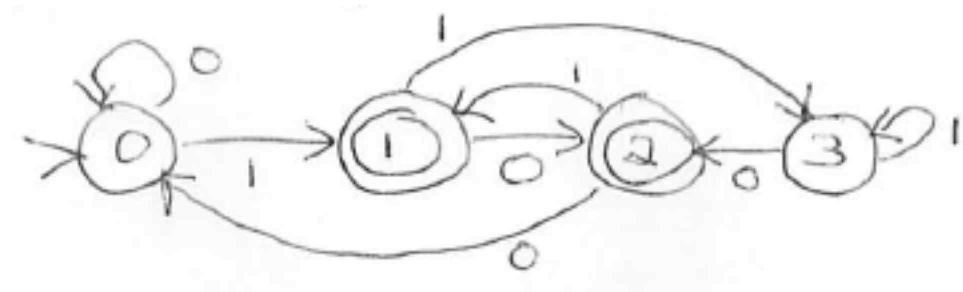
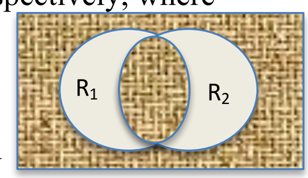
COT 4210 Fall 2017 Sample2 Midterm#1 Key Name:________________________

Total Points Available <u>75</u> Your Raw Score _______ Grade:____________

5 **1.** Present the transition diagram for a **DFA** that accepts the set of binary strings that represent numbers that have a remainder of either 1 or 2, when divided by 4. Numbers are read most to least significant digit, so 01 (1), 110 (6) and 10010 (18) are accepted, but 0000 (0), 111 (7) and 010011 (19) are not. Note: Leading zeroes are allowed.



8 **2.** Consider the following assertion:
Let $\mathbf{R_1}$ and $\mathbf{R_2}$ be regular languages that are recognized by $\mathbf{A_1}$ and $\mathbf{A_2}$, respectively, where $\mathbf{A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)}$ and $\mathbf{A_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)}$ are **DFA**s. Show that $\mathbf{L = \sim(R_1 \oplus R_2) = \{ w \mid w}$ is in the complement of $\mathbf{(R_1 \oplus R_2) \}}$
is also regular, where ~ means NOT and ⊕ means exclusive union.
Note: The figure on right shoes the set as the hatched part. From this you can see that the set can be described as $(\mathbf{R_1 \cap R_2}) \cup (\sim\mathbf{R_1} \cap \sim\mathbf{R_2})$

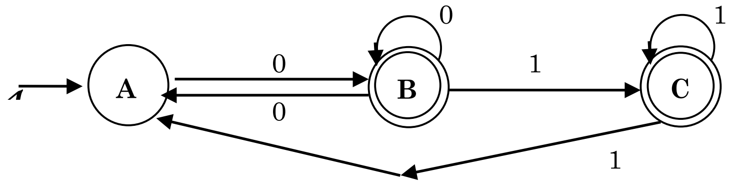


Present a **DFA** construction $\mathbf{A_3 = (Q_3, \Sigma, \delta_3, q_3, F_3)}$, where $\mathbf{L(A_3) = \sim(R_1 \oplus R_2)}$. You do not need to present a formal proof that it works, but you must clearly define $\mathbf{Q_3}$, $\mathbf{\delta_3}$, $\mathbf{q_3}$, and $\mathbf{F_3}$, and you must also justify its appropriateness by discussing how $\mathbf{\delta_3^*(q_3, w) \in F_3}$ **iff** $\mathbf{w \in \sim(R_1 \oplus R_2)}$.

$A_3 = (Q_3, \Sigma, \delta_3, q_3, F_3)$ where $Q_3 = Q_1 \times Q_2$ $q_3 = < q_1, q_2>$ and
$F_3 = F_1 \times F_2 \cup ((Q_1 - F_1) \times (Q_2 - F_2))$
$\delta_3(<q,r>, a) = < \delta_2(q, a), \delta_1(r, a)>$

Based on above, it is evident that $\delta_3^*(q_3, w) = < \delta_1^*(q_1, w), \delta_2^*(q_2, w)>$.
Hence $L(A_3) = \{ w \mid <\delta_2^*(q_1, w), \delta_1^*(q_2, w)> \in F_1 \times F_2 \cup ((Q_1 - F_1) \times (Q_2 - F_2))$
Restated, $w \in L(A_3)$ iff $\delta_1^*(q_1, w) \in F_1$ and $\delta_2^*(q_2, w) \in F_2$
or $\delta_1^*(q_1, w) \in Q_1 - F_1$ and $\delta_2^*(q_2, w) \in Q_2 - F_2$
But this says $w \in L(A_3)$ iff $w \in (R_1 \cap R_2)$ or $w \in (\sim R_1 \cap \sim R_2)$.
Thus, $L(A_3) = (R_1 \cap R_2) \cup (\sim R_1 \cap \sim R_2) = L = \sim(R_1 \oplus R_2)$,showing L is regular, as was desired.

3. Let **L** be defined as the language accepted by the following finite state automaton $\mathcal{A}$



8 **a.)** Present the regular equations associated with each of $\mathcal{A}$'s states, solving for the regular expression associated with the language recognized by $\mathcal{A}$. Hint: I would personally solve for **B** and **C** in terms of **A** first. Once you solve for these, then solve for **A** and substitute back in.

$A = \lambda + B0 + C1$

$B = A0 + B0 = A0^+$

$C = B1 + C1 = B1^+ = A0^+1^+$

$A = \lambda + A00^+ + A0^+1^+1 = (00^+ + 0^+1^+1)^*$

$L = B + C = A(0^+ + 0^+1^+) = A(0^+1^*) = (00^+ + 0^+1^+1)^*\ 0^+1^*$

5 **b.)** Assuming that we designate **A** as state **1**, **B** as state **2** and **C** as state **3**. Kleene's Theorem allows us to associate regular expressions $R_{i,j}^{k}$ with $\mathcal{A}$, where $i \in \{1..3\}$, $j \in \{1..3\}$, and $k \in \{0..3\}$.

The following are values of $R_{1,2}^{0} = 0$, $R_{2,1}^{0} = 0$

What are the values of $R_{2,2}^{0} = \lambda + 0$, $R_{1,1}^{0} = \lambda$?

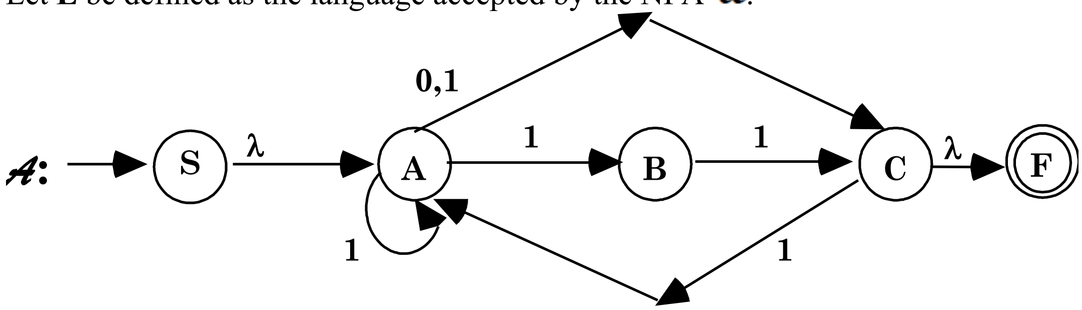
How is $R_{2,2}^{1}$ calculated from the set of $R_{i,j}^{0}$'s above? Give this abstractly in terms of the $R_{i,j}^{k}$'s

$$R_{2,2}^{1} = R_{2,2}^{0} + R_{2,1}^{0} R_{1,1}^{0\,*} R_{1,2}^{0}$$

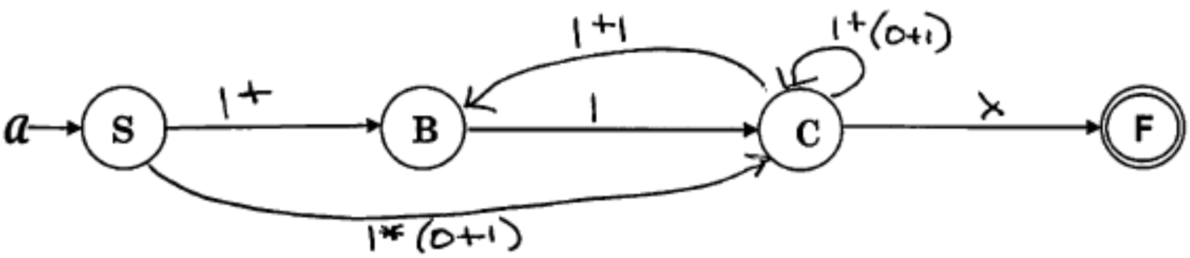
What expression does $R_{2,2}^{1}$ evaluate to, given that you have all the component values?

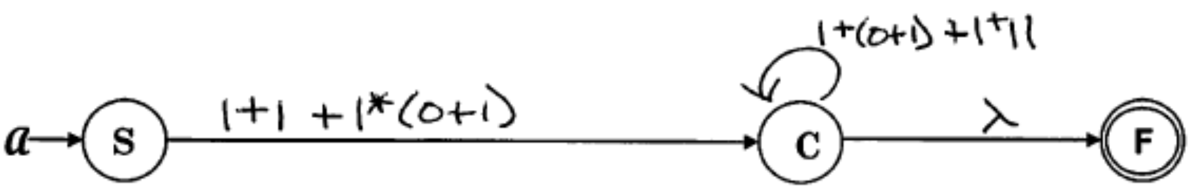
$$\lambda + 0 + 0\ \lambda^*\ 0 = \lambda + 0 + 00$$

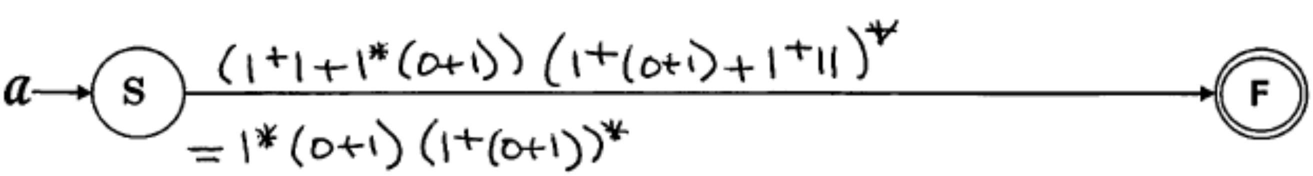
7 **4**. Let **L** be defined as the language accepted by the NFA $\mathcal{A}$:



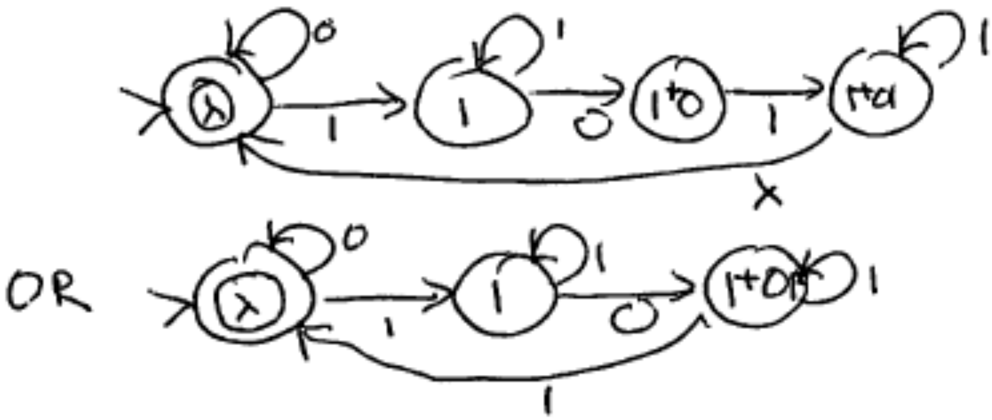
Using the technique of replacing transition letters by regular expressions and then ripping states from a GNFA to create new expressions, develop the regular expression associated with the automaton $\mathcal{A}$ that generates **L**. I have included the states of GNFA's associated with removing states **A**, **B** and then **C**, in that order. You must use this approach of collapsing one state at a time, showing the resulting transitions with non-empty regular expressions.







4 **5.** Consider the regular expression $\mathbf{R = (0 + 1^+ 01^+)^*}$
The first + is an "or" and the others (those superscripted) are related to Kleene * ($\mathbf{S^+ = S^* - \{\lambda\}}$)
Show a **NFA** (do by transition diagram) that accepts **R**.



6 **6.** Apply the Pumping Lemma to show the following is **NOT** regular. Be sure to differentiate the steps (contributions) to the process provided by the Pumping Lemma and those provided by you. Be sure to be clear about the contradiction. I'll even start the process for you.

$\mathbf{L = \{ a^i\, b^j\, c^k \mid k > i \text{ or } k > j \}}$

PL: Gives you a value of N>0.

You: $a^N\, b^{N+2}\, c^{N+1} \in L$ ***(Note: N+1 is greater than N but not greater than N+2)***

PL: $x\,y\,z = a^N\, b^{N+2}\, c^{N+1}$ ***where*** $|y|>0$, $|xy|\le N$ ***and*** $\forall i\ge 0$, $x\,y^i\,z \in L$

You: $i = 2$. ***Then*** $x\,y^2\,z = a^{N+|y|}\, b^{N+2}\, c^{N+1} \in L$ ***by PL. However,*** $N+|y| \ge N+1$ ***and*** $N+1<N+2$, ***so this string,*** $a^{N+|y|}\, b^{N+2}\, c^{N+1}$, ***is not in L, contradicting the PL. Thus, L cannot be Regular.***

4 **7.** Analyze the following language, **L**, proving it is **non**-regular by showing that there are an **infinite** number of equivalence classes formed by the relation $\mathbf{R_L}$ defined by:

$\mathbf{x\, R_L\, y}$ if and only if [$\forall \mathbf{z} \in \mathbf{\{a,b,c\}^*}$, $\mathbf{xz} \in \mathbf{L}$ exactly when $\mathbf{yz} \in \mathbf{L}$].

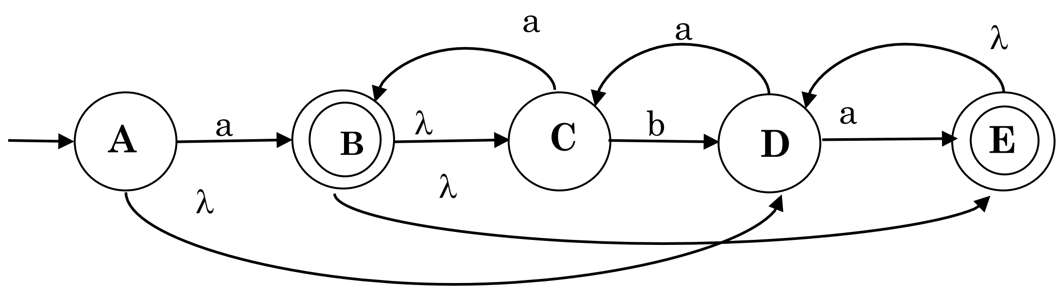
where $\mathbf{L = \{ c^k\, a^n\, b^m \mid n < m,\ k > 0 \}}$.

You don't have to present all equivalence classes, but you must demonstrate a pattern that gives rise to an infinite number of classes, along with evidence that these classes are distinct from one another.

Consider the set of equivalence classes $[ca^ib]$, $i \ge 0$.
Clearly $ca^ib \bullet b^i$ ***is in L, but*** $ca^jb \bullet b^i$ ***is not in L, when j>i.***
Thus, $[ca^ib] \ne [ca^jb]$ ***when j>i leading to an infinite number of equivalence classes induced by*** R_L
Consequently L is not Regular.

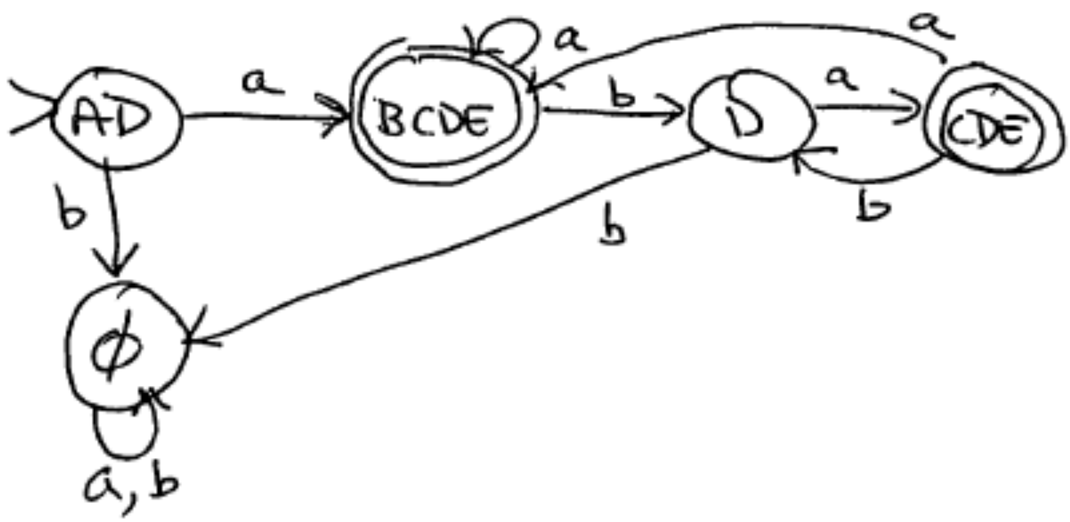
8. Let **L** be defined as the language accepted by the finite state automaton $\mathcal{A}$:

3 **a.)** Fill in the following table, showing the λ-closures for each of $\mathcal{A}$'s states.



State	A	B	C	D	E
λ-closure	**AD**	**BCDE**	**C**	**D**	**DE**

5 **b.)** Convert $\mathcal{A}$ to an equivalent deterministic finite state automaton. Use states like **AC** to denote the subset of states **{A,C}**. Be careful -- **λ-closures** are important.



3 **9.** Define **Prefix(L) = { x | ∃ y ∈ Σ*** where **xy ∈ L}, L^R = { w^R | w ∈ L }** and **Substring(L) = { y | ∃ x, z ∈ Σ*** where **xyz ∈ L}**

Assuming that Regular languages are already shown to be closed under **Prefix** and **Reversal** show that they are also closed under **Substring**. No proof is required, just a description of **Substring** that uses only the operations described above under which regular languages are closed.

Postfix(L) = (Prefix(L^R))^R, so SubstringL) = Prefix (Prefix(L^R))^R

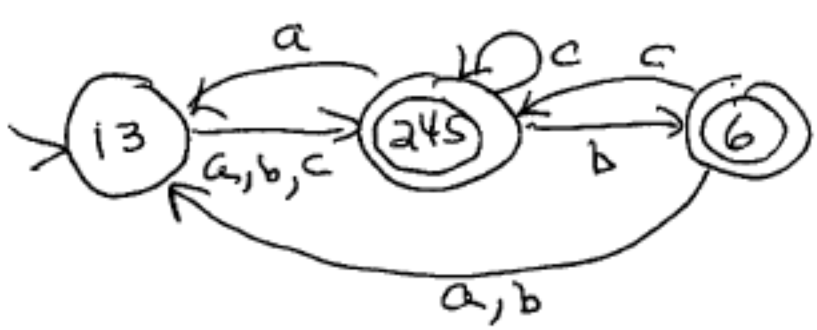
12 **10.** Given a DFA denoted by the transition table shown below, and assuming that **1** is the start state and **2** , **4** , **5** and **6** are final states, fill in the equivalent states matrix I have provided. Use this to create an equivalent, minimal state DFA.

	a	**b**	**c**
>1	**5**	**2**	**2**
2	**1**	**6**	**2**
3	**2**	**4**	**5**
4	**3**	**6**	**2**
5	**3**	**6**	**5**
6	**1**	**3**	**4**

2	**X**				
3	**2,5** **2,4**	**X**			
4	**X**	**1,2**	**X**		
5	**X**	**1.3**	**X**	**2,5**	
6	**X**	**3,6 X** **2,4**	**X**	**1,3** **3,6 X** **2,4**	**1,3** **3,6 X** **4,5**
	1	**2**	**3**	**4**	**5**

Don't forget to construct and write down your new, equivalent automaton!! Be sure to clearly mark your start state and your final state(s). In your minimum state DFA, label merged states with the states that comprise the merge. Thus, if **1** and **3** are indistinguishable, label the merged state as **13**.

{1,3} {2,4,5} {6} are right invariant equivalence classes



5 **11.** Present a Mealy Model finite state machine that reads an input **x** ∈ **{0, 1}*** and produces the binary number that represents the result of adding the twos complement representation of decimal **-6**, that is adding binary **1…1010** to **x** (this assumes all numbers are in two's complement notation, including results). Assume that **x** is read starting with its least significant digit.
Examples: **00010 → 11100; 11001 → 10011; 01011 → 00101**