2010/11 exam

Model solutions (and some comments): Section A

1. **a.** The strings $\{\mathtt{a}, \mathtt{aa}, \mathtt{aaa}\}$ for example; alternatives are ϵ or $\mathtt{baa}$ etc.

   b. Strings (over a and b) consisting of a's (possibly empty), plus any other string with the feature that any b is followed by at least 2 a's.

   c. An equivalent FA is obtained by just removing the a transition from q_1 to q_2. A regular expression could be $\{\mathtt{a}, \mathtt{baa}\}^*$ or $\mathtt{a}^*(\mathtt{a}^*\mathtt{baaa}^*)^*$ or $\{\mathtt{a}^*, \mathtt{a}^*\mathtt{aaa}, \mathtt{a}^*\mathtt{baa}\}^*$

   Not quite correct: $(\mathtt{a}^*\mathtt{baaa}^*)^*$ or $\{\mathtt{a}^*\{\mathtt{a}, \mathtt{b}\}\mathtt{aaa}^*\}^*$ or or $(\mathtt{a}^*(\mathtt{baa})^*)^*$ or $\mathtt{a}^*(\mathtt{baa})^*\mathtt{a}^*$ or $\{\mathtt{a}, \mathtt{aaa}, \mathtt{baa}\}^*$ or $\mathtt{a}^*(\mathtt{baa})^*\mathtt{a}^*$ or $\mathtt{a}^* \cup \mathtt{baaa}^*$.

   d. This is basically bookwork but it was done quite poorly.

   Sets of states in the NFA correspond to individual states in the corresponding DFA; new state transitioned to from state q in the DFA is the union over all states of the sets of states labelled by that symbol in the NFA coming out of states in q. Accept if any state in the union accepts.

2. **a.** Given a regular language, there exists N such that any word w in the language of length $> N$ has a prefix w' of length N, where w' has a non-empty substring that can be "pumped", ie by replacing that substring with any sequence of copies of it, one obtains other words in the language. A substring that can be pumped corresponds to a loop in the FA. (strictly, an *accessible* loop, but wouldn't require that to be pointed out.)

   b. For any N, 0^N10^{N-1} is in the language; assume N is large enough to apply the PL with that value of N; then $0^{N+r}10^{N-1}$ belongs to the language, where 0^r can be pumped, and we have r is positive. But this can be seen to fail to comply with the description of strings in the language.

3. The question was not done well in the exam. Answers were commonly omitted or definitions of the relevant classes of languages were given, as opposed examples. Some confusion between "context-free language" and "context-free grammar"? "deterministic" means that the language has a deterministic PDA, it does not mean that you can write down a CFG where each variable has only one rule.

   a. e.g. palindromes over $\{a, b, c\}$ where there is a single occurrence of c in a word (by necessity, in the centre).

   b. e.g. $\{0^n1^n : n \geq 1\}$, Palindromes, etc. Full credit requires formal definition of language (but not proof).

   c. e.g. $\{a^nb^nc^n : n \geq 1\}$; $\{1^p : p \text{ is a prime number}\}$, etc.

   d. Expect to see Halting Problem for Turing Machines,

   $$\{\eta(M)w\ \eta(M) \text{ is encoding of TM, } M, \text{ and } M \text{ halts on } w\}$$

   but other (correct) answers acceptable.

4. This question was done well in the exam.

a.

	a	b	c	d
S	$S \longrightarrow ABCS$	$S \longrightarrow ABCS$	$S \longrightarrow ABCS$	$S \longrightarrow d$
A	$A \longrightarrow aA$	$A \longrightarrow \epsilon$	$A \longrightarrow \epsilon$	
B		$B \longrightarrow bB$	$B \longrightarrow \epsilon$	
C			$C \longrightarrow c$	

b. FIRST(S) is all four letters. A and B are nullable.

c. $S \Longrightarrow ABCS \Longrightarrow BCS \Longrightarrow bBCS \Longrightarrow bbCS \Longrightarrow bbcS \Longrightarrow bbcABCS \Longrightarrow bbcABCS \Longrightarrow bbcaABCS \Longrightarrow bbcaBCS \Longrightarrow bbcaCS \Longrightarrow bbcacS \Longrightarrow bbcacd$

Section B

Answer **two** questions in this section.

1. A common error was to claim that states 4 and 8 are inaccessible, on the grounds that because you can't accept after they have been reached, it is possible to remove them. They can be removed (given the convention that undefined transitions lead to rejection) but they are accessible.

   a. There are no inaccessible states; it is straightforward to check that we can reach any state starting at the initial one.

   Indistinguishable states: if we do this by working out what words are accepted starting from each state, we note that states 4 and 8 are all-rejecting, so they are indistinguishable.

   state 5 accepts the empty word and nothing else.

   state 7 accepts the empty word and nothing else, so same as state 5

   state 6 accepts strings of aś

   state 2 accepts $a \cup ba^*$.

   state 3 accepts $ba^* \cup a$, same as 2.

   state 1 accepts $\{a, b\}L_2$ where L_2 is the above language accepted by state 2.

   Generally, delete the inaccessible states and merge the indistinguishable states (combining transitions).

   A minimal DFA has state 1 with an a, b-transition to state 2; (state 3 is merged with state 2); state 2 has a b transition to state 6 and an a transition to state 5 (merged with 7); state 5 has a, b-transition to state 4 (merged with 8). States 4 and 5 are accepting.

   b. Suppose M has n states that are accessible and distinguishable.

   Suppose M' is equivalent to M but has only $n - 1$ states.

   We can find n words that reach each of the n states of M

   use accessibility here

   Two of these words must reach the same state of M'.

   We can find a suffix for these two words such that M would accept one but not the other

   here we use distinguishability

   But both words, with any suffix attached, must reach the same state in M'. So M' cannot accept the same set of words.

2. In part (a) of this question, a non-deterministic PDA was commonly given as an answer. The inattention to the requirement of being deterministic also affected part (b), where a common answer was "yes", and a NPDA was given as solution. Part b(ii) was done well in general.

   a. States A, A', I, F; stack alphabet x, y; I is initial state; F is accepting state; assume the criterion for accepting is that you must end in state F with an empty stack.

   $\delta(I, \epsilon, \epsilon) = (A, y)$ (mark the bottom of the stack)

$\delta(A, a, \epsilon) = (A, x)$ (add stack symbol for each a preceding b)
$\delta(A, b, \epsilon) = (A', \epsilon)$
$\delta(A', a, x) = (A', \epsilon)$ (remove stack symbol for each a following b)
$\delta(A', a, y) = (A', y)$ (allow excess stack contents to be discarded)
$\delta(A', b, y) = (F, \epsilon)$
A, perhaps nicer, approach is to allow the final state to pop stack symbols, so it can empty the stack.

b. (i.) Not accepted by a deterministic PDA. Consider words such as $a^n b a^n c a^m b a^m$. On reaching the central c, a DPDA has to have an empty stack since it should be able to accept at that point. Consequently it cannot remember the value n, and the entire word should be accepted provided that $m = n$, so in fact it is necessary to remember the number n.
(ii.) $S \longrightarrow aSA$; $S \longrightarrow bSB$; $S \longrightarrow cSC$;
$A \longrightarrow a$; $B \longrightarrow b$; $C \longrightarrow c$;
$S \longrightarrow a|b|c|\epsilon$

3. Not many attempts at giving an *unrestricted* grammar: CFGs were offered, which usually generated $a^*b^*c^*$. TMs were offered a couple of times, which is arguably harder than writing down an unrestricted grammar. Problem here was probably forgetting what is meant by "unrestricted grammar".

a. $S \longrightarrow XABCZ$
$X \longrightarrow F$
$FA \longrightarrow F$; $FB \longrightarrow F$; $FC \longrightarrow F$;
$Fa \longrightarrow aF$; $Fb \longrightarrow bF$; $Fc \longrightarrow cF$; $FZ \longrightarrow \epsilon$
$X \longrightarrow R$
$RA \longrightarrow AaR$; $Ra \longrightarrow aR$, and similarly for B, C
$RZ \longrightarrow LZ$
$\alpha L \longrightarrow L\alpha$ for all symbols α other than A
$AL \longrightarrow XA$

$S \Longrightarrow XABCZ \Longrightarrow RABCZ \Longrightarrow AaRBCZ \Longrightarrow AaBbRCZ \Longrightarrow AaBbCcRZ$
$\Longrightarrow AaBbCcLZ \cdots \Longrightarrow ALaBbCcZ \Longrightarrow XAaBbCcZ$
$\Longrightarrow RAaBbCcZ \Longrightarrow AaRaBbCcZ \Longrightarrow AaaRBbCcZ \Longrightarrow AaaBbRbCcZ$
$\Longrightarrow AaaBbbRCcZ \Longrightarrow AaaBbbCcRcZ \Longrightarrow AaaBbbCccRZ \Longrightarrow AaaBbbCccLZ$
$\ldots \Longrightarrow ALaaBbbCccZ \Longrightarrow XAaaBbbCccZ \Longrightarrow FAaaBbbCccZ$
(the above produces the F that initiates the final deletion of variables)

b. bookwork.