Theorem 5.4. *Let $T_0, T_1, T_2, \ldots,$ be the standard Turing machines, as described above. Let K be the language over the alphabet $\{a\}$ defined by*

$$K = \{a^n \mid T_n \text{ halts when run with input } a^n\}.$$

Then K is a recursively enumerable language, but K is not recursive. The complement

$$\overline{K} = \{a^n \mid T_n \text{ does not halt when run with input } a^n\}.$$

is a language that is not recursively enumerable.

$$K = \{a^n \mid T_n \text{ halts when run with input } a^n\}.$$

- K is recursively enumerable because there's a Turing machine M which accepts it –
  - let M be a Turing machine which, given input a^n –
    - copies the input and runs G on the first copy of a^n, producing a symbolic description of the Turing machine T_n
    - runs U to simulate the computation of T_n on (the second copy of) input a^n
  - this simulation ends if and only if T_n halts when run with input a^n i.e. $a^n \in K$

$$K = \{a^n \mid T_n \text{ halts when run with input } a^n\}.$$

- K is not recursive because there's not a Turing machine M which decides it –
  - let H be any Turing machine
  - let M be a Turing machine which does the same thing as H until H halts (if H halts)
    - if H halts with output 1, M goes into an infinite loop
    - otherwise (any other output) M halts
  - assume, without loss of generality, that M is a standard Turing machine i.e. $M = T_n$ for some $n \in \mathbb{N}$
  - run $M = T_n$ on input a^n –
    - if H halts with output 1 on input a^n, T_n doesn't halt on input a^n
    - if H halts with output 0 on input a^n, T_n halts on input a^n
    - (what happens with other output or if H doesn't halt doesn't matter)
  - we need to show that H doesn't decide K
    - if H decides K, running H on a^n means that it should halt with output 1 if $a^n \in K$, that is, T_n halts when run with input a^n, and output 0 if $a^n \notin K$, that is, T_n doesn't halt when run with input a^n
    - but that's not what happens – H doesn't give the right answer, and so a Turing machine H that decides K doesn't exist

Theorem 5.4. *Let $T_0, T_1, T_2, \ldots,$ be the standard Turing machines, as described above. Let K be the language over the alphabet $\{a\}$ defined by*

$$K = \{a^n \mid T_n \text{ halts when run with input } a^n\}.$$

Then K is a recursively enumerable language, but K is not recursive. The complement

$$\overline{K} = \{a^n \mid T_n \text{ does not halt when run with input } a^n\}.$$

is a language that is not recursively enumerable.

- $\overline{K}$ is not recursively enumerable because if both K and $\overline{K}$ were, K would be recursive

Theorem 5.3. *Let Σ be an alphabet and let L be a language over Σ. Then L is recursive if and only if both L and its complement, $\Sigma^* \smallsetminus L$, are recursively enumerable.*

The Halting Problem

$$K = \{a^n \mid T_n \textit{ halts when run with input } a^n\}.$$

- deciding *K* is known as the *halting problem*
- no Turing machine – and thus no computer program – can solve this problem
  - it is *computationally unsolvable*
  - note that this doesn't mean that no instances of the problem can be solved, just that no Turing machine (or program) can produce the correct answer in all cases
- the halting problem is not the only computationally unsolvable problem
  - e.g. does a particular Turing machine halt for all possible inputs?
  - e.g. does a program halt with a particular input?
  - e.g. are two Turing machines (or programs) equivalent, that is, do they produce the same output for each possible input?
  - e.g. will a particular Turing machine halt if started with a blank tape?