Name ______________________ Student No. ______________________

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

Total Marks: 50

[10] 1. (a) Convert the grammar:

$$S \longrightarrow 0S1|A$$
$$A \longrightarrow 1A0|S|\varepsilon$$

to a PDA that accepts the same language by empty stack.

(Problem 6.3.1)
Let $P = (\{q\}, \{0,1\}, \{0,1,A,S\}, \delta, q, S)$, where δ is given as follows:

(q,ε,S)	$\{(q,0S1),(q,A)\}$
(q,ε,A)	$\{(q,1A0),(q,S),(q,\varepsilon)\}$
$(q,0,0)$	$\{(q,\varepsilon)\}$
$(q,1,1)$	$\{(q,\varepsilon)\}$

[10] (b) Convert the PDA $P = (\{p, q\}, \{0, 1\}, \{X, Z_0\}, \delta, q, Z_0)$ to a CFG, if δ is given by:

$$\begin{aligned}
\delta(q, 1, Z_0) &= \{(q, XZ_0)\} \\
\delta(q, 1, X) &= \{(q, XX)\} \\
\delta(q, 0, X) &= \{(p, X)\} \\
\delta(q, \varepsilon, X) &= \{(q, \varepsilon)\} \\
\delta(p, 1, X) &= \{(p, \varepsilon)\} \\
\delta(p, 0, Z_0) &= \{(q, Z_0)\}
\end{aligned}$$

(Problem 6.3.3)

1. $S \to [qZ_0q] | [qZ_0p]$

The following four productions come from rule (1).

2. $[qZ_0q] \to 1[qXq][qZ_0q]$
3. $[qZ_0q] \to 1[qXp][pZ_0q]$
4. $[qZ_0p] \to 1[qXq][qZ_0p]$
5. $[qZ_0p] \to 1[qXp][pZ_0p]$

The following four productions come from rule (2).

6. $[qXq] \to 1[qXq][qXq]$
7. $[qXq] \to 1[qXp][pXq]$
8. $[qXp] \to 1[qXq][qXp]$
9. $[qXp] \to 1[qXp][pXp]$

The following two productions come from rule (3).

10. $[qXq] \to 0[pXq]$
11. $[qXp] \to 0[pXp]$

The following production comes from rule (4).

12. $[qXq] \to \varepsilon$

The following production comes from rule (5).

13. $[pXp] \to 1$

The following two productions come from rule (6).

14. $[pZ_0q] \to 0[qZ_0q]$
15. $[pZ_0p] \to 0[qZ_0p]$

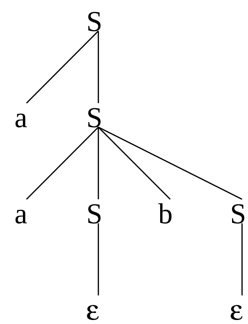
2. Consider the grammar G given by:

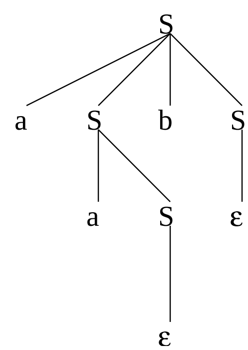
$$S \longrightarrow aS|aSbS|\varepsilon$$

[10] (a) Show that G is ambiguous.

(Problem 5.4.1)

To show that G is ambiguous, we show that the string aab has two different parse trees:





[10] (b) Prove that $L(G)$ contains the set $\{w \in \{a,b\}^* |$ every prefix of w has at least as many a's as b's$\}$.

Substantial Hint: Let $w = w_1 w_2 \dots w_n$, and for $1 \le i \le n$, let $(w)_{a,i}$ and $(w)_{b,i}$ denote the number of a's and b's in $w_1 w_2 \dots w_i$, respectively. You have to show that $L(G)$ contains the set:

$$L = \{w \in \{a,b\}^* | \forall i \in \{1,2,\dots,|w|\}, (w)_{a,i} \ge (w)_{b,i}\}$$

Use induction on the length of w, i.e., on $|w|$, to show that if $w \in L$, then w can be derived from S. In the Induction Step, consider two cases: (i) $w = aw_1$, where $w_1 \in L$, and (ii) $w = aw_1$ where $w_1 \notin L$. Why must $w \in L$, $|w| > 0$, always start with a?

(Problem 5.4.2)

We show by induction on $|w|$, that every string in L can be derived from S. In the Basis Case we consider $|w| = 0$, so $w = \varepsilon \in L$ (in L since ε has no initial segments, and hence it is vacuously in L), and $S \to \varepsilon$, so we are done.

For the induction step assume that the claim holds for $|w| = n$, i.e., if w is a string of length n in L, then $S \overset{*}{\Rightarrow} w$. Consider $w \in L$, such that $|w| = n+1$. First of all, $w = aw_1$, since $(w)_{a,i} \ge (w)_{b,i}$ holds true for all $1 \le i \le n+1$, and hence in particular for $i = 1$. This suggests two cases to consider:

case (1) $w_1 \in L$ as well, so $S \Rightarrow aS \overset{*}{\Rightarrow} aw_1$, and we are done.

case (2) $w_1 \notin L$. In that case there must exists a smallest i_0 (Least Number Principle) such that $(w_1)_{a,i_0} < (w_1)_{b,i_0}$. Thus the situation looks as follows:

a	w_2	b	w_3

where $w_1 = w_2 b w_3$ and b is in position i_0 of w_1 (and hence in position $i_0 + 1$ in w), and we know that it is a b because i_0 has been chosen as the smallest number such that $(w_1)_{a,i_0} < (w_1)_{b,i_0}$, so we know that $(w_1)_{a,i_0-1} = (w_1)_{b,i_0-1}$ (note that this includes the case $i_0 = 0$, i.e., $w_2 = \varepsilon$), and hence there *must* be a b in position i_0 in w_1.

Furthermore, by our choice of i_0, it follows that $w_2 \in L$. Now we show that $w_3 \in L$ as well. Since $(w_1)_{a,i_0-1} = (w_1)_{b,i_0-1}$, it follows that w_2 has as many a's as b's, and hence aw_2b also has as many a's as b's. Therefore, since $w \in L$, it follows that every initial segment of w_3 must obey the property that $(w_3)_{a,i} \ge (w_3)_{b,i}$ for $1 \le i \le |w_3|$, and hence $w_3 \in L$.

So now, $S \Rightarrow aSbS \overset{*}{\Rightarrow} aw_2bS \overset{*}{\Rightarrow} aw_2bw_3 = w$, since by induction hypothesis w_2, w_3 are derivable from S.

[10] 3. Show that for every non-deterministic Turing Machine N, there exists a deterministic Turing Machine M, such that $L(N) = L(M)$.

Hint: The general idea is more important than the details of the simulation. Don't spend too much time on this question, rather give a high level answer.

(Theorem 8.11)

Suppose that we are given a nondet TM N. We show how to simulate it with a det TM M.

M has two tapes, and it works as follows:

On the first tape it stores IDs of N, so the tape contains $\mathrm{ID}_1, \mathrm{ID}_2, \mathrm{ID}_3, \ldots$. One of these IDs is going to be marked as the *current* ID. If ID_i is the current ID, then M will generate all the IDs that can be obtained from it by a single move of N, and it is going to write them at the right end of its list of IDs. When M starts on input x, it contains the initial configuration of N on its tape, i.e., ID_1.

For example, if the tape contains $\mathrm{ID}_1, \mathrm{ID}_2, \mathrm{ID}_3$, and ID_2 is marked as the current ID, and from ID_2 N can produce 2 new IDs in a single move, then M would append $\mathrm{ID}_4, \mathrm{ID}_5$ to the end of the tape.

Once it generates all the IDs that can be obtained from ID_i by a single move of N, M marks the next ID on the list as the current ID, and it repeats. M stops when it encounters an ID with an accepting state. On the second tape, M does its scratch work, such as when it generates new IDs, it stores the current ID on the scratch tape, and then copies it to the end of the first tape, so that M can then perform the necessary modifications to those IDs.

Suppose N accepts a string x. Then, there exists a sequence of, say, n moves, such that the n-th ID of N is an accepting one. Suppose that m bounds the number of possible moves of N on any given state/symbol pair. Then, M will encounter this ID within at most $1 + m + m^2 + \cdots + m^n \leq nm^n$ stages, and hence it will also accept. On the other hand, if N does not accept x, then no ID will be an accepting one, so M will also not accept. Thus, $L(M) = L(N)$.