Mid II **Theory of Computation** **Total: 30 Marks**

1. Define $L_t = \{(M, w, t)$: M is a Turing machine that accepts input w in less than t steps $\}$. Is L_t Turing decidable for each positive integer $t > 0$? Is $L = L_1 \cup L_2 \cup .....$ Turing decidable? Justify your answer. $\boxed{3}$

   Soln: L_t is decidiable for each t because on input (M, w, t), a universal Turing machine (UTM) can be used to simulate M on w for t steps. If M accepts w within t steps, the UTM can accept the input (M, w, t) and reject otherwise. However, $\bigcup_t L_t = L_u$ and hence undecidable. (In particular, although the union of a finite collection of decidable languages is decidable, the union of an infinite collection of decidable languages need not be decidable. In fact, if countable unions of decidable languages were decidable, every language would be decidable! (why?))

2. Show that $L = \{M : M$ does not accept any input $\}$ is undecidable assuming that $L_u = \{(M, w) : M$ accepts $w\}$ is undecidable. $\boxed{3}$

   Soln: Since L_u is undecidable, its complement $\overline{L}_u$ is undecidable as well. We show that $\overline{L}_u \leq_m L$. For this, Consider a reduction algorithm A that takes as input a machine input pair (M, w) and outputs a machine M', whose operation is defined as follows. M' on input x first simulates M on w. (The simulation may loop for ever if M doesn't halt on w.) M' accepts x if M accepts w. If M rejects w, then M' rejects x. We also assume that if M is not a valid Turing machine, M' rejects x. With this operation, $L(M') = \Sigma^*$ if $(M, w) \notin \overline{L}_u$ and $L(M') = \phi$ if $(M, w) \in \overline{L_u}$.

3. Show that the language $L = \{a^n b^m : n > m\}$ has infinite number of Myhill Nerode equivalence classes. What can you conclude about the language from this? $\boxed{3}$

   Soln: Let i, j be positive integers with $i < j$. Consider $x_i = a^i$ and $x_j = a^j$. $x_i b^i \notin L$ whereas $x_j b^i \in L$. Hence x_i and x_j are inequivalent with respect to the Myhill Nerode relation. Since for all positive integers i, j the argument holds, it follows that $\equiv_L$ has infinitely many classes, and by Myhill Nerode theorem, we can conlude that L is not regular.

4. Give an example for a regular language L such that any DFA accepting L requires at least two final states. (Hint: You need to use Myhill Nerode theorem). $\boxed{3}$

   Soln: Consider the language $a(a+b)^* + b$. The strings $x = a$ and $y = b$ are both in the language, but are not in the same equivalence class with respect to the Myhill Nerode relation $\equiv_L$. This is because $xa \in L$ whereas $ya \notin L$. Hence, any DFA $M = (Q, \Sigma, \delta, q_0, F)$ accepting L must satisfy $\delta^*(q_0, x) \neq \delta^*(q_0, y)$, and both $\delta^*(q_0, x)$ and $\delta^*(q_0, y)$ must be final states. (There are several languages other than $a(a+b)^* + b$ on which is an argument similar to the above can be carried out, and any correct argument will be accepted).

5. Design a minimum DFA for accepting all binary strings with number of $1s$ being a multiple of 3. Prove that your DFA has minimum number of states. $\boxed{3}$

   Soln: Let L be the language under consideration. It is not hard to design a DFA with 3 states. The optimality of the design follows from the fact that $\equiv_L$ has exactly three classes. An alternate method is to show that the state minimization algorithm, when run on your minimum state DFA does not identify any equivalent states and hence your DFA is minimum. (In this argument, you are using the fact that the DFA is optimal if and only if the algorithm fails to identify any equivalent states.)