Exercise: (Section 3.3.2 from textbook)

Consider the following languages over \( \Sigma = \{0,1\} \)

\[
L_e = \{ \varepsilon(M) \mid \ell(M) = \emptyset \} \\
L_{\text{ne}} = \{ \varepsilon(M) \mid \ell(M) \neq \emptyset \}
\]

Hence: \( L_e \) ... set of all strings that encode T.M.'s that accept the empty language

\( L_{\text{ne}} \) ... complement of \( L_e \)

Claim 1: \( L_{\text{ne}} \) is R.E.

Proof: construct NTM \( N \) for \( L_{\text{ne}} \)

(and then convert \( N \) to an ordinary T.M.)

\( N \) works as follows: on input \( \varepsilon(M) \)

1) guess a string \( w \in \Sigma^* \)
2) simulate \( M \) on \( w \) (like a UTM)
3) accept \( \varepsilon(M) \) if \( M \) accepts \( w \)

\[
\varepsilon(M) \xrightarrow{\text{guessed } w} N \xrightarrow{\text{yes}} \text{yes}
\]

We have

\[
\varepsilon(M) \in L(N) \iff \exists w \text{ s.t. } \langle M, w \rangle \in L(U) \\
\iff \exists w \text{ s.t. } w \in \ell(M) \\
\iff \varepsilon(M) \in L_{\text{ne}}
\]
Claim 2: \( L_{rec} \) is non-recursively enumerable.

Proof: by reduction from \( L_{H} \) to \( L_{rec} \):

Reduction \( R \) is a function computable by a halting T.M.

- **Input**: instance \( <M, w> \) of \( L_{H} \)
- **Output**: instance \( \Sigma(M') \) of \( L_{rec} \)
- **End set**: \( <M, w> \in L_{H} \iff \Sigma(M') \in L_{rec} \)

Description of \( M' \):

- \( M' \) ignores completely its own input string \( x \).
- Instead, it replaces its input by the string \( <M, w> \) and runs \( M \) on \( w \) (see (*) below).

  - If \( M \) accepts \( w \), then \( M' \) accepts \( x \).
  - If \( M \) never halts on \( w \) or rejects \( w \), then \( M' \) also never halts on \( x \).

Note:

- If \( w \notin \Sigma(M) \Rightarrow \Sigma(M') = \Sigma^* \)
- If \( w \in \Sigma(M) \Rightarrow \Sigma(M') = \emptyset \)

hence \( <M, w> \in L_{H} \iff \Sigma(M') \in L_{rec} \)

We can construct a halting T.M. \( M_\epsilon \) that, given \( <M, w> \) as input, constructs \( \Sigma(M') \) for an \( M' \) that behaves as above.

(*) \( M' \) has the following form: (let \( w = a_1 a_2 \ldots a_n \))

- \( \delta(q_0, 1) = \delta(q_0, 1) = 1 \)
- \( \delta(q_0, i) = \delta(q_0, i) = 0 \)
- \( \delta(q_0, w) = q_0 \)
- \( \delta(q_0, a) = q_0 \)

To summarize, we have that \( L_{rec} \) is RE but non-recursively enumerable. Hence \( L_{rec} \) must be non-RE.
Exercise: 3.2.1

The halting problem \( H \), i.e., the set \( \langle M, w \rangle \) s.t. \( M \) halts on \( w \) (with or without accepting) is R.E., but not recursive.

To show R.E., we construct a T.M. \( H \), s.t.
\[
L(H) = L_H = \{ \langle M, w \rangle | M \text{ halts on } w \}
\]

\[
\langle M, w \rangle \rightarrow H \rightarrow \text{yes or no}
\]

To show that \( L_H \) is not recursive, we assume by contradiction it is no, and derive that \( L_H \) is recursive.

By contradiction, let \( U \) be an algorithm for \( L_H \), and \( U \) a procedure for \( L_H \)

\[
\langle M, w \rangle \rightarrow H \rightarrow A_m \rightarrow \text{yes or no}
\]

\( A_m \) would be an algorithm for \( L_H \). Contradiction
Let \( L \) be R.E. and \( \overline{L} \) be non-R.E.

Consider \( L' = \{0w | w \in L\} \cup \{1w | w \notin L\} \).

What do we know about \( L' \) and \( \overline{L'} \)?

We show that \( L' \) is non-R.E.

Suppose by contradiction that we have a procedure \( M_L \) for \( L' \).
Then we can construct a procedure \( M_{\overline{L}} \) for \( \overline{L} \) as follows:

- on input \( w \), \( M_{\overline{L}} \) changes the input to \( 1w \) and simulates \( M_L \).
- if \( M_L \) accepts \( 1w \), then \( w \in L \), and \( M_{\overline{L}} \) accepts.
- if \( M_L \) does not terminate or simulates \( M_L \) on \( w \notin L \), and \( M_{\overline{L}} \) does not terminate or simulates \( M_L \) on \( w \notin L \).

\[ \Rightarrow M_{\overline{L}} \text{ would accept exactly } \overline{L}. \text{ Contradiction.} \]

\( L' = \{0w | w \in L\} \cup \{1w | w \in L\} \cup \{\epsilon\} \)

Reasoning as for \( L' \), we get that \( \overline{L'} \) is non-R.E.
Exercise 3.3.7 a)

$H$, the complement of the halting problem, i.e.,
the set of pairs $<M, w>$ such that $M$ on input $w$
does not halt, is non-R.E.

Proof: By reduction from $T_m$, which is non-R.E.

Idea: we show how to convert any TM $M$ into another
TM $M_h$ such $M_h$ halts on $w$ iff $M$ accepts $w$.

Construction:

1) Ensure that $M_h$ does not halt unless $M$ accepts.
   - add to the states of $M$ a new loop state $p$, with
     $\delta(p, x) = (p, x, y, z)$ for all $x \in \Gamma$
   - for each $\delta(q, y)$ that is undefined and $q \in F$,
     add $\delta(q, y) = (p, z, y, z)$

2) Ensure that, if $M$ accepts, then $M_h$ halts.
   - make $\delta(q, x)$ undefined for all $q \in F$ and $x \in \Gamma$

3) The other moves of $M_h$ are as those of $M$.

qed.