C1 **Theory** (25 points)

a. (6.25 points)
Let $L = \{x \in \{a, b\} \mid x\text{'s final five symbols include two } a\text{'s and three } b\text{'s}\}$. *Explicitly prove by Myhill-Nerode* that L *is* regular.

b. (6.25 points)
Let $L = \{w \cdot w^R \mid w \in \{a, b\}^*\}$, *where* w^R is w spelled backwards, and '$\cdot$' is the string concatenation operation — not an alphabet symbol. *Explicitly prove by Generalized Pumping for Finite Automata* that L is *not* regular.

c. (6.25 points)
Construct a *four* state finite automaton $\mathcal{M}$ for accepting $L = \{ab\}$ *and explicitly prove by Myhill-Nerode* your $\mathcal{M}$ *is* minimal state.

d. (6.25 points)
Explicitly use Generalized Pumping for Finite Automata to show that *no* finite automaton accepting $L = \{ab\}$ has fewer than *three* states.

C2 Theory (25 points)

a. (12.5 points)
Explicitly draw a state diagram for a *PDA* which accepts all and only the words in

$$L = \{xayubv \mid x, y, u, v \in \{a, b\}^* \ \wedge\ |x| = |y| \ \wedge\ |u| = |v|\}. \tag{1}$$

b. (12.5 points)
Let $L = \{a^m b^n a^m b^n \mid m, n > 1\}$. *Explicitly employ Pumping for PDA* to show that L is *not* a Context Free Language, i.e., that it is *not* accepted by any PDA.

C3 Theory (25 points)

Let $N \stackrel{\text{def}}{=}$ the set of all non-negative integers.

Definition Consider all the finite sets of equations defining primitive recursive functions and which contain a special one argument function letter $\mathbf{f}$ and whose remaining function letters are chosen only from the list $\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, \ldots$. The Base Functions s, n, and the u_i^n's $(1 \leq i \leq n)$ are also allowed. *Suppose* available to you is a Gödel numbering (code numbering) of all these finite sets of equations 1-1 onto N.

1. Let E_q be (by definition) the finite set of equations with Gödel number q.
2. $f_q \stackrel{\text{def}}{=}$ the primitive recursive function which $\mathbf{f}$ defines in E_q.[1]
3. The E_qs are the *programs* of a functional programming language for all and only the class of one-argument primitive recursive functions.

Clearly, then, $f_0, f_1, f_2, \ldots$ is a list of all and only the primitive recursive functions of one argument.

Let

$$h(x) \stackrel{\text{def}}{=} 1 + \max_{p \leq x} f_p(x). \tag{2}$$

a. (12.50 points)

Prove h is computable. You should present an algorithm for h *in*formally.

b. (12.50 points)

Suppose g is any primitive recursive function of one argument. *Prove* that

$$(\overset{\infty}{\forall} x)[h(x) > g(x)].^2 \tag{3}$$

[1] For *example*, consider equations by which $\mathbf{f}_1$ defines binary addition as primitive recursive (in the usual way), $\mathbf{f}_2$ defines binary multiplication as primitive recursive (using $\mathbf{f}_1$ in the usual way), and $\mathbf{f}$ defines factorial as primitive recursive (using $\mathbf{f}_2$ in the usual way). Let E be all and only the resultant finite set of equations. Let q be E's code number. Then $E_q = E$, and f_q is the factorial function.

[2] The quantifier '$(\overset{\infty}{\forall} x)$' means 'for all except finitely many non-negative integers x'.

C4 Theory (25 points)

Fix a standard programming formalism φ for computing all the one-argument partial computable functions which map the non-negative integers into themselves. *Suppose* you are provided with a coding (Gödel numbering) of the φ-programs *onto* the entire set of non-negative integers. Let φ_p denote the partial function computed by program (number) p in the φ-system. Let Φ denote a standard Blum step-counting measure associated with φ.[3] You may assume any standard characterizations of r.e. sets, that K ($\stackrel{\text{def}}{=} \{x \mid \varphi_x(x) \text{ is defined}\}$) *is* r.e., that $\overline{K}$ is *not* r.e., and that, in the φ-system, universality, S-m-n, and the Kleene Recursion and Rice Theorems hold.[4]

For the sets described in C4 part a and C4 part b below, *prove or disprove each* of the two bulleted items just below.

- *The set* is r.e.
- The set's *complement* is r.e.

Be sure to *explicitly say*, in each case, whether you are proving or disproving.

a. (12.5 points)
$A_1 = \{x \mid [\varphi_x(0) \textit{ is } \text{defined}] \wedge [\varphi_x(1) \text{ is } \textit{un}\text{defined}]\}$.

b. (12.5 points)
$A_2 = \{x \mid (\forall u)(\exists v > u)[\varphi_x(v) \text{ is } \textit{un}\text{defined}]\}$.

[3]Hence, (i) $(\forall p)[\text{domain}(\Phi_p) = \text{domain}(\varphi_p)]$, *and* (ii) $[\{(p, x, t) \mid \Phi_p(x) \leq t\}$ is an algorithmically decidable set$]$.

[4]You won't need all these assumptions.

Some of these assumptions refer to multiple argument partial computable functions, so you may assume these are formulated employing some standard computable pairing function iterated as needed to code multiple arguments as one.