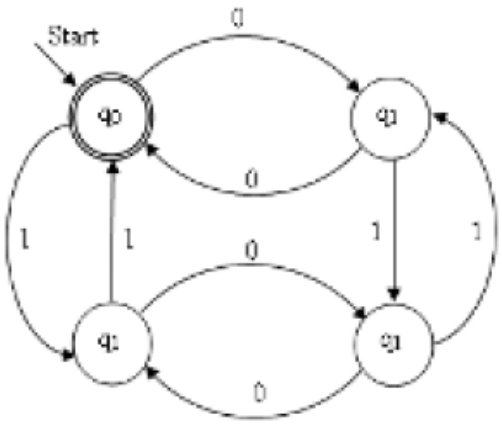
Tema 3 - Automate finite deterministe



- prin tabel:

δ		Σ	
		0	1
Q	q_0	q_1	q_2
	q_1	q_0	q_3
	q_2	q_3	q_0
	q_3	q_2	q_1

Cuvant acceptat: ajung intr-una din starile finale

Cuvant respins: NU ajung in stare finala sau am blocaj

Blocaj

Cerințe:

Să se implementeze un AFD (automat finit determinist) atfel: se citesc din fișier elementele componente ale automatului **Q, Σ, δ, q_0, F**. Se citește de la tastatură un cuvant și se verifică, dacă este acceptat de către automat.

Indicații:

Q {**q0**, q1, q2, q3} NU {0,1,2,3} – multime de stringuri

Sigma {a,b,c} - alfabet/vocabular (unice) - string, sir de caractere char*, char vocabular[100]

Functia de tranzitie: variante posibile de implementare

- matrice de stringuri

- Tranzitie obj (stare_plecare, symbol, stare_sosire) ex q0 0 q1, q2 0 q3, q7 0 $…

- tuple

- unordered_map sau map – alegeți cheia și valoarea în mod convenabil. Atenție la funcția de hashing.

	a	b	c
q0	q1	q2	q3
Q1	Q2	Q1	-
Q2			
Q3			
Q0			

F={q3, q2}multime de stringuri

```
int verifica(string cuvant){  //0 – neeacceptat, -1 blocaj, 1 – acceptat sau bool cu true/false dar
                              //afisez mesaj la  blocaj

stareCurenta=getInitialState();

for( unsigned int index = 0; index<lungime_cuvant; ++i) sau for( auto caracter : cuvant)

{

        If (..)

        stareCurenta=aplicaTranzitie(stareCurenta, caracter);

        //BLOCAJ IF AJUNGI LA STAREA – BREAK, FALSE


}


If (stareCurenta in multimer stari finale)
```

DA

Else

NU

}

Q0, abababaaaaaa (DA, NU, BLOCAJ-NU)

Q1, bbababaaaaaa

Q1, ababaaaaaa

….. **q_CEVA**, lambda

q_CEVA, lambda, se verifica daca q**_CEVA** face parte din F, DA, daca nu, NU

Barem

1. Se cere creearea unei clase AFD (alta decât clasa principală). În funcția principală main se declară un obiect de tip AFD

Membrii clasei vor fi: Stari, Sigma, Delta, StareInit, Finale

Printre metodele clasei obligatoriu:

(1) afisare () - afișarea frumoasă a automatului **0.5p**

(2) **accepta(cuvant)** - verifică dacă cuvântul dat ca parametru este acceptat de către automat și afișează: "accepta" - dacă este cuvânt acceptat, "neacceptat" - dacă nu este accepta , "blocaj" - dacă automatul se blochează pe parcurs. **5p**

(3) verifică() – verifică dacă automatul este ok (dacă starea inițială / stările finale se găsesc în mulțimea de stări, dacă tranzițiile conțin doar elemente ale automatului)

Construcția corectă a clasei **1p**

Cititrea elementelor automatului se face din fișier. Funcția de citire poate fi membră a clasei sau nu.

2. Citirea din fișier a elementelor AFD-ului - 1p

3. Posibilitatea de a verifica mai multe cuvinte, fără a reporni algoritmul (do-while)– **0.5p**

Un punct din oficiu

Un algoritm funcțional care doar citește și afișază elementele automatului - se puncteaza cu nota 3.