s[k+1] = 1**, иначе **s[k+1] = 0**

* + **s[k] = 1**:

если **x[k] = 0**, то **s[k+1] = 3**, иначе **s[k+1] = 2**

* + **s[k] = 2**:

если **x[k] = 0**, то **s[k+1] = 3**, иначе **s[k+1] = 2**

* + **s[k] = 3**:

если **x[k] = 0**, то **s[k+1] = 1**, иначе **s[k+1] = 0**

1. Если **s[k+1] = 1**, то значение переменной **count** увеличивается на 1, иначе уменьшается на 1
2. Функция завершается командой **return**, функция возвращает значение переменной **count**

Пошаговый режим работы программы:

1. Пользователь вводит строку в поле **textBox2**
2. Значение **textBox2** записывается в переменную **sx** типа **String^**
3. Значение переменной **sx** типа **String^** преобразуется в тип **string** и записывается в переменную **str\_x** типа **string**
4. Инициализируются строки **str\_s**, **str\_true**, **str\_false**, изначально они пустые
5. Введенная пользователем строка (**str\_x**) проверяется на корректность. Если программа находит в строке пробел, то в **textBox4** выводится сообщение «*Ошибка! Введено больше одной строки*». Если программа находит в строке какие-либо посторонние символы не из алфавита языка (не **0** или **1**), то в **textBox4** выводится сообщение «*Ошибка! В строке содержатся символы, не входящие в алфавит*»
6. Инициализируются переменные **i**, **j**, **count** (им присваивается значение **0**), строка **str\_result** (изначально она пуста), массивы **x[]** и **s[]** (размером в длину строки **str\_x + 1**)
7. Символьная строка **str\_x** переводится в числовой массив **x[]**, с каждым символом значения переменных **i** и **j** увеличивается на **1**
8. Для всех элементов массива выполняется функция **Automat**, ее выходное значение записывается в переменную count
9. Массив **x[]** переводится в строку **str\_result**
10. Массив **s[]** переводится в строку **str\_s**
11. Если значение переменной count равно нулю, то значение строки **str\_x** записывается в строку **str\_true**, иначе – в строку **str\_false**
12. Массивы **s[]** и **x[]** удаляются, память высвобождается
13. Значение переменных **str\_true**, **str\_false** и **str\_s** типа **string** переводится в тип **String^** и записывается в переменные **str\_t**, **str\_f** и **str\_automat** соответственно
14. Строки **str\_true**, **str\_false** и **str\_s** выводятся в поля **textBox4**, **textBox1** и **textBox3** соответственно

Автоматический режим работы программы:

1. Инициализируется переменная n, ей присваивается значение 0
2. Пользователь вводит строки через пробел в поле **textBox2**
3. Значение **textBox2** записывается в переменную **sx** типа **String^**
4. Значение переменной **sx** типа **String^** преобразуется в тип **string** и записывается в переменную **str\_x** типа **string**
5. Инициализируются строки **str\_true**, **str\_false**, изначально они пустые
6. Подсчитывается количество строк: если встречается пробел или нуль-символ (конец строки) – значение переменной n увеличивается на 1
7. Введенная пользователем строка (**str\_x**) проверяется на корректность. Если программа находит в строке 2 пробела подряд, то в **textBox4** выводится сообщение «*Ошибка! Введен лишний пробел*». Если программа находит в строке какие-либо посторонние символы не из алфавита языка (не **0** или **1**), то в **textBox4** выводится сообщение «*Ошибка! В строке содержатся символы, не входящие в алфавит*»

Для каждой введенной строки выполняются действия 8-14:

1. Инициализируются переменные **i**, **j**, **count** (им присваивается значение **0**), строка **str\_result** (изначально она пуста), массивы **x[]** и **s[]** (размером в длину строки **str\_x + 1**)
2. Символьная строка **str\_x** переводится в числовой массив **x[]**, с каждым символом значения переменных **i** и **j** увеличивается на **1**
3. Для всех элементов массива выполняется функция **Automat**, ее выходное значение записывается в переменную count
4. Массив **x[]** переводится в строку **str\_result**
5. Если значение переменной count равно нулю, то значение строки **str\_x** записывается в строку **str\_true**, иначе – в строку **str\_false**
6. Массивы **s[]** и **x[]** удаляются, память высвобождается
7. Значение переменной i увеличивается на 1
8. Значение переменных **str\_true** и **str\_false** типа **string** переводится в тип **String^** и записывается в переменные **str\_t** и **str\_f** соответственно
9. Строки **str\_true** и **str\_false** выводятся в поля **textBox4** и **textBox1** соответственно
10. Демонстрация работы программы

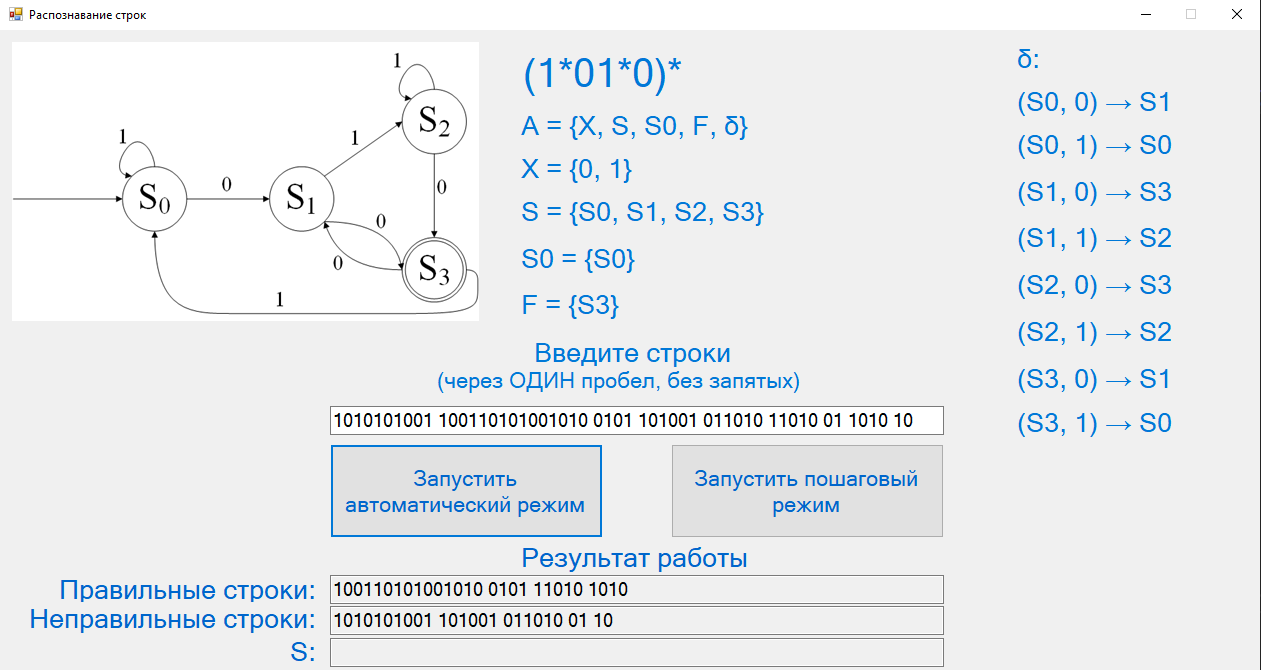


Рис. 19. Работа программы в автоматическом режиме

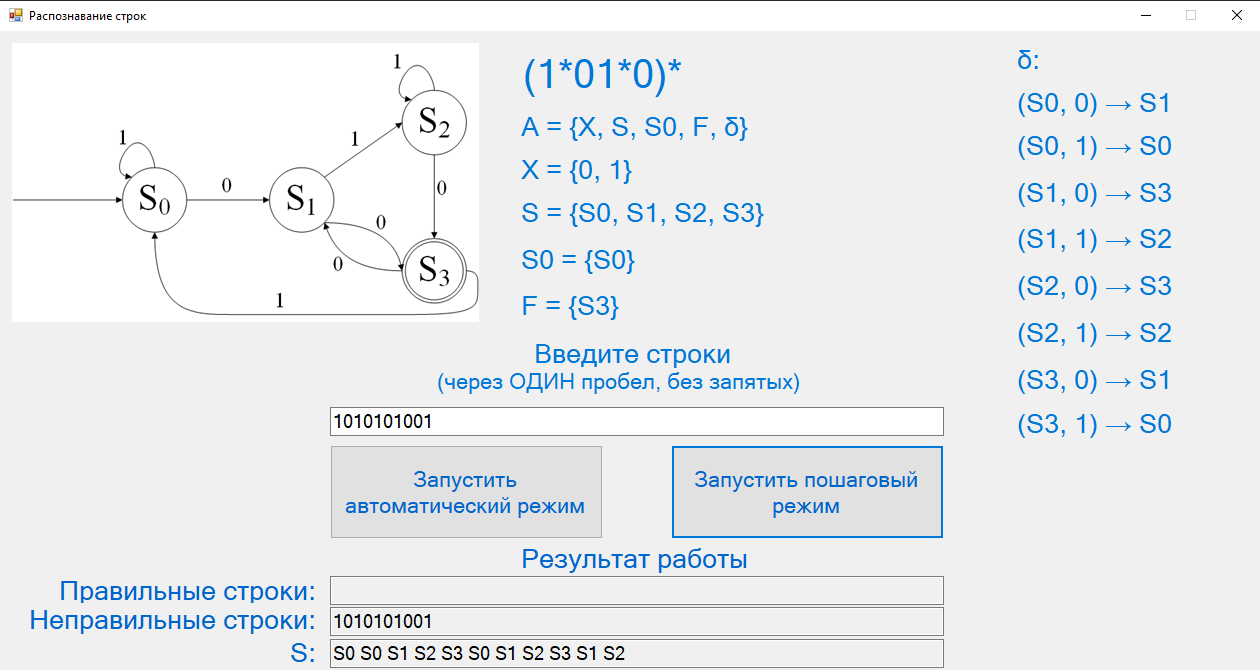


Рис. 20. Работа программы в пошаговом режиме

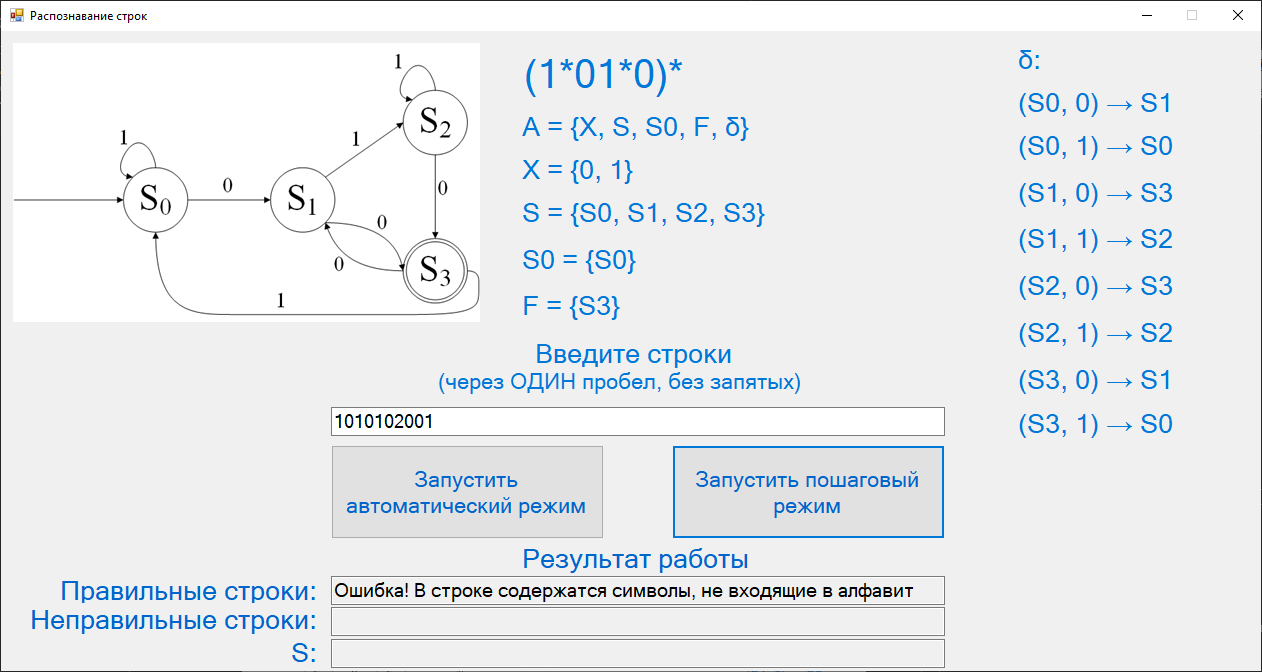


Рис. 21. Ошибка при вводе символа не из алфавита

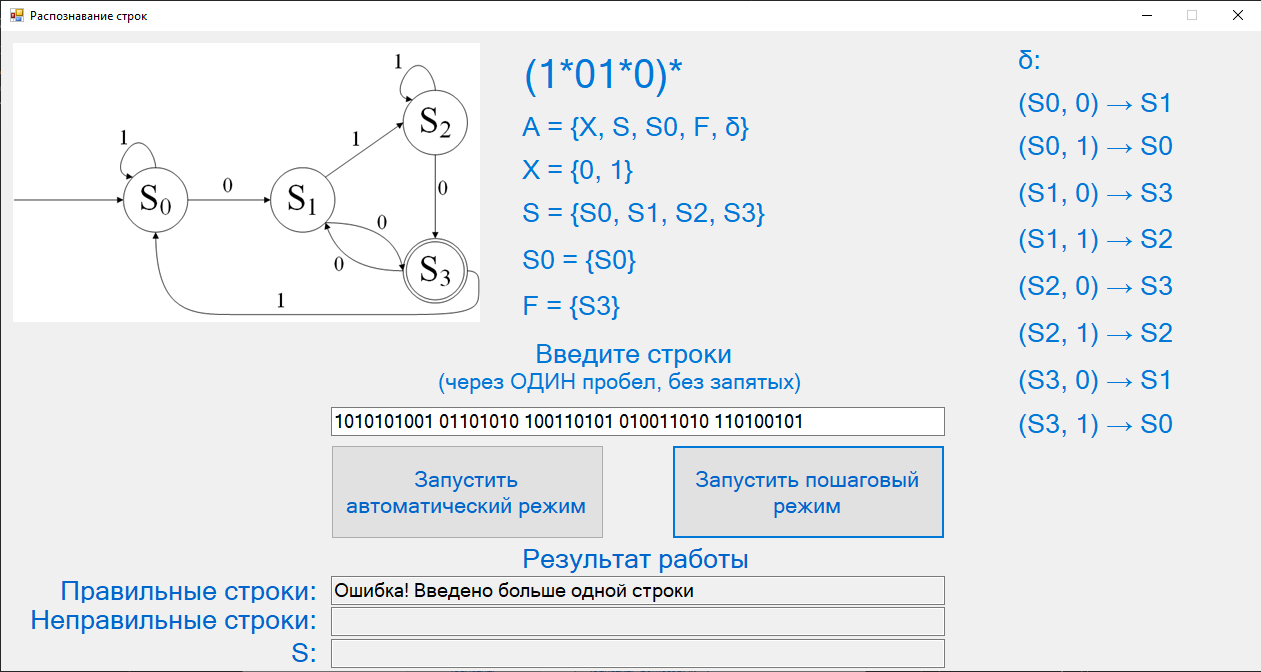


Рис. 22. Ошибка при вводе нескольких строк в пошаговом режиме

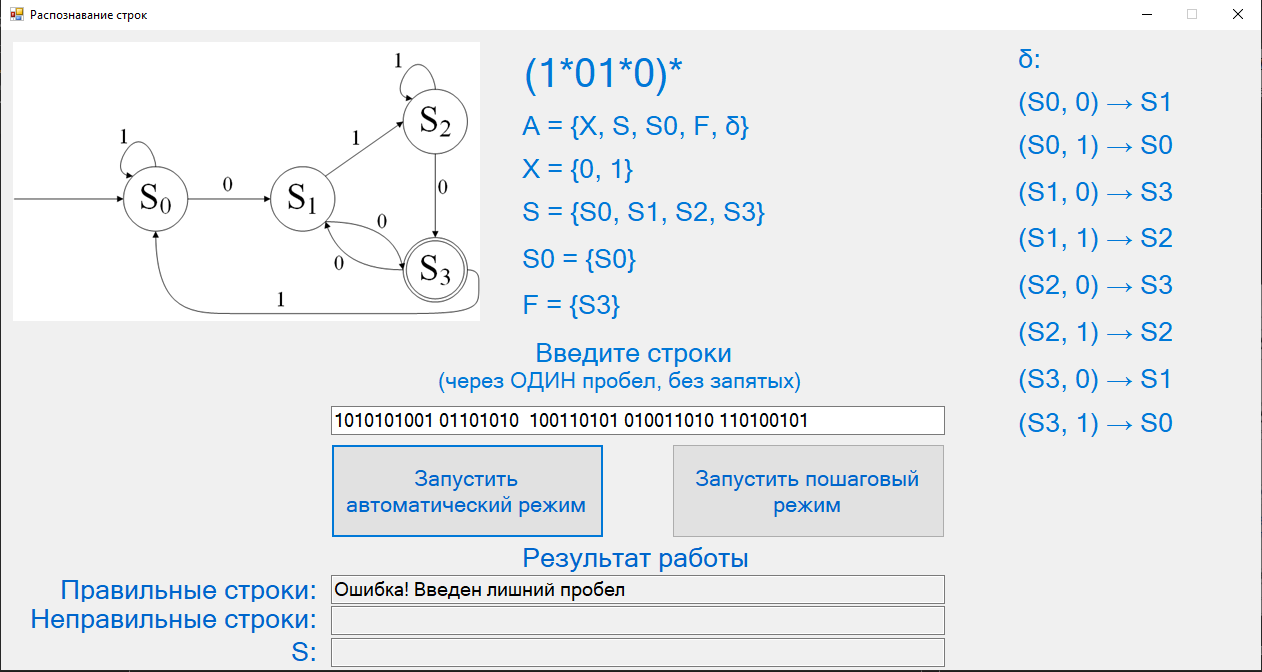


Рис. 23. Ошибка при неправильном формате ввода строк (с несколькими пробелами между ними)

1. Текст программы

int Automat(int x[], int s[], int k, int count) {

switch (s[k]) {

case 0: {

if (x[k] == 0) s[k + 1] = 1;

else s[k + 1] = 0;

}

break;

case 1: {

if (x[k] == 0) s[k + 1] = 3;

else s[k + 1] = 2;

}

break;

case 2: {

if (x[k] == 0) s[k + 1] = 3;

else s[k + 1] = 2;

}

break;

case 3: {

if (x[k] == 0) s[k + 1] = 1;

else s[k + 1] = 0;

}

break;

}

if (s[k + 1] == 1) count++;

else if (s[k + 1] == 3) count--;

return count;

}

#pragma endregion

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

setlocale(LC\_ALL, "Russian");

String^ sx = textBox2->Text;

string str\_x = marshal\_as<string>(sx), str\_s = "", str\_true = "", str\_false = "";

for (int i = 0; i < str\_x.length() + 1; i++) {

if (str\_x[i] == ' ') {

textBox4->Text = "Ошибка! Введено больше одной строки";

textBox1->Text = "";

textBox3->Text = "";

return;

}

if (str\_x[i] != '0' && str\_x[i] != '1' && str\_x[i] != ' ' && str\_x[i] != '\0') {

textBox4->Text = "Ошибка! В строке содержатся символы, не входящие в алфавит";

textBox1->Text = "";

textBox3->Text = "";

return;

}

}

int i = 0, j = 0;

string str\_result = "";

int count = 0;

int\* x = new int[str\_x.length() + 1];

int\* s = new int[str\_x.length() + 1];

s[0] = 0;

j = 0;

while (str\_x[i] != ' ' && str\_x[i] != '\0') {

if (str\_x[i] == '0') x[j] = 0;

else x[j] = 1;

i++;

j++;

}

for (int k = 0; k < j; k++) {

count = Automat(x, s, k, count);

}

for (int k = 0; k < j; k++) {

if (x[k] == 0) str\_result += "0";

else str\_result += "1";

}

for (int k = 0; k <= j; k++) {

if (s[k] == 0) str\_s += "S0 ";

else if (s[k] == 1) str\_s += "S1 ";

else if (s[k] == 2) str\_s += "S2 ";

else if (s[k] == 3) str\_s += "S3 ";

}

str\_result += " ";

if (count == 0) str\_true += str\_result;

else str\_false += str\_result;

delete[] x;

delete[] s;

i++;

String^ str\_t = marshal\_as<String^>(str\_true);

String^ str\_f = marshal\_as<String^>(str\_false);

String^ str\_automat = marshal\_as<String^>(str\_s);

textBox4->Text = str\_t;

textBox1->Text = str\_f;

textBox3->Text = str\_automat;

}

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

setlocale(LC\_ALL, "Russian");

int n = 0;

String^ sx = textBox2->Text;

string str\_x = marshal\_as<string>(sx), str\_true = "", str\_false = "";

for (int i = 0; i < str\_x.length() + 1; i++) {

if (str\_x[i] == ' ' || str\_x[i] == '\0') n++;

if (str\_x[0] == ' ' || (str\_x[i] == ' ' && str\_x[i - 1] == ' ')) {

textBox4->Text = "Ошибка! Введен лишний пробел";

textBox1->Text = "";

textBox3->Text = "";

return;

}

if (str\_x[i] != '0' && str\_x[i] != '1' && str\_x[i] != ' ' && str\_x[i] != '\0') {

textBox4->Text = "Ошибка! В строке содержатся символы, не входящие в алфавит";

textBox1->Text = "";

textBox3->Text = "";

return;

}

}

int i = 0, j = 0;

for (int m = 1; m <= n; m++) {

string str\_result = "";

int count = 0;

int\* x = new int[str\_x.length() + 1];

int\* s = new int[str\_x.length() + 1];

s[0] = 0;

j = 0;

while (str\_x[i] != ' ' && str\_x[i] != '\0') {

if (str\_x[i] == '0') x[j] = 0;

else x[j] = 1;

i++;

j++;

}

for (int k = 0; k < j; k++) {

count = Automat(x, s, k, count);

}

for (int k = 0; k < j; k++) {

if (x[k] == 0) str\_result += "0";

else str\_result += "1";

}

str\_result += " ";

if (count == 0) str\_true += str\_result;

else str\_false += str\_result;

delete[] x;

delete[] s;

i++;

}

String^ str\_t = marshal\_as<String^>(str\_true);

String^ str\_f = marshal\_as<String^>(str\_false);

textBox4->Text = str\_t;

textBox1->Text = str\_f;

textBox3->Text = "";

}

};

}

Вывод

В ходе выполнения лабораторной работы были решены задания из учебника Ю. Г. Карпова «Теория автоматов»: построены регулярные выражения, а также конечные автоматы, их распознающие. Также были построены графы для этих конечных автоматов.

Было построено регулярное выражение, состоящее из символов 0 и 1, задающее множество слов, в которых четное число символов 0. Был построен конечный автомат для распознавания данного выражения и граф. Конечный автомат был смоделирован с помощью языка программирования C++.

Характеристики построенного автомата:

1. Детерменированный – следующее состояние определяется однозначно текущим состоянием и входным символом. Функция переходов имеет только одно результирующее состояние
2. Абстрактный – математическая модель дискретного устройства, которое в каждый момент времени находится в каком-либо одном состоянии из множества возможных
3. Конечный – число состояний конечно
4. Асинхронный – работает над последовательностью при ее поступлении на вход, что может произойти с различными интервалами времени
5. Распознаватель – автомат распознает строки и принимает либо отвергает их