We can encode every Turing machine with a string over $\{0, 1\}$. For example, if M is the following TM:

$$(\{q_1, q_2\}, \{0, 1\}, \delta, \ldots)$$

where $\delta(q_1, 1) = (q_2, 0, \rightarrow), \ldots$, then it could be encoded as follows:

$$\underbrace{0}_{q_1} 1 \underbrace{00}_{1} 1 \underbrace{00}_{q_2} 1 \underbrace{0}_{0} 1 \underbrace{0}_{\rightarrow} 11 \ldots$$

Not every string is going to be a valid encoding of a TM (for example the string 1 does not encode anything in our convention). So, let all those bad strings encode a default TM M_{default} which has one state, and halts immediately, so $L(M_{\text{default}}) = \emptyset$.

Let the following be a list of all strings of 0s and 1s (in *lexicographic* ordering):

$$0, 1, 00, 01, 10, 11, 000, 001, 010, 100, 101, \ldots$$

Let $w_i \in \{0, 1\}^*$ be the i-th string on this list:

$$w_1, w_2, w_3, w_4, w_5, \ldots, w_i, \ldots$$

For example, $w_8 = 001$.

Then, let M_i be the Turing machine encoded by w_i. If w_i is a valid encoding, then M_i is the corresponding TM, but if w_i is garbage, then $M_i = M_{\text{default}}$.

In other words:

$$M_i = \begin{cases} \text{TM described by } w_i & \text{if } w_i \text{ is valid} \\ M_{\text{default}} & \text{otherwise} \end{cases}$$

For example, $M_2 = M_{\text{default}}$ as $w_2 = 1$ which is not a valid encoding of a TM.

Thus, we have the corresponding list of TMs:

$$M_1, M_2, M_3, M_4, M_5, \ldots, M_i, \ldots$$

Any TM under the sun is somewhere in this list.

A language L is **recursive** if there exists a TM M that halts in an accepting state for all $x \in L$, and halts in a non-accepting state for all $x \notin L$. In other words, L is recursive if there exists a TM M such that $L = L(M)$, and M *halts on all inputs*.

Recursive languages correspond to languages that can be recognized *algorithmically*.

A language L is **recursively enumerable (RE)** if there exists a TM M that halts in an accepting state for all $x \in L$. In other words, L is RE if there exists an M such that $L = L(M)$ (but M does not necessarily halt on all inputs).

L_d, the **diagonal language**, is the set of strings w_i such that M_i does not accept w_i.

In words, L_d is the set of strings which encode TM which do not accept their own encoding when presented to them as input.

Formally,

$$L_d = \{w | w = w_i \text{ and } M_i \text{ does not accept } w_i\}$$

Theorem 0: L_d is not RE.

Proof: Suppose that $L_d = L(M)$, for some M. Since all TM under the sun are encoded somewhere on the list $w_1, w_2, w_3, \ldots$, there exists a w_i that encodes M, i.e., $M = M_i$.

Is $w_i \in L_d$?

If $w_i \in L_d$, then $M(= M_i)$ accepts w_i, but then by def of L_d, $w_i \notin L_d$.

If $w_i \notin L_d$, then $M(= M_i)$ does not accept w_i, so $w_i \in L_d$.

Contradiction, so our assumption that L_d is RE is wrong.

Theorem 1: If L is recursive, so is $\bar{L} = \Sigma^* - L$.

Proof: If L is recursive, then it is decided by M. Let M' be the following modification of M: on input x, M' runs just like M.

However, when M is about to enter an accepting state, M' enters a new state q_{reject}, and when M is about to halt in a non-accepting state, M' enters a new state q_{accept}.

Clearly, $\bar{L} = L(M')$. Since M always halts, so does M', so by definition, $\bar{L}$ is recursive.

Theorem 2: If both L and $\bar{L}$ are RE, then L is recursive.

Proof: Let $L = L(M_1)$ and $\bar{L} = L(M_2)$. Let M be the following TM: on input x it simulates M_1 on x on one tape, and M_2 on x on another tape.

Since $L \cup \bar{L} =$everything, x must be accepted by one or the other.

If M_1 accepts, so does M,

if M_2 accepts, M rejects.

L_U, the **Universal Language**, is the set of binary strings that encode pairs (M, w) such that $w \in L(M)$.

That is, L_U is the set of all those strings of the form $w_i 111 w_j$ where the TM M_i accepts the input w_j.

We show that L_U is accepted by U, the **Universal Turing machine**, i.e., $L_U = L(U)$.

U is simply a TM which on input $w_i 111 w_j$ starts simulating the TM M_i on w_j and accepts iff M_i accepts. If the input to U, say x, is not of the form $w_i 111 w_j$, then it just rejects.

Theorem: L_U is RE but not recursive.

Proof: Since $L_U = L(U)$, by definition L_U is RE.

Suppose that L_U is also recursive.
Then so is $\bar{L}_U$ by Theorem 1.
Then there exists a TM M which decides $\bar{L}_U$, i.e., M always halts, and $\bar{L}_U = L(M)$.

Let M' be a TM which does the following: on input w_i, M' runs just like M would on $w_i 111 w_i$. *Since M exists (by assumption) so does M'.*

M accepts $w_i 111 w_i$ if M_i does not accept w_i (because $L(M) = \bar{L}_U$).

Therefore, M' accepts w_i iff M_i does not accept w_i, and THEREFORE, $L(M') = L_d$.

But there is no such M' (By Theorem 0)!
Contradiction, so L_U is not recursive.