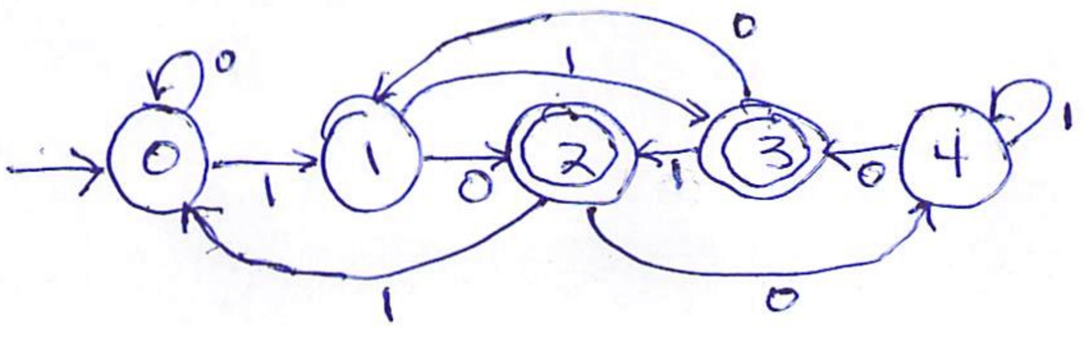
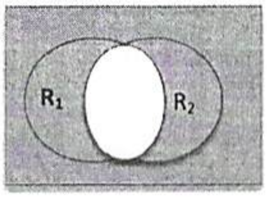
**COT 4210 Spring 2017** **Midterm#1** **Name:** KEY

**Total Points Available 75** **Your Raw Score** X **Grade:** $\frac{X+8}{75+3}$

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 2 or 3, when divided by 5. Numbers are read most to least significant digit, so 10 (2), 1101 (13) and 10001 (17) are accepted, but 0000 (0), 110 (6) and 010011 (19) are not. Note: Leading zeroes are allowed.



4 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 $\mathbf{L = \sim(R_1 \cap R_2) = \{ w \mid w}$ **is in the complement of** $\mathbf{(R_1 \cap R_2) \}}$ is also regular, where ~ means NOT.
Note: The figure on right shows the set as the non-white part.
Present a **DFA** construction $\mathbf{A_3 = (Q_3, \Sigma, \delta_3, q_3, F_3)}$, where $\mathbf{L(A_3) = \sim(R_1 \cap R_2)}$. You do not need to present a formal proof that it works, but you must clearly define $\mathbf{Q_3, \delta_3, q_3}$, and $\mathbf{F_3}$ .



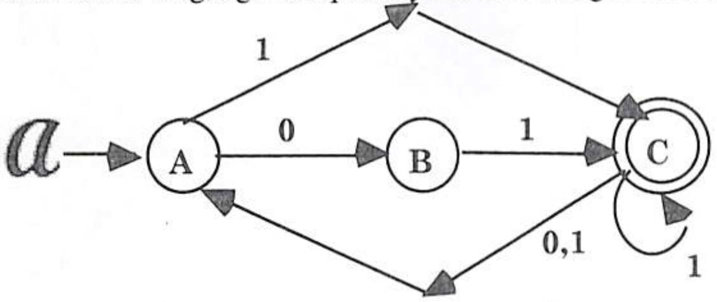
$$Q_3 = Q_1 \times Q_2$$

$$q_3 = \langle q_1, q_2 \rangle$$

$$\delta_3(\langle p, q\rangle, a) = \langle \delta_1(p,a), \delta_2(q,a)\rangle \quad \begin{matrix} a \in \Sigma \\ p \in Q_1 \\ q \in Q_2 \end{matrix}$$

$$F_3 = (Q_1 - F_1) \times Q_2 \cup Q_1 \times (Q_2 - F_2)$$
$$= Q_1 \times Q_2 - F_1 \times F_2$$

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 $\mathcal{a}$.

$A = \lambda + C(0+1)$
$B = A0 = 0 + C(0+1)0$
$C = A1 + B1 + C1$
$= 1 + C(0+1)1 + 01 + C(0+1)01 + C1$
$= (1+01) + C((0+1)1 + (0+1)01 + 1)$
$= (1+01)((0+1)1 + (0+1)01 + 1)^*$
$= (1+01)(01 + 11 + 001 + 101 + 1)^*$
$= (1+01)(01 + 001 + 1)^*$

4 **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 $R^k_{i,j}$ with $\mathcal{a}$, where $i \in \{1..3\}, j \in \{1..3\}$, and $k \in \{0..3\}$.

The following are values of $R^0_{1,1} = \lambda$, $R^0_{2,2} = \lambda$, $R^0_{3,3} = \lambda + 1$,

$R^0_{1,2} = 0$, $R^0_{1,3} = 1$, $R^0_{2,1} = \emptyset$, $R^0_{2,3} = 1$, $R^0_{3,1} = 0+1$, $R^0_{3,2} = \emptyset$

How is $R^1_{3,3}$ calculated from the set of $R^0_{i,j}$'s above? Give this abstractly in terms of the $R^k_{i,j}$'s

$R^1_{3,3} = R^0_{3,3} + R^0_{3,1}(R^0_{1,1})^* R^0_{1,3}$

What expression does $R^1_{3,3}$ evaluate to, given that you have all the component values?

$R^1_{3,3} = \lambda + 1 + (0+1)\lambda^* 1 = \lambda + 1 + 01 + 11$

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

$$\begin{tikzcd}
\mathcal{A} \arrow[r] & S \arrow[r, "\lambda"] & A \arrow[r, "0"] \arrow[rr, bend left, "1"] & B \arrow[r, "1"] & C \arrow[r, "\lambda"] \arrow[ll, bend left, "{0,1}"] \arrow[loop below, "1"] & \boxed{F}
\end{tikzcd}$$

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 a GNFA 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 Note: Your results here should be equivalent to but not necessarily the same as those in 3(a)

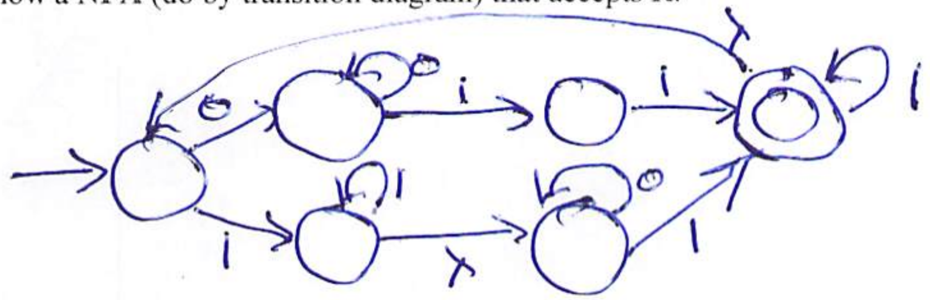
$$\begin{tikzcd}
\arrow[r] & S \arrow[r, "0"] \arrow[rr, bend left, "1"] & B \arrow[r, "1"] & C \arrow[r, "\lambda"] \arrow[l, bend left, "(0+1)0"] \arrow[loop above, "(0+1)1+1"] & \boxed{F}
\end{tikzcd}$$

$$\begin{tikzcd}
\arrow[r] & S \arrow[r, "1+01"] & C \arrow[r, "\lambda"] \arrow[loop above, "(0+1)1+1+(0+1)01"] & \boxed{F}
\end{tikzcd}$$

$$\begin{tikzcd}
\arrow[r] & S \arrow[r, "(1+01)((0+1)1+1+(0+1)01)^*"] & \boxed{F}
\end{tikzcd}$$

$$= (1+01)(01+1+001)^*$$

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



5 **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.

$\mathbf{L} = \{ \mathbf{a}^i \mathbf{b}^j \mathbf{c}^k \mid \mathbf{k = i+j, i,j > 0} \}$

**You: L is Regular**
**PL: Provides N>0 associated with L**
ME: $a^N b c^{N+1} \in L$
PL: $a^N b c^{N+1} = xyz$, $|xy| \le N$, $|y| > 0$ & $\forall i \ge 0$ $xy^iz \in L$
ME: $i = 0$: $a^{N-|y|} b c^{N+1} \notin L$ AS $N-|y|+1 < N+1$ SINCE $|y|>0$

THUS, L IS NOT REGULAR

4 **7.** Analyze the language, $\mathbf{L} = \{ \mathbf{a}^i \mathbf{b}^j \mathbf{c}^k \mid \mathbf{k = i+j, i,j > 0} \}$, 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:

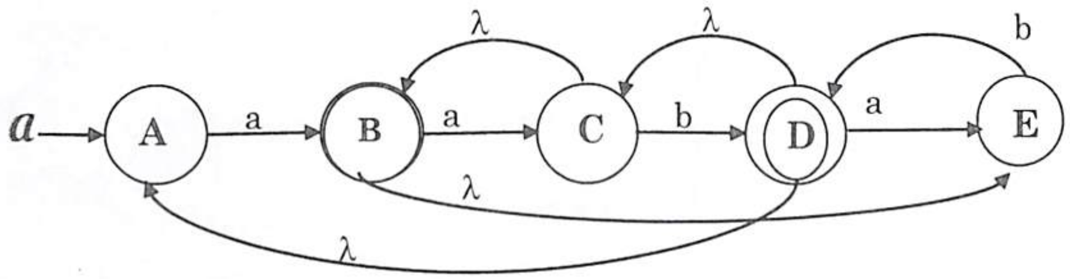
$\mathbf{x\, R_L\, y}$ if and only if $[\ \forall \mathbf{z} \in \{\mathbf{a,b,c}\}^*, \mathbf{xz} \in \mathbf{L}$ exactly when $\mathbf{yz} \in \mathbf{L}\ ]$.

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.

$[a^i b]_{R_L} a^{i+1} \in L$
$[a^j b]_{R_L} a^{i+1} \notin L$ WHEN $i \neq j$
SO $[a^i b]_{R_L} \neq [a^j b]_{R_L}$ $\forall i,j > 0$ $i \neq j$
THUS, $R_L$ HAS INFINITE INDEX AND SO
L IS NOT REGULAR

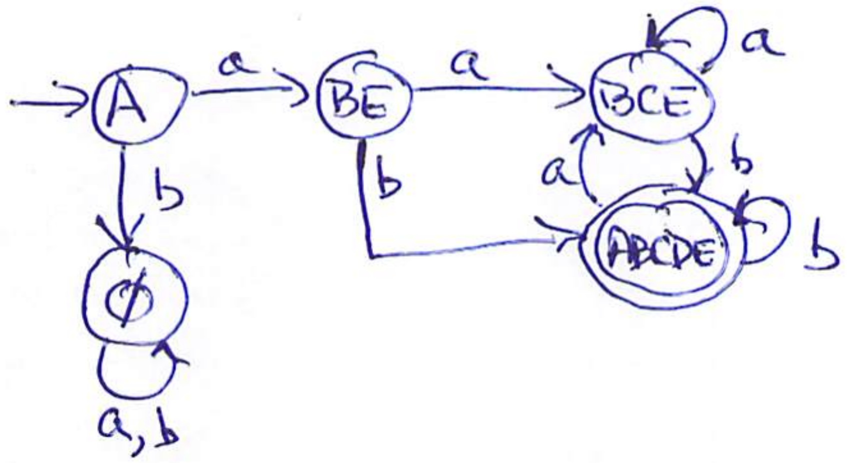
**8.** Let L be defined as the language accepted by the finite state automaton $\mathcal{A}$:

2 **a.)** Fill in the following table, showing the λ-closures for each of $\mathcal{A}$'s states. Note: **F={D}**.



| **State** | **A** | **B** | **C** | **D** | **E** |
|---|---|---|---|---|---|
| **λ-closure** | A | BE | BCE | ABCDE | E |

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.



4 **9.** Define **ProperMid(L)** = { **y** | ∃ **x,y,z** ∈ $\Sigma^+$ where **xyz** ∈ **L** }
Assuming that Regular languages are already shown to be closed under **Substitution**, **Homomorphism**, **Concatenation** and **Intersection with Regular Languages**, show they are closed under **ProperMid.** You should find it useful to employ the substitution **f(a) = {a, a'}**, and the homomorphisms **g(a) = a'** and **h(a) = a, h(a') = λ**. Here **a** ∈ **Σ** and **a'** is a new symbol associated with **a.** You do not need to show your construction works, but it must be based on the meta technique I showed in class for languages closed as above.

$$\text{PROPERMID}(L) = h\left(f(L) \cap g(\Sigma^+)\,\Sigma^+\,g(\Sigma^+)\right)$$

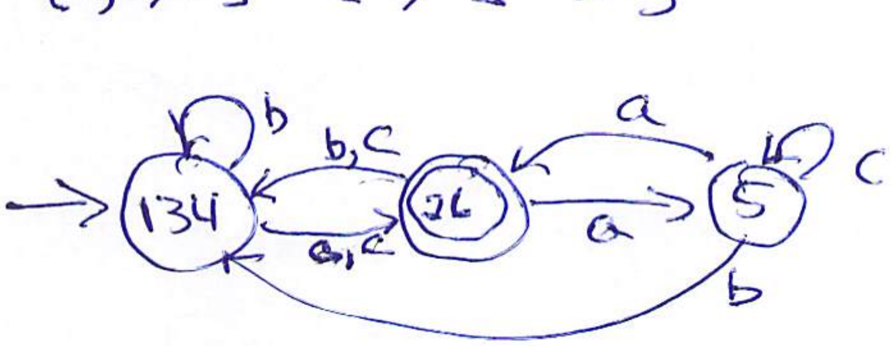
***10*** **10.** Given a DFA denoted by the transition table shown below, and assuming that **1** is the start state and 2 and 6 are final states, fill in the equivalent states matrix I have provided. Use this to create an equivalent, minimal state DFA.

| | a | b | c |
|---|---|---|---|
| >1 | 6 | 3 | 2 |
| 2 | 5 | 3 | 1 |
| 3 | 2 | 3 | 6 |
| 4 | 2 | 4 | 2 |
| 5 | 6 | 1 | 5 |
| 6 | 5 | 3 | 1 |

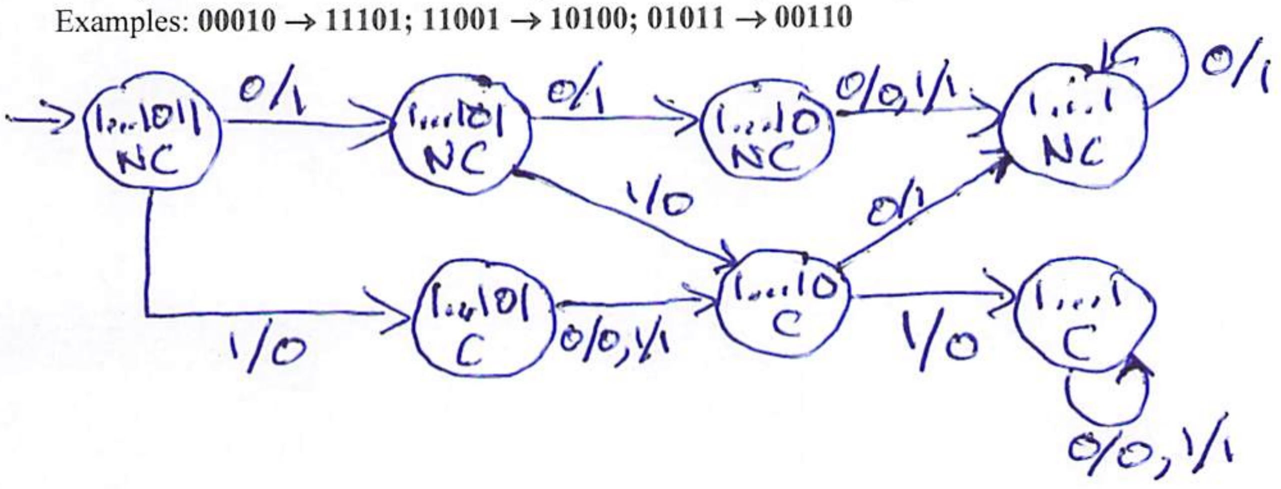
| | | | | | |
|---|---|---|---|---|---|
| 2 | X | | | | |
| 3 | 2,6 | X | | | |
| 4 | 2,6 3,4 | X | 2,6 | | |
| 5 | 1,3 2,5, X' | X | 2,6 1,3 5,6 X' | 2,6 1,4 2,5 X' | |
| 6 | X | | X | X | X |
| | 1 | 2 | 3 | 4 | 5 |

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, 2** and **3** are indistinguishable, label the merged state as **123**.

{1,3,4} {2,6}* {5}


$$\begin{tikzcd}
\arrow[r] & 134 \arrow[loop above, "b"] \arrow[r, bend right, "{a,c}"'] & 26 \arrow[l, bend right, "{b,c}"'] \arrow[r, "a"'] & 5 \arrow[l, bend right, "a"'] \arrow[loop right, "c"] \arrow[ll, bend left=60, "b"]
\end{tikzcd}$$

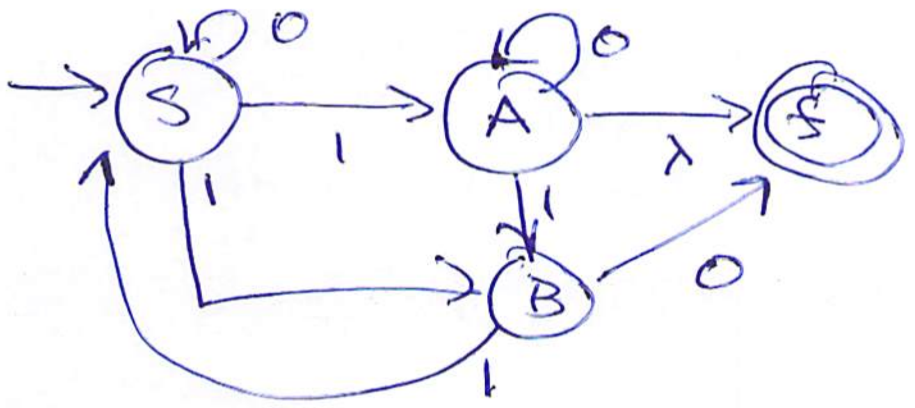
*7* **11.** Present a Mealy Model finite state machine that reads an input **x** ∈ **{0, 1}*** and produces the binary number that represents the result of adding the twos complement representation of decimal **-5**, that is adding binary **1…1011** to **x** (this assumes all numbers are in two's complement notation, including results). Assume that **x** is read starting with its least significant digit.
Examples: **00010 → 11101; 11001 → 10100; 01011 → 00110**



*5* **12.** Consider the regular grammar **G = ( {S, A, B}, {0, 1}, R, S )** where **R** is the set of rules:

| | | | | | | |
|---|---|---|---|---|---|---|
| **S** | → | **0 S** | \| | **1 A** | \| | **1 B** |
| **A** | → | **1 B** | \| | **0 A** | \| | **λ** |
| **B** | → | **1 S** | \| | **0** | | |

Present an NFA $\mathcal{A}_G$ that accepts the language generated by **G**: Be sure to indicate explicitly the states, alphabet, transition diagram, starting state and final state(s).



CALL ALSO MAKE A FINAL AND OMIT λ TRANSITION.