Nome: ____nº _______________

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

2014-2015 **TEORIA DA COMPUTAÇÃO** 5/Nov/2014 16h

Duração: 120m. 1ª Frequência

Leia atentamente:

1º- A prova é **sem** consulta.
2º- Responda na folha do enunciado.
3º- Não responda à sorte: respostas (de escolha) erradas têm pontuação negativa; respostas em branco têm pontuação nula.
4º- Para responder só pode utilizar os espaços do enunciado. Seja conciso e diga só o essencial. Quando a resposta for de escolha, assinale com X a que julgar certa.
5º- Coloque o nome e o nº de estudante em **todas** as folhas da prova.

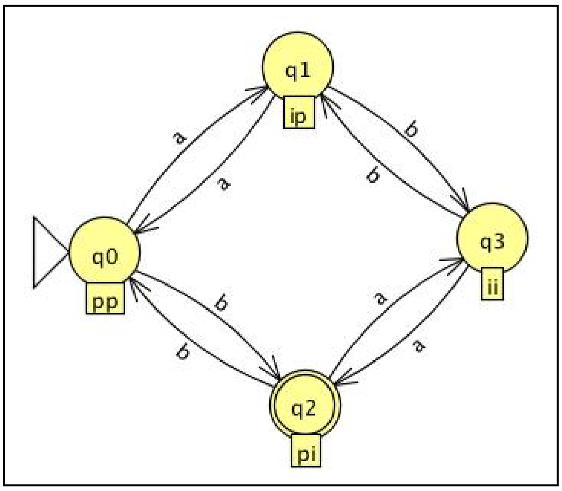
1. Considere as linguagens no alfabeto $\Sigma = \{a, b\}$:

$L_1 = \{(ab)^n(ba)^{2n} : n > 0\}$
$L_2 = \{w \in \{a, b\} : n_a(w) \geq n_b(w)\}$

Escreva uma gramática para a linguagem $L = L_1 \cup L_2$

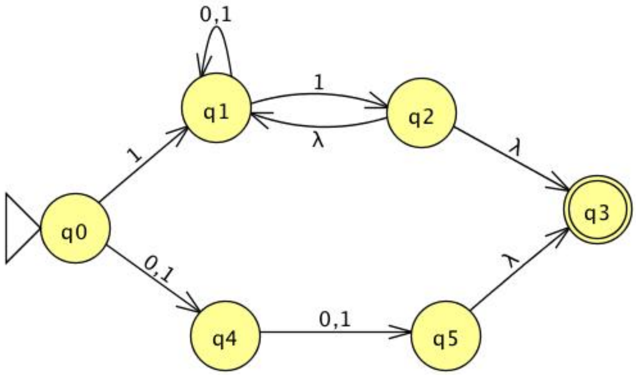
$S \rightarrow S_1 \mid S_2$
$S_1 \rightarrow ABB \mid AS_1BB$
$A \rightarrow ab$
$B \rightarrow ba$
$S_2 \rightarrow aS_2b \mid bS_2a \mid S_2S_2 \mid aS_2 \mid S_2a \mid \lambda$

2. Desenhe um autómato finito determinístico que aceite a linguagem no alfabeto $\Sigma = \{a, b\}$ composta por cadeias com um número par de ‘a’ (incluindo 0) e um número ímpar de ‘b’.



Nome: ____nº _______________

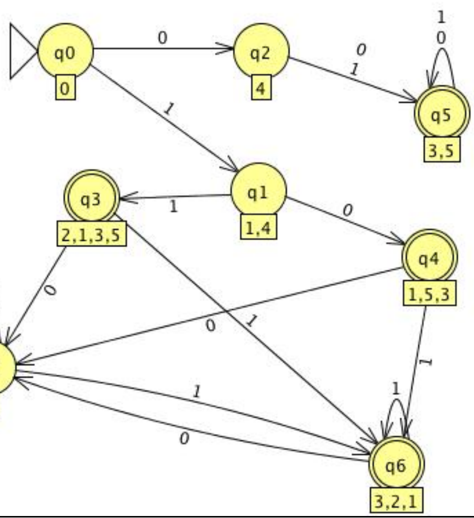
3. Seja o seguinte autómato finito não determinístico no alfabeto Σ ={0, 1}.



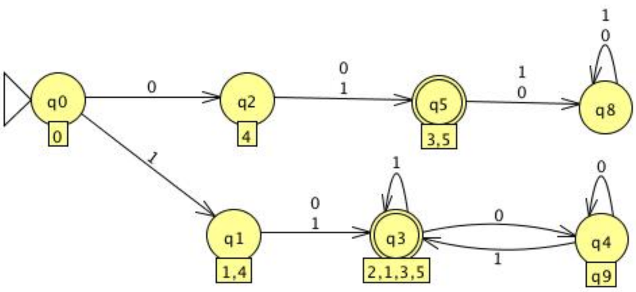
a) Converta-o num autómato finito determinístico com o menor número de estados possível, escrevendo a tabela de transições e desenhando o grafo do DFA.

Tabela	0	1
q0	q4	q14
q4	q35	q35
q14	q135	q1235
q35	∅	∅
q135	q1	q123
q1235	q1	q123
q123	q1	q123
q1	q1	q123
∅	∅	∅

Retirado da tabela



Com número mínimo de estados:

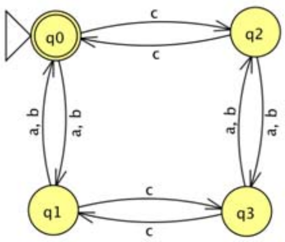


b) Descreva a linguagem por ele aceite

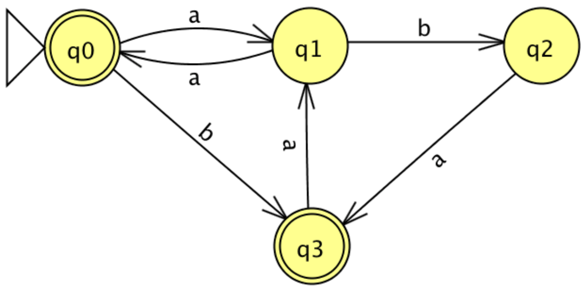
Todas as cadeias com apenas dois carateres ou iniciadas e terminadas pelo caracter ‘1’.

Nome: ____n° _______________

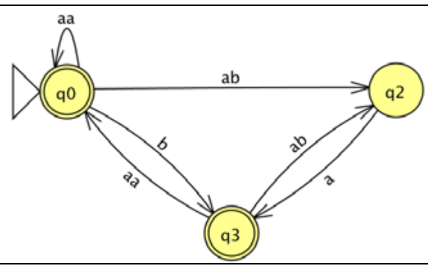
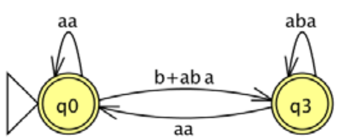
4. Construa um NFA com o menor número de estados possível no alfabeto Σ ={a, b, c} que aceite todas as cadeias de comprimento par (incluindo zero) e que contenham um número par de c's.



5. Construa a sua expressão regular do seguinte autómato, sendo Σ = {a, b}, apresentando todos os passos do algoritmo de eliminação de estados estudado nas aulas.



Eliminando q1:

q0q0 –∅ + a.∅.a **q0q2**-∅ + a.∅.b **q3q0**-∅+a. ∅.a **q3q2**-∅+a.∅.b	
Eliminado q2: **q0q3-** b+ab.∅.a **q3q3**-∅+ab.∅.a	

Finalmente:

ER = ((aa)+(b+aba)(aba)*aa)*.(λ+(b+aba)(aba)*)

Nome: ____nº _______________

6. Converta a seguinte Expressão Regular (a*b+c)+(a*a(a+b)*(a+b*)) numa Gramática Regular. Sugestão: desenhe previamente o NFA correspondente.

> S->c | A
> A->aA | B | b
> B->aB | aC
> C->aC | bC | D
> D->a | E
> E->bE| λ

7. Classifique as seguintes afirmações como verdadeiras ou falsas:

(i)	Qualquer gramática livre de contexto é linear.	V ☐ F ☒ X
(ii)	Uma linguagem gerada por qualquer gramática linear é regular.	V ☐ F ☒ X
(iii)	Uma gramática com duas produções lineares à esquerda e três lineares à direita pode gerar uma linguagem regular.	V ☐ F ☒ X

Justifique (iii):

> Para ser regular, todas as produções têm que ser lineares à direita, ou todas lineares à esquerda. Se forem lineares "mistas", não geram uma linguagem regular (por isso (ii) é falsa, mas esta justificação não era pedida).

8. Sendo L, M, N expressões regulares, diga se as seguintes igualdades são verdadeiras ou falsas:

a) (L + M) N= L + MN	Verdadeira: ☐ / Falsa: X☐
b) (NL + NM) = (L+ M)N	Verdadeira: ☐ / Falsa: x☐
c) (L*M*)* = (M + L)* + M*	Verdadeira: X☐ / Falsa: ☐

Justifique c):

> (M+L)*=(L+M)*=(M*L*)*=(M*L*)* que contém M*λ e portanto contém M*. Por isso (M+L)*+M* = (M+L)*.

9 Sendo A, B e C linguagens regulares, e $\underline{L}$ complemento de L, as linguagens seguintes são regulares:

(i) $\underline{A}$ Verdadeiro: x☐ / Falso: ☐

> Demonstre (i):
> Dado o DFA de A, invertem-se os papéis dos seus estados, passando os aceitadores a não-aceitadores e vice-versa. O autómato resultante aceita as cadeias do complemento de A, logo esta linguagem é regular porque existe um autómato finito que a aceita.

Nome: ____nº ______________

(ii) (A∩ (B∪C)   Verdadeiro: x☐ / Falso: ☐

Justifique (ii)

A família das CFL é fechada em relação à interseção, à união e à complementação, e portanto também em relação às operações lógicas em que entrem estes operadores lógicos.

10 Prove pelo lema da bombagem que a linguagem $L = \{ baba^p b^{3p} aba, p>0 \}$ não é regular.

Para qualquer m considere-se a cadeia

$$baba^m b^{3m} aba$$

Ela é de comprimento $4m+3+3$ e portanto maior do que m e pertence à linguagem. Nela poderemos contradizer o lema; se o conseguirmos provamos que a linguagem é nao-regular. **A identificação da cadeia na qual vamos trabalhar é importante na demonstração**. Note-se que ao escolhermos esta cadeia **assegura-se que os primeiros m caracteres serão necessáriamente *bab* seguido de m-3 ou menos *a*'s**. A decomposição $xy<=m$, como tem que estar no princípio da cadeia , é composta apenas por $bab+a$´s e portanto y será sempre composta ou por a's ou por a's precedidos por b, ab ou bab, - y pode ser a, ou a^2, ou a^3, ... ou a^{m-3} precedidas ou não por aqueles prefixos A bombagem só produz a's ou então a's e prefixos. Por exemplo se $y = a$, $x= baba^{m-4}$, $y^5 =aaaaa$ vai produzir uma cadeia com $m+4$ a's, mantendo-se os $3m$ b's. Portanto quebrando a regra das cadeias, que obriga a que nestas condições se tivesse $3(m+4)=3m+12$ b's em vez de $3m$. Bombeou-se assim para fora da linguagem. Qualquer bombagem de y, excepto y^1, resulta numa cadeia que não pertence a L (mas bastava até que só acontecesse para um caso). Se escolhêssemos qualquer outro y possível, por exemplo $y=a^2$, chegaríamos à mesma conclusão. Não existe a tal decomposição xyz. E se o y contém um prefixo (b, ab ou bab) além do desiquilíbrio do número de a's vai alterar a estrutura da cadeia.

Como tudo isto acontece para quaquer m, não existe nenhuma m que satisfaça o lema da bombagem, e por isso a linguagem não é regular.

Poderíamos escolher outras cadeias para a demonstração, com por exemplo, $baba^{m-3}b^{3(m-3)}aba$. A prova seria semelhante.

11. Dada uma gramática regular existe algum algoritmo para saber se a linguagem que gera é infinita ? Se sim, qual ?

Sim: constrói-se o NFA ou DFA a partir da gramática. Se houver no NFA ou no DFA pelo menos um ciclo num caminho aceitador, a linguagem é infinita.