**Федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

«Санкт-Петербургский государственный университет»

Факультет прикладной математики - процессов управления

Курсовая работа

по дисциплине

“Теория автоматов и формальных языков”

Выполнил: Голубев Роман Андреевич

студент 2 курса

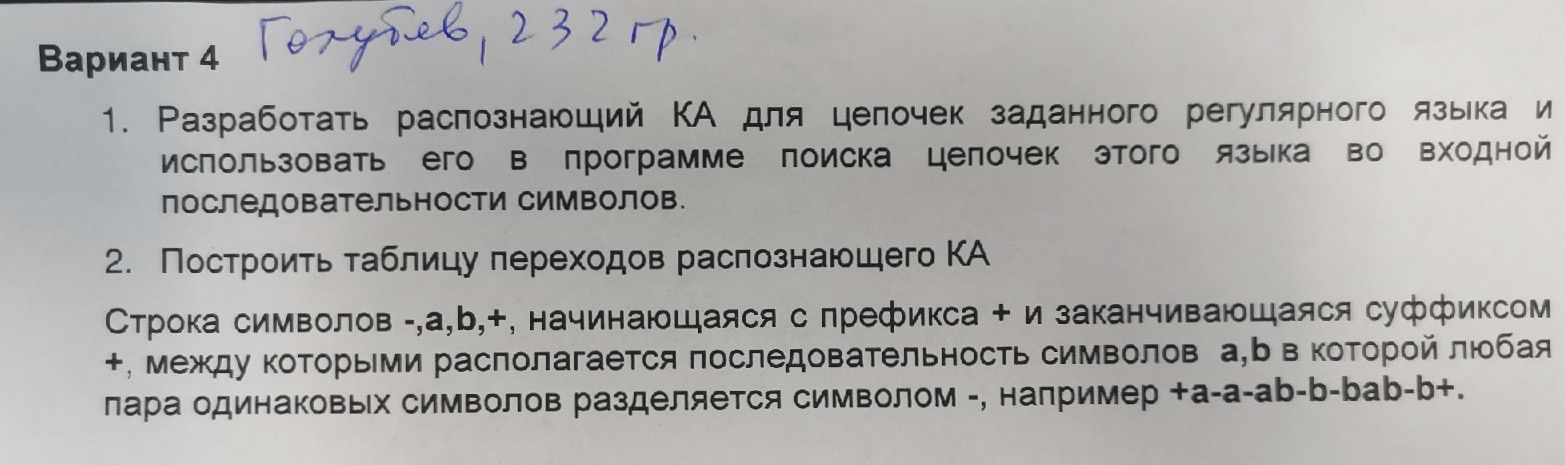
группа 232

Преподаватель: Матросов Александр Васильевич

Санкт-Петербург

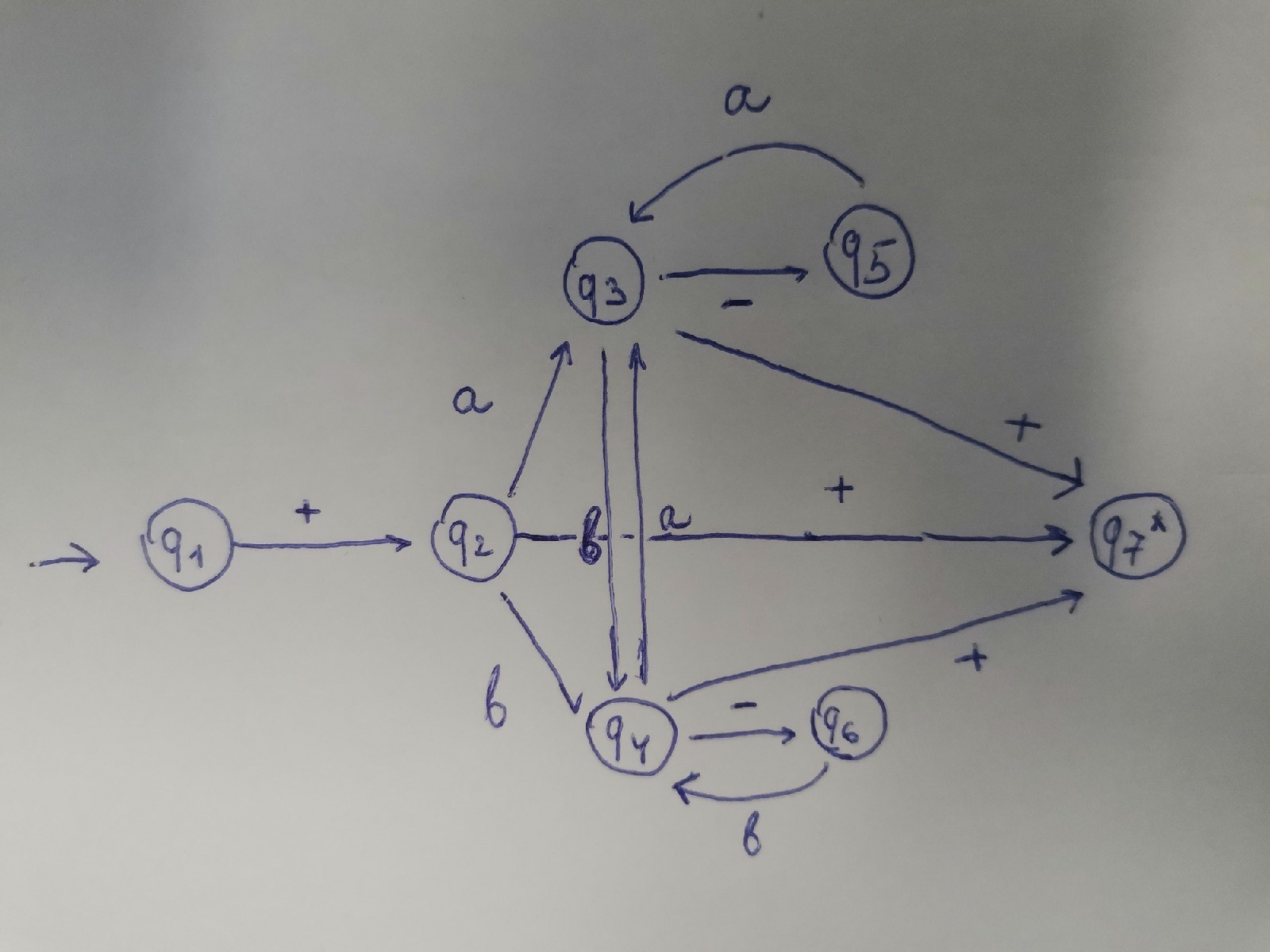
2019

1. Постановка задачи.



1. График НКА

Ниже приведен рисунок.



Регулярное выражение(заменил символ + на с):

с(E + b(-b)\* + a((-a)\* + b(-b)\*a)\* + a((-a)\* + b(-b)\*a)\*b(-b)\* +

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

1. Таблица переходов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | + | - | a | b |
| q1 | q2 | Ø | Ø | Ø |
| q2 | q7 | Ø | q3 | q4 |
| q3 | q7 | q5 | Ø | q4 |
| q4 | q7 | q6 | q3 | Ø |
| q5 | Ø | Ø | q5 | Ø |
| q6 | Ø | Ø | Ø | q4 |
| \*q7 | Ø | Ø | Ø | Ø |

1. Код программы.

Код написан на C++.

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

char q1(char\* arr, int i);

char q2(char\* arr, int i);

char q3(char\* arr, int i);

char q4(char\* arr, int i);

char q5(char\* arr, int i);

char q6(char\* arr, int i);

char q7(char\* arr, int i);

// создаем функции для каждого состояния и после чего определяем их в силу синтаксиса C++

char q7(char\* arr, int i)

{

return i;}

char q2(char\* arr, int i)

{

if(arr[i] == 'a')

{return q3(arr,i+1);}

else if (arr[i] == 'b')

{ return q4(arr, i+1);}

else if(arr[i] == '+')

{return q7(arr, i+1);}

return -1;

}

char q1(char\* arr, int i)

{

if(arr[i] == '+')

return q2(arr, i+1);

return -1;

}

char q3(char\* arr, int i)

{

if(arr[i] == '-')

{return q5( arr, i+1);}

else if(arr[i] == 'b')

{return q4(arr, i+1);}

else if(arr[i] == '+')

{return q7(arr, i+1);}

return -1;

}

char q4(char\* arr, int i)

{

if(arr[i] == '-')

{return q6(arr, i+1);}

else if(arr[i] == 'a')

{return q3(arr, i+1);}

else if(arr[i] == '+')

{return q7(arr, i+1);}

return -1;

}

char q5(char\* arr, int i)

{

if(arr[i] == 'a')

return q3(arr, i+1);

return -1;

}

char q6(char\* arr, int i)

{

if(arr[i] == 'b')

return q4(arr, i+1);

return -1;

}

int main() {

string chain;

cin >> chain; // вводим нужную нам цепочку символов

char sym[chain.size()+1];

strcpy(sym, chain.c\_str()); // посимвольно выгружаем цепь в массив

// И дальше для каждого символа в цепи проверяем, не начинается ли нужная нам цпеь с данного символа

for (int first = 0; first < chain.size()+1; ++first) {

if (q1(sym, first) != -1 && sym[first] == ‘+’)

{ cout << first << ‘:’;

int last = q1(sym, first);

for (int p = first; p < last; ++p)

cout << sym[p];

cout << endl;}

} return 0;}

1. Описание работы программы.

Программа сначала получает на вход строчку из символов, после чего посимвольно загружает в массив. На каждое состояние КА есть функция, и для каждого плюса в массиве происходит проверка, является ли этот плюс начал нужной нам цепочки. Если на какой-то функции становится понятно, что это неподходящая цепочка, то функция возвращает “-1” и данная цепочка не выведется. Если все же мы доходим до состояния q7, то данная цепочка выводится.

1. Тесты

|  |  |
| --- | --- |
| Ввод | Вывод |
| ++ | 0:++ |
| +-+ |  |
| +ab+ | 0:+ab+ |
| +aab+ |  |
| +a-a-ab-b-bab-b+ | 0:+a-a-ab-b-bab-b+ |
| +aba+bab+ | 0:+aba+ ;4:+bab+ |
| +aba-bab+ |  |
| +bab-bab-ba-ab+ | 0:+bab-bab-ba-ab+ |
| +++ | 0:++ ;1:++ |
| +ecds+ |  |
| +a+b+a+ | 0:+a+ ;2: +b+ ;4:+a+ |
| +s+bab+ | 2:+bab+ |
| +f+aba+g+ | 2:+aba+ |
| +a-a-a-a-a-a-a-a-a-a-aba+ | 0:+a-a-a-a-a-a-a-a-a-a-aba+ |
| +a-a-a-a-a-a-a-aba-ab-babab-dfa+ |  |