Read Chapter 11 in Linz.

**Definition:** A language $L$ is *recursively enumerable* if there exists a TM $M$ such that $L=L(M)$.

**Definition:** A language $L$ is *recursive* if there exists a TM $M$ such that $L=L(M)$ and $M$ halts on every $w \in \Sigma^+$.  

**Enumeration procedure for recursive languages**

To enumerate all $w \in \Sigma^+$ in a recursive language $L$:

- Let $M$ be a TM that recognizes $L$, $L = L(M)$.
- Construct 2-tape TM $M'$
  - Tape 1 will enumerate the strings in $\Sigma^+$
  - Tape 2 will enumerate the strings in $L$.
  - On tape 1 generate the next string $v$ in $\Sigma^+$
  - simulate $M$ on $v$
  - if $M$ accepts $v$, then write $v$ on tape 2.
Enumeration procedure for recursively enumerable languages

To enumerate all $w \in \Sigma^+$ in a recursively enumerable language $L$:

Repeat forever

- Generate next string (Suppose $k$ strings have been generated: $w_1, w_2, ..., w_k$)
- Run $M$ for one step on $w_k$
  - Run $M$ for two steps on $w_{k-1}$.
  - ...
  - Run $M$ for $k$ steps on $w_1$.
- If any of the strings are accepted then write them to tape 2.

Theorem Let $S$ be an infinite countable set. Its powerset $2^S$ is not countable.

Proof - Diagonalization

- $S$ is countable, so it’s elements can be enumerated.
  $S = \{s_1, s_2, s_3, s_4, s_5, s_6, ...\}$
  An element $t \in 2^S$ can be represented by a sequence of 0’s and 1’s such that the $i$th position in $t$ is 1 if $s_i$ is in $t$, 0 if $s_i$ is not in $t$.
  Example, $\{s_2, s_3, s_5\}$ represented by
  Example, set containing every other element from $S$, starting with $s_1$ is $\{s_1, s_3, s_5, s_7, ...\}$ represented by

Suppose $2^S$ countable. Then we can enumerate all its elements: $t_1, t_2, ...$.

<table>
<thead>
<tr>
<th></th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_3$</th>
<th>$s_4$</th>
<th>$s_5$</th>
<th>$s_6$</th>
<th>$s_7$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$t_4$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$t_5$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$t_6$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$t_7$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
**Theorem** For any nonempty $\Sigma$, there exist languages that are not recursively enumerable.

**Proof:**

- A language is a subset of $\Sigma^*$.
  
  The set of all languages over $\Sigma$ is

---

**Theorem** There exists a recursively enumerable language $L$ such that $\bar{L}$ is not recursively enumerable.

**Proof:**

- Let $\Sigma = \{a\}$

  Enumerate all TM’s over $\Sigma$:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>aa</th>
<th>aaa</th>
<th>aaaa</th>
<th>aaaaa</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L(M_1)$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$L(M_2)$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$L(M_3)$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>$L(M_4)$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>$L(M_5)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The next two theorems in conjunction with the previous theorem will show that there are some languages that are recursively enumerable, but not recursive.

**Theorem** If languages \( L \) and \( \bar{L} \) are both RE, then \( L \) is recursive.

**Proof:**

- There exists an \( M_1 \) such that \( M_1 \) can enumerate all elements in \( L \).
- There exists an \( M_2 \) such that \( M_2 \) can enumerate all elements in \( \bar{L} \).

To determine if a string \( w \) is in \( L \) or not in \( L \) perform the following algorithm:

**Theorem:** If \( L \) is recursive, then \( \bar{L} \) is recursive.

**Proof:**

- \( L \) is recursive, then there exists a TM \( M \) such that \( M \) can determine if \( w \) is in \( L \) or \( w \) is not in \( L \). \( M \) outputs a 1 if a string \( w \) is in \( L \), and outputs a 0 if a string \( w \) is not in \( L \).
  
  Construct TM \( M' \) that does the following. \( M' \) first simulates TM \( M \). If TM \( M \) halts with a 1, then \( M' \) erases the 1 and writes a 0. If TM \( M \) halts with a 0, then \( M' \) erases the 0 and writes a 1.

Hierarchy of Languages:

```
all languages
  recursively enumerable languages
    recursive languages
      context-free languages
        regular languages
```
**Definition** A grammar $G=(V,T,S,P)$ is *unrestricted* if all productions are of the form $u \rightarrow v$

where $u \in (V \cup T)^+$ and $v \in (V \cup T)^*$

**Example:**
Let $G=\{\{S,A,X\},\{a,b\},S,P\}$, $P=$

$$
S \rightarrow bAaX \\
bAa \rightarrow abA \\
AX \rightarrow \lambda
$$

**Example** Find an unrestricted grammar $G$ s.t. $L(G)=\{a^n b^n c^n | n > 0\}$

$G=(V,T,S,P)$

$V=\{S,A,B,D,E,X\}$

$T=\{a,b,c\}$

$P=$

1) $S \rightarrow AX$
2) $A \rightarrow aAabc$
3) $A \rightarrow aBbc$
4) $Bb \rightarrow bB$
5) $Bc \rightarrow D$
6) $Dc \rightarrow CD$
7) $Db \rightarrow bD$
8) $DX \rightarrow EXc$

There are some rules missing in the grammar.

To derive string $aabbcc$, use productions 1,2 and 3 to generate a string that has the correct number of $a$'s $b$'s and $c$'s. The $a$'s will all be together, but the $b$'s and $c$'s will be intertwined.

$$
S \Rightarrow AX \Rightarrow aAbcX \Rightarrow aaAbcbX \Rightarrow aaaBbcbcbX
$$
Theorem If G is an unrestricted grammar, then L(G) is recursively enumerable.

Proof:

- List all strings that can be derived in one step.
  
List all strings that can be derived in two steps.

Theorem If L is recursively enumerable, then there exists an unrestricted grammar G such that L=L(G).

Proof:

- L is recursively enumerable.
  ⇒ there exists a TM M such that L(M)=L.
  
M = (Q, Σ, Γ, δ, q₀, B, F)

q₀w ⊢ x₁q_fx₂ for some q_f ∈ F, x₁, x₂ ∈ Γ*

Construct an unrestricted grammar G s.t. L(G)=L(M).

S ⊢ w

Three steps

1. S ⊢ B...B#x₁yB...B
   with x,y ∈ Γ* for every possible combination

2. B...B#x₁yB...B ⊢ B...B#q₀wB...B

3. B...B#q₀wB...B ⊢ w
Definition A grammar $G$ is *context-sensitive* if all productions are of the form

$$x \rightarrow y$$

where $x, y \in (V \cup T)^+$ and $|x| \leq |y|$.

Definition $L$ is context-sensitive (CSL) if there exists a context-sensitive grammar $G$ such that $L=L(G)$ or $L=L(G) \cup \{\lambda\}$.

Theorem For every CSL $L$ not including $\lambda$, $\exists$ an LBA $M$ s.t. $L=L(M)$.

Theorem If $L$ is accepted by an LBA $M$, then $\exists$ CSG $G$ s.t. $L(M)=L(G)$.

Theorem Every context-sensitive language $L$ is recursive.

Theorem There exists a recursive language that is not CSL.