1. For each of the following languages over the alphabet $\Sigma = \{0, 1\}$, either prove that the language is regular (by constructing a DFA or regular expression) or prove that the language is not regular (using fooling sets). Recall that Σ^+ denotes the set of all nonempty strings over Σ.

   (a) $L_{2a} = \{0^n 1^n w \mid w \in \Sigma^* \text{ and } n \geq 0\}$

   Solution: The language is regular.When $n = 0$ the whole string is just represented by w.In this case all the strings over $\{0, 1\}$ can be just represented by w.So,we can ignore the $0^n 1^n$ portion as every string is covered by w.So every string in L_{2a} can be represented by w .

   The regular expression for L_{2a} will be $(0 + 1)^*$.

   Hence,the language L_{2a} is a regular language. ■

   (b) $L_{2b} = \{w0^n w| \ w \in \Sigma^* \text{and } n > 0\}$

   Solution: Let us consider the fooling set $F = \{1^n 0^n | n > 0\}$

   Let x and y be arbitrary strings in F.

   Then $x = 1^i 0^i$ and $y = 1^j 0^j$ for some positive integers $i \neq j$.

   Let $z = 1^i$.

   Then $xz = 1^i 0^i 1^i \in L_{2b}$.

   And $yz = 1^j 0^j 1^i \notin L_{2b}$, because $i \neq j$.

   Thus, F is a fooling set for L_{2b}.

   Because F is infinite, L_{2b} cannot be regular. ■

   (c) $L_{2c} = \{xwwy | w, x, y \in \Sigma^+\}$

   Solution: L_{2c} is a regular language. The language only contains a limited number of strings that are not part of it. By the fact that any language of finite size is regular and regularity is preserved under complement. We can prove L_{2c} is regular.

   We can say that any string of length at least 4 is in the language. For an arbitrary string z of length at least 4, Let us define z as $xz'y$ where both a and b are a single symbol in Σ. So, z' has a of length at least 2.

   x and y are two non-empty strings which are covered by $((0 + 1)^+)$. z' will be ww. Since, our alphabet consists of just 0s and 1s, there are two cases. First, z' must be 00 or 11. This satisfies the constraint that there must be a repeating string in the middle with length at least one. Now, let us consider the case where we can have alternating 0's and 1's i.e, $w=$ 10 or 01. To include this case we add 1010 and 0101 to the regular expression.

   Putting it all together you get the regular expression for L_{2c}: $(0 + 1)^+(00 + 11 + 1010 + 0101)(0 + 1)^+$.

   Hence, the language L_{2c} is a regular language. ■

(d) $L_{2d} = \{xwwx | w, x \in \Sigma^+\}$

Solution: Let us consider the fooling set $F = \{1^n 0^n | n > 0\}$

Let x and y be arbitrary strings in F.

Then $x = 1^i 0^i$ and $y = 1^j 0^j$ for some positive integers $i \neq j$.

Let $z = 0^i 1^i$.

Then $xz = 1^i 0^i 0^i 1^i \in L_{2d}$.

And $yz = 1^j 0^j 0^i 1^i \notin L_{2d}$, because $i \neq j$.

Thus, F is a fooling set for L_{2d}.

Because F is infinite, L_{2d} cannot be regular. ∎

2. Describe the context-free grammar that describes each of the following languages:

(a) All strings in $\{0,1\}^*$ whose length is divisible by 5.

Solution: We want 5 steps to "count" which index we are on. The only symbol that can end the recurrence is the ε in S. Then we allow any number to transition to the next step and wrap it back to S after 5 inputs.

$$\begin{aligned} S &\rightarrow 1A \mid 0A \mid \varepsilon \\ A &\rightarrow 1B \mid 0B \\ B &\rightarrow 1C \mid 0C \\ C &\rightarrow 1D \mid 0D \\ D &\rightarrow 1S \mid 0S \end{aligned}$$

■

(b) $L_{3b} = \{0^i 1^j 2^{i+j} | i, j \geq 0\}$

Solution: We want to add a 2 for every 0 and 1 added. To do this we divide it into 2 steps. First we add an equal number of 0s and 2s on each side. Then we add an equal number of 1s and 2s on the inside.

$$\begin{aligned} S &\rightarrow 0S2 \mid A \\ A &\rightarrow 1A2 \mid \varepsilon \end{aligned}$$

■

(c) $L_{3c} = \{0^i 1^j 2^k | i = j \text{ or } j = k\}$

Solution: There are 2 cases, one where i=j and one where j=k. When i=j we add equal 0s and 1s then an arbitrary number of 2s. When j=k we add equal 1s and 2s then an arbitrary number of 0s.

$$\begin{aligned} S &\rightarrow AB \mid XY \\ A &\rightarrow 0A1 \mid \varepsilon \\ B &\rightarrow 2B \mid \varepsilon \\ X &\rightarrow 0X \mid \varepsilon \\ Y &\rightarrow 1Y2 \mid \varepsilon \end{aligned}$$

∎

(d) $L_{3d} = \{w \in \{0,1\}^* | \#(01,w) = \#(10,w)\}$ (function $\#(x,w)$ returns the number of occurrences of a substring x in a string w)

Solution: There are 2 cases, one where the string starts with 0 and one where it starts with 1. When it starts with 0 we can add as many 0s as we want but when a 1 is added we eventually need to add another 0 to balance the substrings. The second case is the same with 0 and 1 switched.

$$
\begin{aligned}
S &\rightarrow 0A \mid 1X \mid \varepsilon \\
A &\rightarrow 0A \mid 1B \mid \varepsilon \\
B &\rightarrow 1B \mid 0A \\
X &\rightarrow 1X \mid 0Y \mid \varepsilon \\
Y &\rightarrow 0Y \mid 1X
\end{aligned}
$$

∎

Solution (clever): A clever observation is that the criteria for having the same number of substrings of 01 and 10 is satisfied if the string starts and ends with the same symbol. This means we can break the string into 2 cases where we guarantee that if the string starts with a 0 it ends with a 0 and same with 1s. (We include the zero and one length cases at the start as they are not included in the recursive step)

$$
\begin{aligned}
S &\rightarrow 0A \mid 1X \mid 0 \mid 1 \mid \varepsilon \\
A &\rightarrow 0A \mid 1A \mid 0 \\
X &\rightarrow 0X \mid 1X \mid 1
\end{aligned}
$$

∎

3. An all-NFA M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ that accepts $x \in \Sigma^*$ if **every** possible state that M could be in after reading input x is a state from F. Note, this is in contrast to an ordinary NFA that accepts a string if some state among these possible states is a an accept state. Prove that all-NFAs recognize the class of regular languages.

Solution: To solve this problem we need to look at it from two directions. The first is to show that, all-NFAs accept all regular languages. To show that, we can take any DFA D which accepts the regular language L. We know that D accepts every string in L and that there is exactly one path that it follows to the accepting state. So any DFA can be considered as an all-NFA, and hence all-NFAs accepts regular languages.

On the other side, to show that any language that is accepted by the all-NFA is regular, we first take an all-NFA $M = (Q, \Sigma, \delta, q_0, F)$ and construct a standard NFA $M' = (Q', \Sigma', \delta', q_0', F')$ that accepts the same language as M. Since we construct M' to have atmost a single path for every computation, we follow a process that is very similiar to the standard NFA to DFA construction, except for two differences in the construction.

First, for every dying path that M takes, there is an equivalent 'dying' path in M' as well. Second, if a string x ends at state $q' \in Q'$, it is accepted by M' if and only if ALL the states in q' belong to the accepting states of M. This is to ensure that M' accepts the string x only if M, upon reading the same string x, ends all it's branches in accepting states. Note that E(S) denotes all the states that could be reached from S using ε-transitions.

M':

$$Q' := P(Q)$$
$$q_0' := \{q_0\}$$
$$\delta'(R, a) := \begin{cases} \emptyset & \textit{if } \delta(r,a) = \emptyset, \textit{ for } r \in R\ , R \in Q\text{'} \\ \{q \in Q \mid q \in E(\delta(r,a))\ , \textit{for } r \in R\ , R \in \text{Q'}\} & \textit{otherwise} \end{cases}$$
$$F' := P(F)$$

Since the standard NFA M' accepts the same language as the all-NFA M, any language that is accepted by M is regular. Hence we have proved that all-NFAs recognize the class of regular languages. ■

4. Prove this language is not regular by providing a fooling set. Be sure to include the fooling set you construct is i) infinite and ii) a valid fooling set.

$$L_{P5} = \{w | w \text{ such that } |w| = \lceil k\sqrt{k} \rceil, \text{ for some natural number} k\}$$

Hint: since this one is more difficult, we'll even give you a fooling set that works: try $F = \{0^{m^6} | m \geq 1\}$. We'll also provide a bound that can help: the difference between consecutive strings in the language, $\lceil (k+1)^{1.5} \rceil - \lceil k^{1.5} \rceil$, is bounded above and below as follows

$$1.5\sqrt{k} - 1 \leq \lceil (k+1)^{1.5} \rceil - \lceil k^{1.5} \rceil \leq 1.5\sqrt{k} + 3$$

All that's left is you need to carefully prove that F is a fooling set for L.

Solution: Let F be the set $\{0^{m^6} | m \in \mathbb{N}\}$.

We can also write this as $\{0^{\lceil k\sqrt{k} \rceil} | k = m^4, m \in \mathbb{N}\}$. Note that each element in F is also an element in L.

Let $x = 0^{m^6}$ and $y = 0^{n^6}$ for some $m < n$.

Let z be the smallest string such that $xz \in L$. By the given bound, $|z| \leq 1.5m^2 + 3$.

Suppose for contradiction $yz \in L$. By the other side of the given bound, we would need $|z| \geq 1.5n^2 - 1$. We can show both of these contraints on z can't be satisfied, since $1 \leq m \leq n - 1$, so

$$1.5m^2 + 3 \leq 1.5(n-1)^2 + 3 = 1.5(n^2 - 2n + 1) + 3 = 1.5n^2 - 1 + (5.5 - 3n) \leq 1.5n^2 - 1$$

.

■

Solution: From my experience in office hours, I wanted to write another solution which clarifies a few things (since this is a difficult problem).

First let's start with the fooling set $F = \{0^{m^6} | m \geq 1\}$. This set is a subset of the language $L_{P5} = \{0^{m^6} | m \in \mathbb{N}\}$ but that's ok for us. If we prove that F has infinite distinguishable states, then it means L_{P5} has at least infinite distinguishable states which is a problem for L_{P5} being regular.

So that's the big picture but how do we get there? Well first let's consider two strings from the fooling set:

$$x = 0^{i^6}$$
$$y = 0^{j^6}$$

for $i < j$. So both these strings are part of the original language (assuming $k = i^4 or k = j^4$). But what about the next string in their sequence? Is there another run of zeros (z) that you can add to x such that $xz \in L_{P5}$. More importantly if x and y are distinguishable then it means $yz \notin L_{P5}$? If $L_{GoforthScientificInc}$ is not regular, then we need to prove that such a z cannot exist which let's xz & $yz \in L_{P5}$.

So let's do a **Proof by Contradiction** as we do with most fooling set problems.

- First let's look at xz which is the next largest run of zeros after x that belongs to L_{P5}.
  - Looking at the definition for L_{P5}, in order for $x \in L_{P5}$, $k = i^4$ which give us the string $x = 0^{i^6} = 0^{(i^4)^{1.5}}$.
  - So the next largest run of 0's in L_{P5} occurs when $k = i^4 + 1$ which would give us the string $xz = 0^{(i^4+1)^{1.5}}$.
  - This means that we can finding the length of z by

$$|xz| - |x| = |0^{(i^4+1)^{1.5}}| - |0^{(i^4)^{1.5}}| = (i^4+1)^{1.5} - (i^4+1)^{1.5} = |z|$$

  - According to boundaries given in the problem this means that

$$1.5\sqrt{i^4} - 1 = 1.5i^2 - 1 \leq |z| \leq 1.5i^2 + 3 = 1.5\sqrt{i^4} + 3 \tag{1}$$

- Next, because of the proof by contradiction we're assuming $yz \in L_{P5}$ as well. This is the next largest run of zeros after y that is in L_{P5}. Here we follow the exact steps as above but with j instead of i.
  - Looking at the definition for L_{P5}, in order for $y \in L_{P5}$, $k = j^4$ which give us the string $y = 0^{j^6} = 0^{(j^4)^{1.5}}$.
  - The next largest run of 0's in L_{P5} occurs when $k = j^4 + 1$ which would give us the string $yz = 0^{(j^4+1)^{1.5}}$.
  - This means that we can finding the length of z by

$$|yz| - |y| = |0^{(j^4+1)^{1.5}}| - |0^{(j^4)^{1.5}}| = (j^4+1)^{1.5} - (j^4+1)^{1.5} = |z|$$

  - According to boundaries given in the problem this means that

$$1.5j^2 - 1 \leq |z| \leq 1.5j^2 + 3 \tag{2}$$

- So we got some boundaries for z defined by xz and yz shown below.

  $1.5i^2 - 1$ $|z|$ according to (1) $1.5i^2 + 3$

  $1.5j^2 - 1$ $|z|$ according to (2) $1.5j^2 + 3$

  Now if the states of x and y are not distinguishable (i.e. both xz and yz can be in $L_{P5)}$), then there should be some value of z that both prefixes can follow to an accept state. Namely,

$$1.5j^2 - 1 \leq |z| \leq 1.5i^2 + 3 \tag{3}$$

- But wait! Didn't we say $i < j$? If $i > 0$ then (3) is impossible!

- Therefore, there is run of zeroes for z where both xz and yz would be in L_{P5}.
- x and y denote distinguishable states states of the language L_{P5}.
- Because F is infinite, the DFA representing L_{P5} would require infinite states which violates the definition of regular language and hence, L_{P5} can't be regular.

■