A set $A$ is *enumerable* if there is a function $f : \mathbb{N} \rightarrow A$.

In this case $A$ can be written as a sequence: writing $a_i$ for $f(i)$, we have

$$A = \text{range}(f) = \{a_0, a_1, a_2, \ldots\} = \{a_i \mid i \geq 0\}.$$ 

To say that a set is enumerably is to say that its elements can be counted.
An **enumerator** is a T.M. that uses one tape as an output tape, on which a symbol, once written, can never be changed, and whose tape head never moves left. On the output tape, \( E \) writes strings over some alphabet \( \Sigma \), separated by a marker symbol \#.

\( E \) may write words in any order and may write words more than once.

\[ G(E) = \{ w | E \text{ eventually writes } w \text{ between a pair of } \# \text{'s} \} \]

**Definition.** A set \( L \) is **computably enumerable** iff \( L = G(E) \) for some enumerator \( E \) or \( L = \emptyset \).

**Theorem.** If \( L \) is \( \Sigma^* \), then \( L \) is Turing-acceptable.

**Proof.** Given \( L = G(E) \) for some enumerator \( E \). Construct \( M \) to have one more tape than \( E \). Place input \( w \) on this tape. \( M \) simulates \( E \); if \( E \) ever writes \( w \) on its output tape between \#'s, then \( M \) accepts.

\( L = \emptyset \) is an easy case.
Let \( P \) be an enumerator that generates ordered pairs of natural numbers \((i, j)\) in binary in the following order:

1. Generate in order of the sum \(i + j\);
2. Among pairs of equal sum, in order of increasing \(i\).

\((1,1), (1,2), (2,1), (1,3), (2,2), (3,1), \ldots\)

**Theorem.** If \( L \) is Turing-acceptable, then \( L \) is c.e.

**Proof.** Let \( L = L(M) \). Construct \( E \) to behave as follows: Simulate \( P \) when \( P \) generates \((i, j)\), then simulate \( M \) on \( w_i \) (the \(i\)-th word in lexicographical order for exactly \( j\) steps. If \( M \) accepts \( w_i \) in \( j\) steps, then write \( w_i \) on the output tape.