**Introduction**

When a Turing machine *T* operates on an input string *S*, there are three outcomes, these are;

* It halts and accepts the string.
* It halts and rejects the string.
* Never halts, proceeds infinitely.

**What are Recursive languages?**

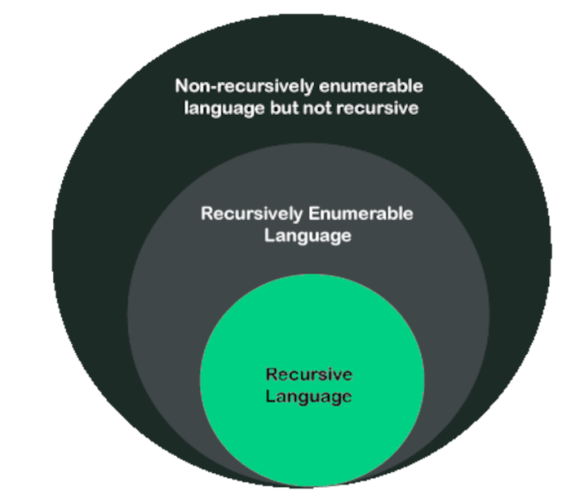
We refer to a language *L* as recursive if there exists a Turing machine *T* for it. In this case, the Turing machine accepts every string in language *L* and rejects all strings that don't match the alphabet of *L*.

In other words, if string *S* is part of the alphabet of language *L*, then the Turing machine *T* will accept it otherwise the Turing machine halts without ever reaching an accepting state.

**Recursively enumerable languages**

Here if there is a Turing machine *T* that accepts a language *L*, the language in which an enumeration procedure exists is referred to as a recursively enumerable language.

Note that some recursive languages are enumerable and some enumerable languages are recursive.



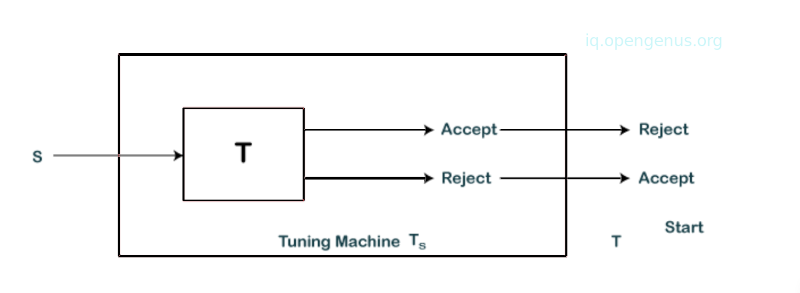
## Differences between recursive and recursively enumerable languages

| **Factor** | **(Recursive / Turing Decidable Languages)** | **(Recursively Enumerable / Turing Recognizable Languages)** |
| --- | --- | --- |
| Examples | Context-sensitive languages | RE languages. |
| States | Halt-Accept, Halt-Reject | Halt-accept, Halt-Reject, Infinite Loop(No halting) |
| Looping | Finite loops | Possibility of infinite loop |
| Accept/reject | Accept (Turing machine) = L, Reject (Turing machine) = L, Loop (Turing machine) = φφ, φ = null φ = null | Accept (Turing machine) = L,  Reject (Turing machine) + Loop (Turing machine) = L’ |

**Properties of both recursive and recursively enumerable languages**

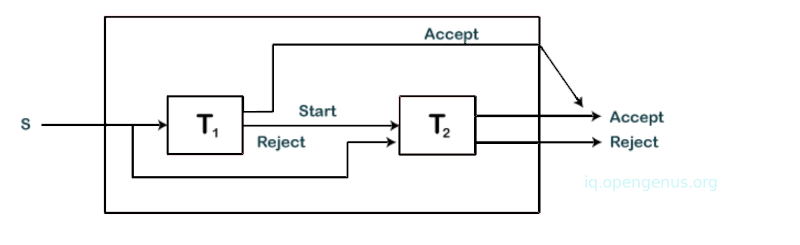
We will state theorems which are also properties of both languages.

1. **If language *L* is recursive, its complement *L'* is also recursive.**  
   *Proof:*  
   *L* is a language accepted by a Turing machine that halts on all inputs. We construct a Turing machine *Ts* from *T* as shown below:



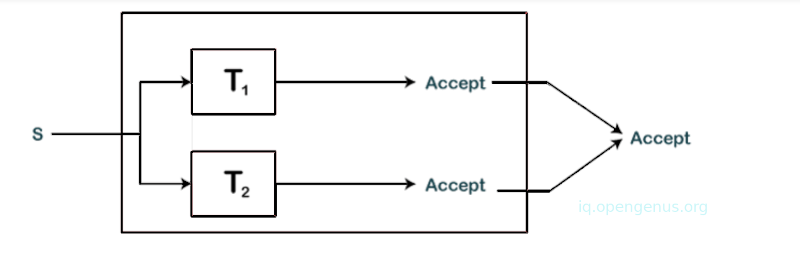
We see that Turing machine *T* given an input string *S* enters into an accepting state then *Ts* rejects and halts for string *W*. Also, if the Turing machine *T* halts without accepting *W*, *Ts* enters into an accepting state. *Ts* accept strings that are not accepted by *T*. Therefore, *Ts* recognizes the complement of *L*.

1. **If the languages *L1* and *L2* are recursive, their union *L1 U L2* is also recursive.***Proof:*  
   We have two Turing machines *T1* and *T2* that recognize languages *L1* and *L2*. We construct a Turing machine *T* as shown:



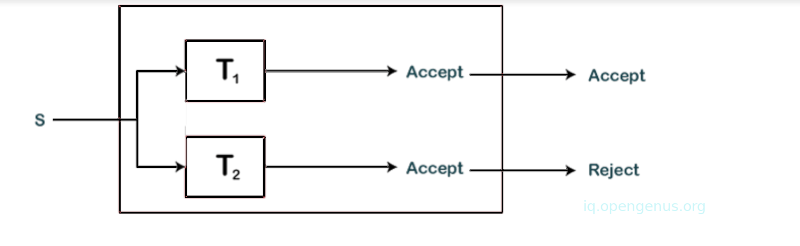
*T* simulates *T1* and *T* accepts input *S* is *T1* accepts it also. On the other hand, if *T1* rejects, *T* simulates *T2* and accepts if *T2* accepts.  
Both *T1* and *T2* are algorithms and therefore they will halt at some point. We conclude that *T* accepts *L1 U L2*.

1. The union of any two recursively enumerable languages is also a recursively enumerable language.  
   *Proof:* We have two recursively enumerable languages *L1* and *L2* that are accepted by Turing machines *T1* and *T2*. We construct a Truing machine *T* as shown below.



The machine simultaneously simulates *T1* and *T2* on separate tapes. If either accepts *S* then machine *T* also accepts *S*.

1. We have a language *L* and its complement *L'*, a recursively enumerable language. Then *L’* will also be a recursive language.  
   *Proof:*  
   We have two Turing machines *T1* and *T2* that recognize languages *L* and its complement *L'*. We construct a Turing machine *T* as shown:



The machine *T* simulates *T1* and *T2* parallelly. States of *T1* and *T2* are components of the state of turing machine *T*. If *T1* accepts *S*, *T* accepts *S* also, if *T2* accepts *S*, *T* rejects *S*. This is so since *S* can either be part of *L* or part of *L'* therefore a single machine between *T1* and *T2* is expected to accept *S*.  
From that, we learn that *T* will always accept or reject either but never both. Since *T* is an algorithm that accepts *L* we say that language *L* is recursive.

**Summary**

We have learned about the properties of recursive and recursively enumerable languages in terms of *union*, *intersection* and *complements*.

A language *L* as *recursive* if there exists a Turing machine *T* for it. The Turing machine accepts every string in language *L* and rejects all strings that don't match the alphabet of *L*

On the other hand, if there is a Turing machine *T* that accepts a language *L*, the language in which an enumeration procedure exists is referred to as a *recursively enumerable language*.