  Uma **linguagem regular** é uma [linguagem formal](https://stringfixer.com/pt/Formal_language) que pode ser determinada por uma [expressão regular](https://stringfixer.com/pt/Regular_expression) ,  ou seja, uma linguagem produzida utilizando as operações de [concatenação](https://pt.wikipedia.org/wiki/Concatena%C3%A7%C3%A3o) e [união](https://pt.wikipedia.org/wiki/Uni%C3%A3o_(matem%C3%A1tica)) sobre os elementos de um alfabeto.

As linguagens regulares são utilizadas para descrever dispositivos que realizam computações simples, como os [autômatos finitos](https://pt.wikipedia.org/wiki/Aut%C3%B4mato_finito_alternado), pois representam a linguagem mais elementar classificada pela hierarquia de Chomsky que não requer [memória](https://pt.wikipedia.org/wiki/Mem%C3%B3ria_(inform%C3%A1tica)) para ser reconhecida.

Como descrição formal temos:

A coleção de línguas regulares sobre um [alfabeto](https://stringfixer.com/pt/Alphabet_(formal_languages)) Σ é definida recursivamente da seguinte forma:

* O idioma vazio Ø é um idioma regular.
* Para cada *a* ∈ Σ ( *a* pertence a Σ), a linguagem [singleton](https://stringfixer.com/pt/Singleton_(mathematics)) { *a* } é uma linguagem regular.
* Se *A* é uma linguagem regular, *A* \* ( [estrela de Kleene](https://stringfixer.com/pt/Kleene_star) ) é uma linguagem regular. Devido a isso, a linguagem de string vazia {ε} também é regular.
* Se *A* e *B* são linguagens regulares, então *A* ∪ *B* (união) e *A* • *B* (concatenação) são linguagens regulares.
* Nenhum outro idioma acima de Σ é regular.

Exemplo de uma linguagem que pertence a linguagem regular:

L1 = { ambn | m,n > 0}

Essa linguagem pertence a uma linguagem regular pois, ele pode adicionar quantos elementos forem precisos de “a’s” seguidos de “b’s” sem precisar conta-los e também é possível criar um autômato finito para essa linguagem.

Exemplo de uma linguagem que não pertence a linguagem regular:

L1 = { ambncz | m = z}

Essa linguagem não pertence a uma linguagem regular pois, ele tem que ter a mesma quantidade de “a’s” e de “z’s”, assim é preciso que seja contado a quantidade de elementos de “a’s” para que seja possível adicionar a mesma quantidade de “z’s”. Desse modo, a linguagem não consegue contar e nem criar uma autômato finito para a linguagem.

Uma **linguagem livre de contexto**(LLC) é uma [linguagem](https://pt.wikipedia.org/wiki/Linguagem_formal) gerada por alguma [gramática livre de contexto](https://pt.wikipedia.org/wiki/Gram%C3%A1tica_livre_de_contexto) (GLC). Diferentes gramáticas livres de contexto podem gerar a mesma linguagem livre de contexto, ou, inversamente, uma dada linguagem livre de contexto pode ser gerada por diferentes gramáticas livres de contexto, percebendo assim que existe uma ambiguidade.

O conjunto de todas as linguagens livres de contexto é idêntico ao conjunto de linguagens aceitas por um [autômato de pilha](https://pt.wikipedia.org/wiki/Aut%C3%B4mato_de_pilha).[[1]](https://pt.wikipedia.org/wiki/Linguagem_livre_de_contexto#cite_note-UFRN-1), o que faz com que essas linguagens já consiga realizar alguns tipo de contagem. Na verdade, dada uma GLC, há uma maneira direta para produzir um autômato com pilha para a gramática como uma linguagem correspondente.

A regras de uma gramática livre de contexto são da forma que do lado esquerdo das produções ficam símbolos não terminais (chamados de variáveis), e do lado direito fica uma cadeia de terminais e não terminais ou somente terminais ou somente não terminais. A gramática permite que regras com o mesmo símbolo a esquerda sejam compactadas.

A diferença da linguagem livre de contexto para a linguagem regular é que a livre de contexto possui uma memória possibilitando contagem e a regular não, sendo quem uma usa autômatos finitos e outra utiliza autômato de pilha .

Exemplo de uma linguagem que pertence a linguagem livre de contexto:

L3 = { cdsr | d = r}

Essa linguagem pertence a uma linguagem livre de contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s” e em seguida adicionar a mesma quantidade de “b’s” por ser possível contar os elementos e também é possível criar um autômato de pilha para essa linguagem que é responsável por permitir contar a quantidade de elementos.

Exemplo de uma linguagem que não pertence a linguagem livre de contexto:

L4 = { dmfngz | m = z}

Essa linguagem pertence a uma linguagem livre de contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s” seguida adicionar a quantidade de “b’s” for preciso, em seguida é preciso adicionar os “z’s” na mesma quantidade de “a’s” porém existe uma quantidade de “b’s” entre os “a’s” e os “z’s” assim não podendo utilizar um autômato de pilha para que seja possível ir retirando os “a’s” para contar sua quantidade de adicionar os “z’s” na mesma quantidade, mostrando assim que está linguagem não é livre de contexto.

Define-se inicialmente linguagem sensível ao contexto como sendo aquela que possa ser definida através de uma gramática sensível ao contexto.

Assim as regras que compõem uma gramática sensível ao contexto são que:

- Tem que possuir pelo menos um não terminal do lado esquerdo.

- Precisa possuir uma sequência de símbolos do lado direito sempre maior ou igual ao lado esquerdo da mesma regra.

- Última regra é que só pode ser adicionada vazio a uma única regra e que a variável dessa regra não apareça de novo em outro lugar.

Assim temos as restrições de criação de uma gramática sensível contexto.

A diferença da linguagem sensível ao contexto para a linguagem livre de contexto é que na sensível ao contexto é possível ter elementos na esquerda das regras da gramática que possua não terminais e terminais juntos, como também possibilita a troca de mais de um símbolo.

Exemplo de uma linguagem que pertence a linguagem sensível ao contexto:

L5 = { anbncn| n > 0}

Essa linguagem pertence a uma linguagem sensível ao contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s”, em seguida adicionar a mesma quantidade de “b’s” e em seguida adicionar a mesma quantidade de “c’s” por ser possível adicionar todos ao mesmo tempo e a gramática possibilitar alterar as posições do elementos da strign deixando-os na posição que for preciso.

Exemplo de uma linguagem que não pertence a linguagem sensível ao contexto:

L6 = { anbm| m = n2}

Essa linguagem não pertence a uma linguagem sensível ao contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s”, porém ela não consegue adicionar a quantidade necessárias de “b’s”, pois será preciso usar exponenciação e para isso acontecer é quebrado as regras da gramática sensível ao contexto, como por exemplo o lado esquerdo apenas terminas e isso na gramática sensível ao contexto não pode.

A linguagem recursiva é chamada recursiva se é um [subconjunto recursivo](https://pt.wikipedia.org/wiki/Conjuntos_recursivamente_enumer%C3%A1veis) no [conjunto](https://pt.wikipedia.org/wiki/Conjunto) de todas as palavras possíveis sobre o [alfabeto](https://pt.wikipedia.org/wiki/Alfabeto) da [linguagem](https://pt.wikipedia.org/wiki/Linguagem_formal). Equivalentemente, uma [linguagem](https://pt.wikipedia.org/wiki/Linguagem_formal) é recursiva se existe uma máquina de Turing que sempre pare quando recebe uma sequência finita de símbolos do alfabeto da linguagem como entrada e que aceita exatamente as palavras do alfabeto da linguagem que são parte da linguagem e rejeita todas as outras palavras.

A diferença da linguagem recursiva para a linguagem sensível ao contexto é que é impossível de a esquerda das regras da gramática seja possível apenas terminais, ou seja possível colocar vazio em qualquer lugar, como também que o lado esquerdo da regra seja maior do que o lado direito da regra, ou seja, na recursiva é possível fazer tudo.

Exemplo de uma linguagem que pertence a linguagem recursiva:

L6 = { anbm| m = 2n}

Essa linguagem pertence a uma linguagem sensível ao contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s” e ir adicionando a quantidade de “b’s” de acordo com a quantidade de “a’s” , ou seja, para cada “a” adicionado é adicionado o dobro de “b’s”, isso só é possível pois é possível percorrer a string.

Uma linguagem recursivamente enumerável formal é um [subconjunto](https://pt.wikipedia.org/wiki/Subconjunto) recursivamente enumerável no conjunto de todas as palavras possíveis sob o alfabeto da linguagem.

Uma linguagem recursivamente enumerável é uma linguagem formal para a qual existe uma [máquina de Turing](https://pt.wikipedia.org/wiki/M%C3%A1quina_de_Turing) que irá enumerar todas as cadeias válidas da linguagem. Note que, se a linguagem é infinita, o [algoritmo](https://pt.wikipedia.org/wiki/Algoritmo) de enumeração fornecido pode ser escolhido de forma que evite repetições, uma vez que podemos testar se a cadeia produzida para o número n já é produzida para um número que é inferior a n. Se ela já é produzida, use a saída da entrada n+1 (recursivamente), mas mais uma vez, teste se é uma cadeia nova.

Uma linguagem recursivamente enumerável é uma linguagem formal para a qual existe uma máquina de Turing que irá parar e aceitar quando se roda com qualquer cadeia da linguagem na entrada e pode parar e rejeitar ou entrar em loop quando se roda com qualquer cadeia que não é da linguagem.

Esta é a diferença entre [linguagem recursiva](https://pt.wikipedia.org/wiki/Linguagem_recursiva), que exige que a máquina de Turing sempre pare.

Exemplo de uma linguagem que pertence a linguagem recursivamente enumeraveis:

Como a linguagem abrange todas as outras vou da um exemplo parecido ao da linguagem recursiva.

L6 = { anbm| m = n2}

Essa linguagem pertence a uma linguagem sensível ao contexto pois, ele pode adicionar quantos elementos forem precisos de “a’s” e ir adicionando a quantidade de “b’s” de acordo com a quantidade de “a’s” , ou seja, para cada “a” adicionado é adicionado o dobro de “b’s”, isso só é possível pois é possível percorrer a string.