

## **Домашнее задание № 1.**

### **Проектирование и реализация конечного распознавателя.**

#### **Вариант №1.3**

Выполнила Домбрина Алёна Игоревна (КТбо1-6).

#### **№1. Постановка задачи:**

1. Построить детерминированный конечный автомат с входным алфавитом  $\{a, b\}$ , допускающий множество цепочек длиной не менее двух символов, начинающихся и заканчивающихся одинаковой парой символов.
2. Разработать программу, реализующую разработанный конечный распознаватель.

#### **№2. Словесное описание автомата**

##### **1. Описание состояний автомата:**

$q_0$  – начальное состояние;

$q_1$  – если первый символ начальной пары  $a$  (N);

$q_2$  - если второй символ начальной пары, от исходной ( $a\_$ )  $a$ , если последующий символ  $a$ , автомат будет оставаться в этом состоянии (Y);

$q_3$  – если следующий символ от исходной последовательности  $b$  по ветке пары ( $aa$ ) (N);

$q_4$  - если следующий символ от исходной последовательности  $a$  по ветке пары ( $aa$ ) (N);

$q_5$  - если следующий символ от исходной последовательности  $a$ , стоящий перед символом  $a$ , по ветке пары ( $aa$ ) (Y);

q6 – если второй символ начальной пары, от исходной (a\_) b (Y);

q7 - если следующий символ от исходной последовательности a по ветке пары (ab) (N);

q8 - если следующий символ от исходной последовательности b по ветке пары (ab) (N);

q9 - если следующий символ от исходной последовательности b, стоящий перед символом a, по ветке пары (ab) (Y);

q10 - если первый символ начальной пары b (N);

q11 - если второй символ начальной пары, от исходной (b\_) b, если последующий символ b, автомат будет оставаться в этом состоянии (Y);

q12 - если следующий символ от исходной последовательности a по ветке пары (bb) (N);

q13 - если следующий символ от исходной последовательности b по ветке пары (bb) (N);

q14 - если следующий символ от исходной последовательности b, стоящий перед символом b, по ветке пары (bb) (Y);

q15 - если второй символ начальной пары, от исходной (b\_) a (Y);

q16 - если следующий символ от исходной последовательности b по ветке пары (ba) (N);

q17 - если следующий символ от исходной последовательности a по ветке пары (ba) (N);

q18 - если следующий символ от исходной последовательности a, стоящий перед символом b, по ветке пары (ba) (Y);

## **2. Описание работы автомата:**

Изначально автомат находится в начальном состоянии q0. После считывания первого символа автомат переходит либо в состояние q1 по символу a, либо в состояние q10 по символу b. Далее, либо в состояние q2, либо q6, либо q11, либо q15. Таким образом, автомат без памяти сможет запомнить первую пару, так как для каждой из 6 пар будет своя ветка. Далее автомат может корректно закончить свою работу в состояниях q2, q6, q11, q15, так как цепочки ab, ba, aa, bb являются подходящими под условие, либо продолжить свою работу, оставшись в состояниях q2 или q11 (если входная цепочка состоит только из символов a или b), или перейдя в промежуточные состояния q3, q7, q8, q16, q17, q12, которые не являются допускающими. Из них в состояния q4, q9, q13, q18, из которых q9, q18 являются допускающими (если цепочка начинается и заканчивается парой ab и ba соответственно). Из q4 и q13 в q5 и q14 (если цепочка начинается и заканчивается парой aa и bb соответственно).

## **№3. Описание распознавателя:**

### **1. Описание в виде пятерки множеств:**

$$V=\{a,b\}$$

$$K=\{q0, q1, q2, q3, \dots, q17, q18\}$$

M:

$M[q0,a]=q1$	$M[q14,a]=q12$
$M[q0,b]=q10$	$M[q14,b]=q14$
$M[q1,a]=q2$	$M[q15,a]=q17$
$M[q1,b]=q6$	$M[q15,b]=q16$
$M[q2,a]=q2$	$M[q16,a]=q18$
$M[q2,b]=q3$	$M[q16,b]=q16$
$M[q3,a]=q4$	$M[q17,a]=q17$
$M[q3,b]=q3$	$M[q17,b]=q16$
$M[q4,a]=q5$	$M[q18,a]=q17$
$M[q4,b]=q3$	$M[q18,b]=q16$
$M[q5,a]=q5$	
$M[q5,b]=q3$	
$M[q6,a]=q7$	
$M[q6,b]=q8$	
$M[q7,a]=q7$	
$M[q7,b]=q9$	
$M[q8,a]=q7$	
$M[q8,b]=q8$	
$M[q9,a]=q7$	
$M[q9,b]=q8$	
$M[q10,a]=q15$	
$M[q10,b]=q11$	
$M[q11,a]=q12$	
$M[q11,b]=q11$	
$M[q12,a]=q12$	
$M[q12,b]=q13$	
$M[q13,a]=q12$	
$M[q13,b]=q14$	

$S=\{q_0\}$

$Z=\{q_2, q_5, q_6, q_9, q_{11}, q_{14}, q_{15}, q_{18}\}$

**2. Описание в виде таблицы переходов:**

<b>№</b>	<b>a</b>	<b>b</b>	<b>Y/N</b>
q0	q1	q10	N
q1	q2	q6	N
q2	q2	q3	Y
q3	q4	q3	N
q4	q5	q3	N
q5	q5	q3	Y
q6	q7	q8	Y
q7	q7	q9	N
q8	q7	q8	N
q9	q7	q8	Y
q10	q15	q11	N
q11	q12	q11	Y
q12	q12	q13	N
q13	q12	q14	N
q14	q12	q14	Y
q15	q17	q16	Y
q16	q18	q16	N
q17	q17	q16	N
q18	q17	q16	Y

Таблица 3.1 - Таблица переходов

### 3. Описание в виде диаграммы переходов автомата:

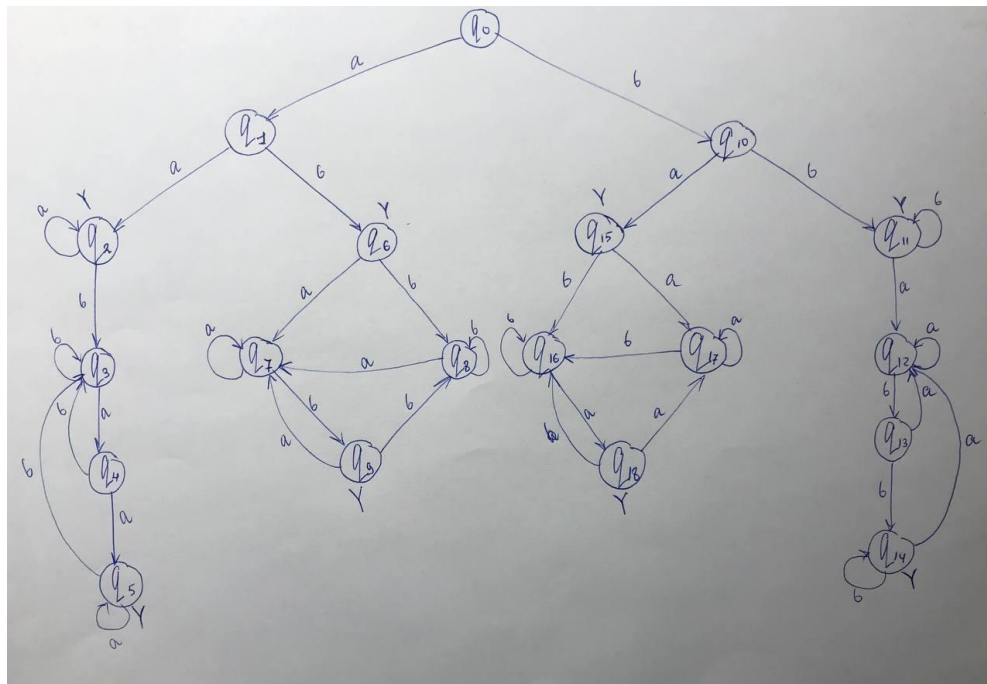


Рисунок 1 - Диаграмма переходов автомата

#### №4. Набор тестов с пошаговой ручной прогонкой:

1. aa

$q_0(a) \rightarrow q_1(a) \rightarrow q_2$  - Y

Ответ: Yes

2. bbb

$q_0(b) \rightarrow q_{10}(b) \rightarrow q_{11}(b) \rightarrow q_{11}$  - Y

Ответ: Yes

3. aabbaa

$q_0(a) \rightarrow q_1(a) \rightarrow q_2(b) \rightarrow q_3(b) \rightarrow q_3(a) \rightarrow q_4(a) \rightarrow q_5$  - Y

Ответ: Yes

4. aba

$q_0(a) \rightarrow q_1(b) \rightarrow q_6(a) \rightarrow q_7$  - N

Ответ: No

5. abbabab

$q_0(a) \rightarrow q_1(b) \rightarrow q_6(b) \rightarrow q_8(a) \rightarrow q_7(b) \rightarrow q_9(a) \rightarrow q_7(b) \rightarrow q_9$  - Y

Ответ: Yes

6. baaba

$q_0(b) \rightarrow q_{10}(a) \rightarrow q_{15}(a) \rightarrow q_{17}(b) \rightarrow q_{16}(a) \rightarrow q_{18}$  - Y

Ответ: Yes

7. babab

$q_0(b) \rightarrow q_{10}(a) \rightarrow q_{15}(b) \rightarrow q_{16}(a) \rightarrow q_{18}(b) \rightarrow q_{16}$  - N

Ответ: No

8. aabb

$q_0(a) \rightarrow q_1(a) \rightarrow q_2(b) \rightarrow q_3(b) \rightarrow q_3$  - N

Ответ: No

9. bbaababb

$q_0(b) \rightarrow q_{10}(b) \rightarrow q_{11}(a) \rightarrow q_{12}(a) \rightarrow q_{12}(b) \rightarrow q_{13}(a) \rightarrow q_{12}(b) \rightarrow q_{13}(b) \rightarrow q_{14}$  - Y

Ответ: Yes

10. ab

$q_0(a) \rightarrow q_1(b) \rightarrow q_6$  - Y

Ответ: Yes

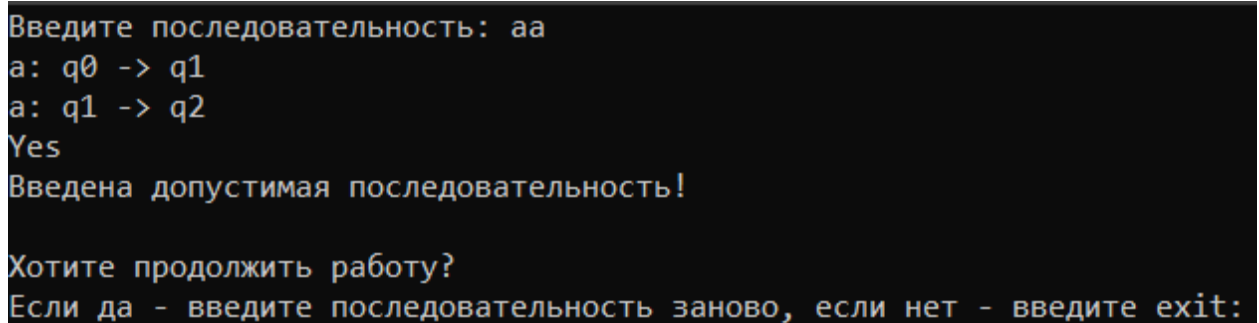
11. ba

$q_0(b) \rightarrow q_{10}(a) \rightarrow q_{15}$  - Y

Ответ: Yes

## №5. Скриншоты выполнения программы на тестовых примерах:

Тест 1:



```
Введите последовательность: aa
a: q0 -> q1
a: q1 -> q2
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 2 – Результаты теста 1

Тест 2:

```
Введите последовательность: bbb
b: q0 -> q10
b: q10 -> q11
b: q11 -> q11
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 3 – Результаты теста 2

Тест 3:

```
Введите последовательность: aabbaa
a: q0 -> q1
a: q1 -> q2
b: q2 -> q3
b: q3 -> q3
a: q3 -> q4
a: q4 -> q5
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 4 – Результаты теста 3

Тест 4:

```
Введите последовательность: aba
a: q0 -> q1
b: q1 -> q6
a: q6 -> q7
No
Введена недопустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 5 – Результаты теста 4



Тест 5:

```
Введите последовательность: abbabab
a: q0 -> q1
b: q1 -> q6
b: q6 -> q8
a: q8 -> q7
b: q7 -> q9
a: q9 -> q7
b: q7 -> q9
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 6 – Результаты теста 5

Тест 6:

```
Введите последовательность: baaba
b: q0 -> q10
a: q10 -> q15
a: q15 -> q17
b: q17 -> q16
a: q16 -> q18
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 7 – Результаты теста 6

Тест 7:

```
Введите последовательность: babab
b: q0 -> q10
a: q10 -> q15
b: q15 -> q16
a: q16 -> q18
b: q18 -> q16
No
Введена недопустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 8 – Результаты теста 7

Тест 8:

```
Введите последовательность: aabb
a: q0 -> q1
a: q1 -> q2
b: q2 -> q3
b: q3 -> q3
No
Введена недопустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 9 – Результаты теста 8

Тест 9:

```
Введите последовательность: bbaababb
b: q0 -> q10
b: q10 -> q11
a: q11 -> q12
a: q12 -> q12
b: q12 -> q13
a: q13 -> q12
b: q12 -> q13
b: q13 -> q14
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 10 - Результаты теста 10

Тест 10:

```
Введите последовательность: ab
a: q0 -> q1
b: q1 -> q6
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 11 – Результаты теста 10

Тест 11:

```
Введите последовательность: ba
b: q0 -> q10
a: q10 -> q15
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 12 – Результаты теста 11

### **Проверка тестов на некорректный ввод:**

Тест на строку меньше двух символов (невозможно составить пару):

```
Введите последовательность: a
Вы ввели слишком короткую последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 13 - Тест на строку меньше двух символов

Тест на пустую строку:

```
Введите последовательность:
Вы ввели пустую последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 14 - Тест на пустую строку

Тест на строку, не соответствующую входному алфавиту:

```
Введите последовательность: ababayaba
Вы использовали недопустимые символы!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit:
```

Рисунок 15 - Тест на строку, не соответствующую входному алфавиту

## Взаимодействие пользователя с программой:

```
Если да - введите последовательность заново, если нет - введите exit:
Вы ввели пустую последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit: b
Вы ввели слишком короткую последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit: abababb
a: q0 -> q1
b: q1 -> q6
a: q6 -> q7
b: q7 -> q9
a: q9 -> q7
b: q7 -> q9
b: q9 -> q8
No
Введена недопустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit: abab
a: q0 -> q1
b: q1 -> q6
a: q6 -> q7
b: q7 -> q9
Yes
Введена допустимая последовательность!

Хотите продолжить работу?
Если да - введите последовательность заново, если нет - введите exit: exit
Вы завершили работу! Спасибо за внимание!
```

Рисунок 16 - Взаимодействие пользователя с программой

## №6. Описание структуры данных:

Структура данных называется recognizer. Она содержит одно поле – table.  
– символ(char), по которому автомат переходит, элемент – состояние(string),  
в которое переходит.

```

struct recognizer
{
    map<string, map<char, string>> table =
    {
        {"q0", { {'a', "q1"}, {'b', "q10"} } },
        {"q1", { {'a', "q2"}, {'b', "q6"} } },
        {"q2", { {'a', "q2"}, {'b', "q3"} } },
        {"q3", { {'a', "q4"}, {'b', "q3"} } },
        {"q4", { {'a', "q5"}, {'b', "q3"} } },
        {"q5", { {'a', "q5"}, {'b', "q3"} } },
        {"q6", { {'a', "q7"}, {'b', "q8"} } },
        {"q7", { {'a', "q7"}, {'b', "q9"} } },
        {"q8", { {'a', "q7"}, {'b', "q8"} } },
        {"q9", { {'a', "q7"}, {'b', "q8"} } },
        {"q10", { {'a', "q15"}, {'b', "q11"} } },
        {"q11", { {'a', "q12"}, {'b', "q11"} } },
        {"q12", { {'a', "q12"}, {'b', "q13"} } },
        {"q13", { {'a', "q12"}, {'b', "q14"} } },
        {"q14", { {'a', "q12"}, {'b', "q14"} } },
        {"q15", { {'a', "q17"}, {'b', "q16"} } },
        {"q16", { {'a', "q18"}, {'b', "q16"} } },
        {"q17", { {'a', "q17"}, {'b', "q16"} } },
        {"q18", { {'a', "q17"}, {'b', "q16"} } }
    };
};

```

Рисунок 17 - Структура для хранения таблицы

## №7. Описание идеи программной реализации одного шага работы автомата:

Пользователю предлагается ввести последовательность. Далее последовательность проходит несколько проверок: на пустоту введенной строки, на размер строки меньше двух символов (т. к. невозможно сформировать пару), на строку с недопустимым алфавитом. Если последовательность попадает в какое-либо из этих условий, то программа выведет эту ошибку и предлагает ввести последовательность заново, либо выйти. Если же последовательность соответствует символам входного алфавита, то тогда вызывается функция, параметрами которой является введенная последовательность и таблица переходов.

Функция `output_of_transition` определяет текущее состояние в `q0` и создает следующее состояние, после чего функция посимвольно проходит по последовательности. Следующему состоянию присваивается состояние, в которое автомат пришел после чтения очередного символа. Затем функция выводит текущий символ последовательности, текущее состояние и состояние, в которое перейдет автомат. Далее текущему состоянию присваивается следующее состояние. После того, как автомат завершит проход по последовательности, функция делает проверку на соответствие текущего состояния на допустимость. Если текущее состояние равно допускающему состоянию, то автомат возвращает `true`, иначе `false`. Если в `main` было возвращено состояние `true`, автомат выводит сообщение о том, что была введена допустимая последовательность, иначе сообщение о том, что не допустимая.

Разберем один шаг работы автомата на примере:

Введена последовательность `abab`. Создается таблица переходов, которая вместе с введенной последовательностью передается в функцию `output_of_transition`. Она определяет текущее состояние в `q0` и создает следующее состояние. Заходит в цикл, пробегающий по всем элементам последовательности: Следующее состояние равно элементу из таблицы переходов, который находится по ключу текущего состояния и текущего элемента последовательности(`q1`). Далее выводится текущий элемент последовательности(`a`), текущее состояние(`q0`), следующее состояние(`q1`), текущему состоянию присваивается следующее.

## **№8. Листинг программы:**

```
//ЮФУ, ИКТИБ, МОП ЭВМ
//Программирование и основы теории алгоритмов часть 2
//Домашнее задание №1 - ДКА
//КТб01-6, Домбрина Алёна Игоревна
//Вариант 1.3
//24.03.24
```

```

#include <iostream>
#include <map>
#include <string>

using namespace std;

//Входные данные: введенная последовательность
//Функция valid_input проверяет вводимую последовательность на соответствие
входному алфавиту
//Если все элементы из алфавита {a, b} - функция возвращает true, если нет - false

//Входные данные: введенная последовательность и таблица переходов
//Функция output_of_transitions из исходного состояния q0 проходит по введенной
последовательности и выводит состояния,
//по которым проходит автомат.
//Если автомат останавливается в допускаящем состоянии - функция возвращает true, в
недопускающем - false
bool output_of_transitions(string , struct recognizer&);

//Структура recognizer содержит в себе поле table, которое представляет собой структуру
данных, хранящую состояния и переходы по ним
//Структура данных основана на контейнере map, в котором ключ - состояние, элемент -

//в котором ключ - символ, по которому мы переходим, элемент - состояние, в которое
переходим.
struct recognizer
{
    map<string, map<char, string>> table =
    {
        {"q0", { {'a', "q1"}, {'b', "q10"} } },
        {"q1", { {'a', "q2"}, {'b', "q6"} } },
        {"q2", { {'a', "q2"}, {'b', "q3"} } },
        {"q3", { {'a', "q4"}, {'b', "q3"} } },
        {"q4", { {'a', "q5"}, {'b', "q3"} } },
        {"q5", { {'a', "q5"}, {'b', "q3"} } },
        {"q6", { {'a', "q7"}, {'b', "q8"} } },
        {"q7", { {'a', "q7"}, {'b', "q9"} } },
        {"q8", { {'a', "q7"}, {'b', "q8"} } },
        {"q9", { {'a', "q7"}, {'b', "q8"} } },
        {"q10", { {'a', "q15"}, {'b', "q11"} } },
        {"q11", { {'a', "q12"}, {'b', "q11"} } },
        {"q12", { {'a', "q12"}, {'b', "q13"} } },
        {"q13", { {'a', "q12"}, {'b', "q14"} } },
        {"q14", { {'a', "q12"}, {'b', "q14"} } },
        {"q15", { {'a', "q17"}, {'b', "q16"} } },
        {"q16", { {'a', "q18"}, {'b', "q16"} } },
        {"q17", { {'a', "q17"}, {'b', "q16"} } },
        {"q18", { {'a', "q17"}, {'b', "q16"} } }
    };
};

```



```

int main()
{
    setlocale(LC_ALL, "Russian");
    string sequence;

    "Введите последовательность:" << endl;
    getline(cin, sequence);
    if (sequence == "exit")

        "Вы завершили работу! Спасибо за внимание!" << endl;

    "Вы ввели пустую последовательность!" << endl;
    "Хотите продолжить работу?" << endl << "Если да - введите последовательность заново,
если нет - введите exit:" << endl;
    if (sequence.empty())
        goto choice;
    else if (invalid_input(sequence))
        goto choice;

    "Вы использовали недопустимые символы!" << endl;
    "Хотите продолжить работу?" << endl << "Если да - введите последовательность заново,
если нет - введите exit:" << endl;
    if (sequence.size() < 2)
    {
        goto choice;
    }

    "Вы ввели слишком длинную последовательность!" << endl;
    "Хотите продолжить работу?" << endl << "Если да - введите последовательность заново,
если нет - введите exit: ";
    {
        recognizer automat;
        if (output_of_transitions(sequence, automat))

            goto choice;
        "Введена допустимая последовательность!" << endl;
        "Хотите продолжить работу?" << endl << "Если да - введите последовательность заново,
если нет - введите exit: ";
        cout << "No" << endl;
    }
}

```

```

"Введена недопустимая последовательность!" << endl;
"Хотите продолжить работу?" << endl << "Если да - введите последовательность заново,
если нет - введите exit: ";

```

```

                                goto choice;
                            }
                        }
                    cout << endl;
                } while (true);
            }

bool valid_input(string sequence)
{
    for (char c : sequence)
    {
        if (!strchr("ab", c))
        {

            return true;
        }
    }

bool output_of_transitions(string sequence, struct recognizer& transition_table)
{
    string q_cur = "q0", q_follow;
    for (auto it = sequence.begin(); it != sequence.end(); it++)
    {
        q_follow = transition_table.table[q_cur][*it];
        cout << *it << ": " << q_cur << " -> " << q_follow << endl;
        q_cur = q_follow;
    }
    if (q_cur == "q2" || q_cur == "q5" || q_cur == "q6" || q_cur == "q9" || q_cur == "q11" ||
q_cur == "q14" || q_cur == "q15" || q_cur == "q18")
    {
        return true;
    }
    else
    {
        return false;
    }
}

```