Name ______________________ Student No. ______________________

No aids allowed. Answer all questions on test paper. Use backs of sheets for scratch work.

Total Marks: 50

[10] 1. Let $B_n = \{a^k|$ where k is a multiple of n $\}$. Show that for each $n \geq 1$, the language B_n is regular.

Solution: Consider A with n states, $q_0, q_1, \ldots, q_{n-1}$, and $\delta(q_i, a) = q_{i+1}$, and q_0 is the only accepting state. The idea is that if you are in state q_i, then the number of a's seen so far, when divided by n, leaves reminder of i.

Another, simpler, solution is to note that $B_n = L((a^n)^*)$ (here a^n is the regular expression consisting of n concatenations of a).

[20] 2. Consider the language $F = \{w |\ w \text{ is } \textit{not} \text{ a palindrome } \}$. (Recall that a palindrome is a string that reads the same left-to-right as right-to-left, i.e., $w = w^R$.)

(a) Show that F is a Context-Free language.

(b) Show that F is not regular.

(c) Show that F satisfies the Pumping Lemma for regular languages, and explain why this does not contradict (b).

(Next page of the test is left blank if you require more space to answer this question.)

Solution: For (a), it may be easier to build a PDA than a CFG: a PDA P would push the first half of the string onto the stack, guess when it has seen the middle of the string, and then start popping the stack. If while popping it encounters a symbol in the input that differs from the corresponding symbol on top of the stack, then it would enter a state $q_{\text{almost-accept}}$ and pop to the end to make sure it guessed the middle correctly; if it did, it would accept.

For (b), recall that regular languages are closed under complementation, so it may be easier to show that the set of palindromes is not regular. To show that the set of palindromes is not regular, use (for example) the Myhill-Nerode Theorem: $[0^i]$ are different equivalence classes for each i: 0^i10^j is not a palindrome for $i \neq j$, but 0^j10^j is a palindrome; so $z = 10^j$ separates 0^i and 0^j.

Question (c) has a mistake, and therefore it won't be counted (so 10 points for (a) and 10 points for (b)). The Pumping Lemma does not hold for F.

[10] 3. Explain informally (and briefly) why a Push Down Automaton (PDA) with two stacks is more powerful than a PDA with a single stack. Give an example of a language that separates the two models of computation. Show that a PDA with two or more stacks is equivalent to a Turing machine.

Solution: A PDA with 2 stacks can recognize the following language $\{a^n b^n c^n | n \geq 0\}$ which is not CF (by pushing the a's onto the first stack, and the b's onto the second stack), and then popping both on the c's. With 2 stacks we can simulate a Turing machine by having the portion to the left of the state in the configuration of the TM on one stack, and the portion to the right on the other stack.

[10] 4. Given any Context Free language L, show that there is a decider M that on input w accepts iff $w \in L$. (In other words, you are asked to show that Context Free languages are Turing-decidable.)

Solution: Given a CFL, there exists a CFG recognizing it. Given any string w, let the decider M search all derivations with intermediate steps having at most $|w|$ many symbols. There are finitely many such derivations; if none yields w, reject, otherwise accept. (Note that this can be made tighter considering Chomsky Normal Forms.)

A second solution is to use a nondet TM, which guesses the next production, and rejects if the string on the tape is longer than the length of $|w|$.

Incidentally (some of you made this mistake), PDAs are by definition nondeterministic. Deterministic PDAs are provably weaker than nondeterministic ones. Furthermore, deterministic PDA that accept by final state are different from those accepting by empty stack (while the two classes coincide for nondeterministic ones). For example, $\{0\}^*$ is regular but deterministic PDAs that accept by empty stack cannot recognize it.