C1 **Theory** (25 points)

a. (5 points)
Suppose L is regular. Prove that

$$L_1 = (L \cap L^R) \tag{1}$$

is regular.[1]

b. (10 points)
Use a Pumping Lemma to prove that

$$L_2 = \{w \in \{a,b\}^* \mid w = w \text{ spelled backwards}\} \tag{2}$$

is *not* regular.[2]

c. (10 points)
Use a Pumping Lemma to show that

$$L_3 = \{a^{\lceil n^2 \rceil} \mid n \in N\} \tag{3}$$

is *not* regular.
Hint: $(\forall m, n \in N \mid n > m)[n^2 - m^2 \geq 2m + 1]$.

---

[1] L^R is (by definition) the set of all words from L *spelled backwards*. For example, $\{a^m b^n \mid m, n > 0\}^R = \{b^n a^m \mid m, n > 0\}$.
[2] For example, $abbba \in L_2$, but $abbb \notin L_2$.

C2 **Theory** (25 points)

a. (5 points)
Suppose that $\mathcal{M}$ is a (deterministic) finite automaton with exactly n states. *Suppose* that

$$(\exists x \in L(\mathcal{M}))[n \leq |x|].^3 \tag{4}$$

Prove that $L(\mathcal{M})$ is infinite.

b. (10 points)
Suppose that $\mathcal{M}$ is a (deterministic) finite automaton with exactly n states. *Suppose* that $L(\mathcal{M})$ is infinite.

Prove that

$$(\exists x \in L(\mathcal{M}))[n \leq |x| < 2n]. \tag{5}$$

c. (10 points)
In finite automata theory, a 0-*move* is (by definition) a machine state change accompanied by *no* forward move of the read head. In some text books 0-moves are called ε-*moves*.

N.B. For this question you may use *without stating how they work or proving them correct* standard compilers to:

1. translate any 0-move non-deterministic finite automaton into a non-deterministic finite automaton with no 0-moves which accepts the same language *and*
2. translate any non-deterministic finite automaton with no 0-moves into a (deterministic) finite automaton which accepts the same language.

Devise an algorithm for deciding of an arbitrary (deterministic) finite automaton $\mathcal{M}$ over a fixed alphabet A, *whether or not* $L(\mathcal{M})$ has an *infinite number of words of odd length*.

N.B. Should your algorithm employ any subalgorithms (including standard ones from text books), other than the compilers mentioned above in this problem, *you must clearly state how those subalgorithms work*.[4]

You may find useful the results of Parts a and b of this question.

You need *not* verify correctness of your algorithms, but *do be both clear and correct*.

[3] $|x|$ denotes the length of the string x.

[4] For *example*, if you were to employ a subalgorithm for translating any (deterministic) finite automaton into a 0-move non-deterministic finite automaton which accepts the Kleene-star of the language the finite automata accepts, you would need to say very clearly how this subalgorithm works.

C3 Theory (25 points)

We consider below functions which have non-negative integer arguments and values. We use Church's lambda-notation to name functions defined by formulas (as is standard in Lisp). For example, $\lambda x.[x+1]$ denotes the function that maps each non-negative integer x to $x+1$, and $\lambda x,y.[x+y]$ denotes the function of two non-negative integer arguments mapping them to their sum.

Let N denote the set of non-negative integers.

For each $x, y \in N$, let

$$x \dot{-} y \stackrel{\text{def}}{=} \begin{cases} x-y, & \text{if } x \geq y; \\ 0, & \text{otherwise;} \end{cases} \tag{6}$$

$$p_x \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } x = 0; \\ \text{the } x\text{-th prime number}, & \text{otherwise;} \end{cases} \tag{7}$$

and let

$$\langle x, y \rangle \stackrel{\text{def}}{=} 2^x(2y+1) \dot{-} 1. \tag{8}$$

You may assume *without* proof that $\lambda x,y.[\langle x,y\rangle]$ maps the set of all pairs of non-negative integers 1-1 onto the set of all non-negative integers. Let ℓ and r be the left and right inverses of $\lambda x,y.[\langle x,y\rangle]$, respectively. Then, we have that, for all x, y, z,

$$\ell(\langle x,y\rangle) = x, r(\langle x,y\rangle) = y, \text{ and } \langle \ell(z), r(z)\rangle = z. \tag{9}$$

You may also assume *without* proof that the functions in the set $\mathcal{S}$ are each primitive recursive, where $\mathcal{S} \stackrel{\text{def}}{=}$

$$\{\lambda x,y.[x+y], \lambda x,y.[x\times y], \lambda x,y.[x \dot{-} y], \lambda x,y.[\langle x,y\rangle], \ell, r, \lambda x.[p_x], \lambda x.[x>0]\}.^5 \tag{10}$$

For each $n, x_1, \ldots, x_n \in N$, let

$$[x_1, \ldots, x_n] \stackrel{\text{def}}{=} \Pi_{i=1}^n p_i^{x_i}.^6 \tag{11}$$

You may also assume *without* proof that, for each positive $n \in N$, $\lambda x_1,\ldots,x_n.[x_1,\ldots,x_n]$ is primitive recursive. You may further assume *without* proof the existence of primitive recursive functions $\lambda x.[\mathrm{Lt}(x)]$ and $\lambda x,i.[(x)_i]$ such that

- $\mathrm{Lt}(0) = \mathrm{Lt}(1) = 0$, and, for each non-negative integer $x > 1$, for each $n \in N$, if $\mathrm{Lt}(x) = n$, then p_n divides x, but *no* prime greater than p_n divides x; and
- for each $x, i \in N$, $(x)_0 = 0$, $(0)_i = 0$, and, if $x, i > 0$, $(x)_i =$ the exponent of p_i in the unique prime power factorization of x.

Definition Suppose $n \in N$. Suppose f has $n+1$ arguments.

f is defined from h by course of values recursion $\stackrel{\text{def}}{\Leftrightarrow}$, for each $x_1, \ldots, x_n, y$,

$$f(x_1,\ldots,x_n,y) = h(y, [f(x_1,\ldots,x_n,0), f(x_1,\ldots,x_n,1),\ldots,f(x_1,\ldots,x_n,y-1)], x_1,\ldots,x_n). \tag{12}$$

Finally, you may also assume *without* proof that the class of primitive recursive functions is closed under definition by course of values recursion.

Definition *f is defined from g_1, g_2, and h by intertwined primitive recursion* $\stackrel{\text{def}}{\Leftrightarrow}$

$$(\forall y \in N)[f(0,y) = g_1(y)]; \tag{13}$$

$$(\forall x \in N)[f(x+1,0) = g_2(x)]; \tag{14}$$

and

$$(\forall x,y \in N)[f(x+1,y+1) = h(x,y,f(x,y+1),f(x+1,y))]. \tag{15}$$

Prove from any or all of the assumptions without proof above *and* the definition of primitive recursive function that, *if* g_1, g_2, and h are primitive recursive and f is defined from g_1, g_2, and h by intertwined primitive recursion, *then* f is also primitive recursive. You need *not* put in all the Base Functions (e.g., projection functions), but give enough detail to show you know what you are doing and don't say false things.

[5] It is to be understood that predicates (such as $\lambda x.[x>0]$) have value 1 when they are true and 0 when they are false.
[6] Of course, as is standard, an empty product evaluates to 1.

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. Code (Gödel) number 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.[7] Let $W_p \stackrel{\text{def}}{=}$ the domain of φ_p.[8] You may assume *without* proof that in the φ-system Universality, S-m-n, and the Kleene Recursion Theorem (KRT) hold.

The first two parts of this question will lead you (with *very* useful hints) through a proof of the following

Theorem *Suppose* Δ is a collection of r.e. sets. *Let*

$$P_\Delta \stackrel{\text{def}}{=} \{p \mid W_p \in \Delta\}. \tag{16}$$

Suppose P_Δ is r.e.
Then

$$(\forall p)[W_p \in \Delta \Leftrightarrow (\exists \text{ a finite set } D \subseteq W_p)[D \in \Delta]]. \tag{17}$$

The third and fourth parts of this question each asks you to apply the theorem and also provides very useful hints. In that interest and for later use, let

$$A = \{p \mid W_p = \{0\}\}. \tag{18}$$

a. (6.25 points)
Assume all the hypotheses of the Theorem. *Explicitly use KRT* in the φ-system (formally or informally — as you choose) to prove that

$$(\forall p)[W_p \in \Delta \Rightarrow (\exists \text{ a finite set } D \subseteq W_p)[D \in \Delta]]. \tag{19}$$

Hint: Suppose that $W_p \in \Delta$. Suppose for contradiction that $(\forall \text{ finite sets } D \subseteq W_p)[D \notin \Delta]$. Apply KRT to obtain an self-referential e which determines its I/O behavior on input x *in part* according to whether or not "e appears in P_Δ within x steps." Make this precise, figure out what to have e do in each case, etc., and get a contradiction.

b. (6.25 points)
Assume all the hypotheses of the Theorem. *Explicitly use KRT* in the φ-system (formally or informally — as you choose) to prove that

$$(\forall p)[(\exists \text{ a finite set } D \subseteq W_p)[D \in \Delta] \Rightarrow W_p \in \Delta]. \tag{20}$$

Hint: Suppose $(\exists \text{ a finite set } D \subseteq W_p)[D \in \Delta]$. Let D be an example. Suppose for contradiction that $W_p \notin \Delta$. Apply KRT to obtain an self-referential e which determines its I/O behavior on input x *in part* according to whether it eventually discovers that "$[x \in D \vee e$ appears in $P_\Delta]$." Make this precise, figure out what to have e do if it makes this discovery, *etc.*

c. (6.25 points)
Explicitly use the Theorem stated above in this question to show that A is *not* r.e., *where* A is defined in (18) above.

Hint: Suppose for contradiction otherwise. Clearly $A = P_\Delta$ for $\Delta = \{\{0\}\}$. Therefore, from (17) above, we have that $(\forall p)[W_p = \{0\} \Leftrightarrow (\exists \text{ a finite set } D \subseteq W_p)[D = \{0\}]]$. Pick D and W_p so that $D = \{0\} \subseteq W_p \neq \{0\}$. Get a contradiction.

d. (6.25 points)
Explicitly use the Theorem stated above in this question to show that $\overline{A}$ is *not* r.e., *where* A is defined in (18) above.

Hint: Suppose for contradiction otherwise. Clearly $\overline{A} = P_\Delta$ for $\Delta = \{W_p \mid W_p \neq \{0\}\}$. Therefore, from (17) above, we have that $(\forall p)[W_p \neq \{0\} \Leftrightarrow (\exists \text{ a finite set } D \subseteq W_p)[D \neq \{0\}]]$. Pick a D and W_p to get a contradiction.

---

[7] 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$]$.

[8] Then $W_0, W_1, W_2, \ldots$ provides a standard listing of *all* the r.e. sets (of non-negative integers).