C1 **Theory** (25 points)

a. (12.5 points)
Let

$$L_1 = \{w \in \{a,b\}^* \mid \text{100th from last symbol of } w \text{ exists } \textit{and} \text{ is an } a\}. \quad (1)$$

Show that L_1 *is* regular *by explicitly exhibiting AND proving correct* a set (as per the Myhill-Nerode Theorem) which spans mod L_1.

b. (12.5 points)
Employ an appropriate pumping lemma to show that

$$L_2 = \{a^m b^n \mid m,n \in N \text{ and } \textit{either } m \text{ is even } \textit{or } n \text{ is a perfect square}\} \quad (2)$$

is *not* regular.

C2 **Theory** (25 points)

a. (6.25 points)

For this problem (C2a) you are to *prove by a product construction* that context free languages (CFLs) are closed under intersection with any regular language. **No credit if you do it some other way.** Following are the details as to what you are to do.

Let pushdown automaton (pda) $\mathcal{M}_1 = (A, \Gamma_1, Q_1, \delta_1, q_1^1, Z_1, F_1)$, where A is $\mathcal{M}_1$'s finite, non-empty input tape alphabet, Q_1 is $\mathcal{M}_1$'s finite set of states, δ_1 is $\mathcal{M}_1$'s transition function (see (3) below), Γ_1 is $\mathcal{M}_1$'s finite, non-empty stack alphabet, q_1^1 is $\mathcal{M}_1$'s start state, Z_1 is $\mathcal{M}_1$'s initial stack symbol, and F_1 is $\mathcal{M}_1$'s set of final states. Suppose $\mathcal{M}_1$ accepts by final state (and jams on empty stack). To remind you:

$$\delta_1 : Q_1 \times (A \cup \{\varepsilon\}) \times \Gamma_1 \rightarrow \text{ the finite subsets of } (Q_1 \times \Gamma_1^*), \tag{3}$$

where ε is the empty string. Let (deterministic) finite automaton $\mathcal{M}_2 = (A, Q_2, \delta_2, F_2, q_1^2)$, where A is $\mathcal{M}_2$'s finite, non-empty input tape alphabet (and the same as $\mathcal{M}_1$'s), Q_2 is $\mathcal{M}_2$'s finite set of states, δ_2 is $\mathcal{M}_2$'s transition function (see (4) below), q_1^2 is $\mathcal{M}_2$'s start state, and F_2 is $\mathcal{M}_2$'s set of final states. To remind you:

$$\delta_2 : Q_2 \times A \rightarrow Q_2. \tag{4}$$

You are to define a pda $\mathcal{M}_3 = (A, \Gamma_3, Q_3, \delta_3, q_1^3, Z_3, F_3)$ such that $L(\mathcal{M}_3) = (L(\mathcal{M}_1) \cap L(\mathcal{M}_2))$. Be sure to say what *you* mean by each of $\Gamma_3, Q_3, \delta_3, q_1^3, Z_3, F_3$. Of course since *your* $\mathcal{M}_3$ is to be based on a product construction, you *must* have $Q_3 = (Q_1 \times Q_2)$. Be sure to include an *explicit, algebraic* definition (by cases) of *your*

$$\delta_3 : Q_3 \times (A \cup \{\varepsilon\}) \times \Gamma_3 \rightarrow \text{ the finite subsets of } (Q_3 \times \Gamma_3^*), \tag{5}$$

in terms of δ_1, δ_2. Be sure to handle thereby the fact that while $\mathcal{M}_1, \mathcal{M}_3$ *can* make ε-moves $\mathcal{M}_2$ can*not*.

b. (6.25 points)

Let

$$L_3 = \{a^m b^n a^m b^n \mid m, n \geq 0\}. \tag{6}$$

You may use *without* proof that L_3 is *not* a CFL.

Let

$$L_4 = \{ww \mid w \in \{a, b\}^*\}. \tag{7}$$

Prove using the result of C2a that L_4 is *not* a CFL.

Hint: Suppose for contradiction L_4 is a CFL.
How much is $(L_4 \cap L(a^*b^*a^*b^*))$? Why is this relevant?

c. (6.25 points)

Draw a state diagram for a *pda* which accepts all and only the words in

$$L_5 = \{xayubv \mid x, y, u, v \in \{a, b\}^* \ \wedge \ |x| = |y| \ \wedge \ |u| = |v|\}. \tag{8}$$

d. (6.25 points)

Let

$$L_6 = \{ab^p \mid p \text{ is prime}\}. \tag{9}$$

Explicitly use *Pumping for PDA* to show that L_6 is *not* accepted by any *pda*.

C3 **Theory** (25 points)

This problem will take you through a *sequence* of simple subproblems to prove the following

Theorem There is an infinite computable set A with no infinite primitive recursive subset.

You may use any subproblems to prove *subsequent* subproblems!

Consider all the finite sets of equations defining primitive recursive functions (with non-negative integer inputs and outputs) and which contain a special *one* argument function letter $\mathbf{f}$ and whose remaining function letters are chosen only from the list $\mathbf{f_1}, \mathbf{f_2}, \mathbf{f_3}, \ldots$. Imagine doing a Gödel numbering (code numbering) of all these finite sets of equations 1-1 onto $\{0, 1, 2, \ldots\}$. Do *not* indicate how to do this.

1. Let E_q be (by definition) the finite set of equations with Gödel number q.
2. $f_q \stackrel{\text{def}}{=}$ the one argument 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 (non-negative integer) argument.

Let $A_q \stackrel{\text{def}}{=} f_q^{-1}(1)$.[2]

Note: the subscript of A_q codes how to decide membership in A_q — using the program E_q. Also the E_q's can be compiled into any standard programming system for the partially computable functions (but do *not* prove this).

Below p, x range over non-negative integers.

We indicate next how to define a set of non-negative integers A by presenting below an informal algorithmic procedure for C_A.[3] Hence, A will be computable (but you should not prove this further). This procedure's recursion is only for handling a *cancellation* mechanism — essentially a priority queue mechanism — but expressed more simply.

You will need to refer to this procedure defining A in your proofs of many of the subproblems of C3.

begin *procedure for defining C_A (on input x)*
  recursively run this procedure to find the *least* p, if any, such that
    (i) p is not *cancelled* by any inputs $< x$,
    (ii) $2p < x$,
    and
    (iii) $x \in A_p$;
  if p is found
    then
      set $C_A(x) = 0$ (* We say that p *contributes* x to $\overline{A}$ *);
      cancel p (* We say that x *cancels* p *)
    else
      set $C_A(x) = 1$
  endif
end *procedure for defining C_A*

(Problem C3 continues onto the next page.)

[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 *this* E's code number. Then $E_q = E$ and f_q is the factorial function.

[2] $f_q^{-1}(1) \stackrel{\text{def}}{=} \{x \mid f_q(x) = 1\}$.

[3] $C_A(x) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } x \in A; \\ 0, & \text{if } x \notin A. \end{cases}$

(This page is the continuation of Problem C3.)

a. (3.00 points)
Prove that $A_0, A_1, A_2, \ldots$ is a list of all and only the primitive recursive *sets* of non-negative integers. You may use any standard text book results about primitive recursive functions *provided you clearly say which results you are using when.*[4]

b. (3.00 points)
Prove that any $x \leq 2p$ contributed to $\overline{A}$ must be contributed by some non-negative integer $p' < p$.

c. (3.00 points)
Prove that *at most* p x's that are $\leq 2p$ get contributed to $\overline{A}$.

d. (3.00 points)
Prove that the number of x's that are $\leq 2p$ is $2p + 1$.

e. (3.00 points)
Prove that the number of x's that are $\leq 2p$ *and in* A is *at least* $p + 1$.

f. (3.00 points)
Prove that A is infinite.

g. (7.00 points)
Suppose $A_p \subseteq A$. To complete the proof of the theorem, we want to show that *then* A_p is finite. To do this, it clearly suffices to show the stronger conclusion that $\max(A_p) \leq 2p$.
Under the just above supposition that

$$A_p \subseteq A, \tag{10}$$

prove that $\max(A_p) \leq 2p$.

Hint for C3g: Suppose for contradiction, for some x,

$$2p < x \in A_p. \tag{11}$$

Argue, from (11) and (10) above, about whether p can be cancelled by any non-negative integer x'. Also argue from your conclusion on that and from (11) and (10) about whether x is contributed to $\overline{A}$. Show, then, that x *must* cancel some non-negative integer $p' < p$. But, then, argue that x gets contributed to $\overline{A}$ by this p'. Get a contradiction.

[4] The result of C3a is *not* such a standard text book result. (⌣̈)

C4 **Theory** (25 points)

Let N denote the set of non-negative integers.

Fix a standard programming formalism φ for computing all the *one-argument* partial computable functions which map N into N.[5] Code (Gödel) number the φ-programs *onto* N. Let φ_p denote the partial function computed by program (number) p in the φ-system.

Let $\Phi_p(x) \stackrel{\text{def}}{=}$ the number of steps φ-program p executes on input x if p on x halts and undefined if p on x does not halt. You may assume: $\Phi_p(x)$ defines a *partially* computable function of p, x; $\Phi_p(x)$ is defined exactly when $\varphi_p(x)$ is defined; and

$$\{(p, x, t) \mid \Phi_p(x) \leq t\} \text{ is a computable set.} \tag{12}$$

We write $\Phi_p(x) > t$ to mean *either* $\Phi_p(x)$ is defined to be some number $> t$ *or* $\Phi_p(x)$ is undefined.

You may assume with*out* proof that, in the φ-system, Universality, S-m-n, and the Kleene Recursion Theorem (KRT) hold.

Prove by explicit application of KRT the following

Theorem *Suppose f is computable. Then there is an e such that*

1. $\varphi_e = f$ *and*
2. $(\forall x \in N)[\Phi_e(x) < \Phi_e(x+1)]$.

Hint for C4: Suppose f is computable. Apply KRT to get an e *which creates a self-copy* and which, on $x > 0$, uses that self-copy to (try to) compare $\Phi_e(x-1)$ and $\Phi_e(x)$. If, as is *not* desired, $\Phi_e(x-1) \geq \Phi_e(x)$, make sure e does something that, in the case that $\Phi_e(x-1)\downarrow$, i.e., in the case that $\Phi_e(x-1)$ is defined, will yield a contradiction. Otherwise, have e output $f(x)$. That was about $x > 0$. Explicitly make $\varphi_e(0) = f(0)$ — with *no* use of e's self-copy.

When you have the behavior of your e on any input x all worked out *and have justified that your e's use of its self-copy and of its input x is algorithmic*, then argue as follows.

Suppose for contradiction that x is the *least* number such that $\varphi_e(x)\uparrow$, i.e., such that $\varphi_e(x)$ is undefined. Argue that, then, $\Phi_e(x)\uparrow$. Argue that $\Phi_e(0)\downarrow$. Show, then, that $[x > 0 \wedge \Phi_e(x-1) \geq \Phi_e(x)]$. What can you then conclude re $\Phi_e(x-1)$? What can you then conclude re $\varphi_e(x-1)$? Get a contradiction. Argue that, then, φ_e is total. Finish the proof of the theorem.

[5] N.B. We are using a *lower* case Greek phi for this notation. Below we employ the upper case of this Greek letter but for a different purpose.