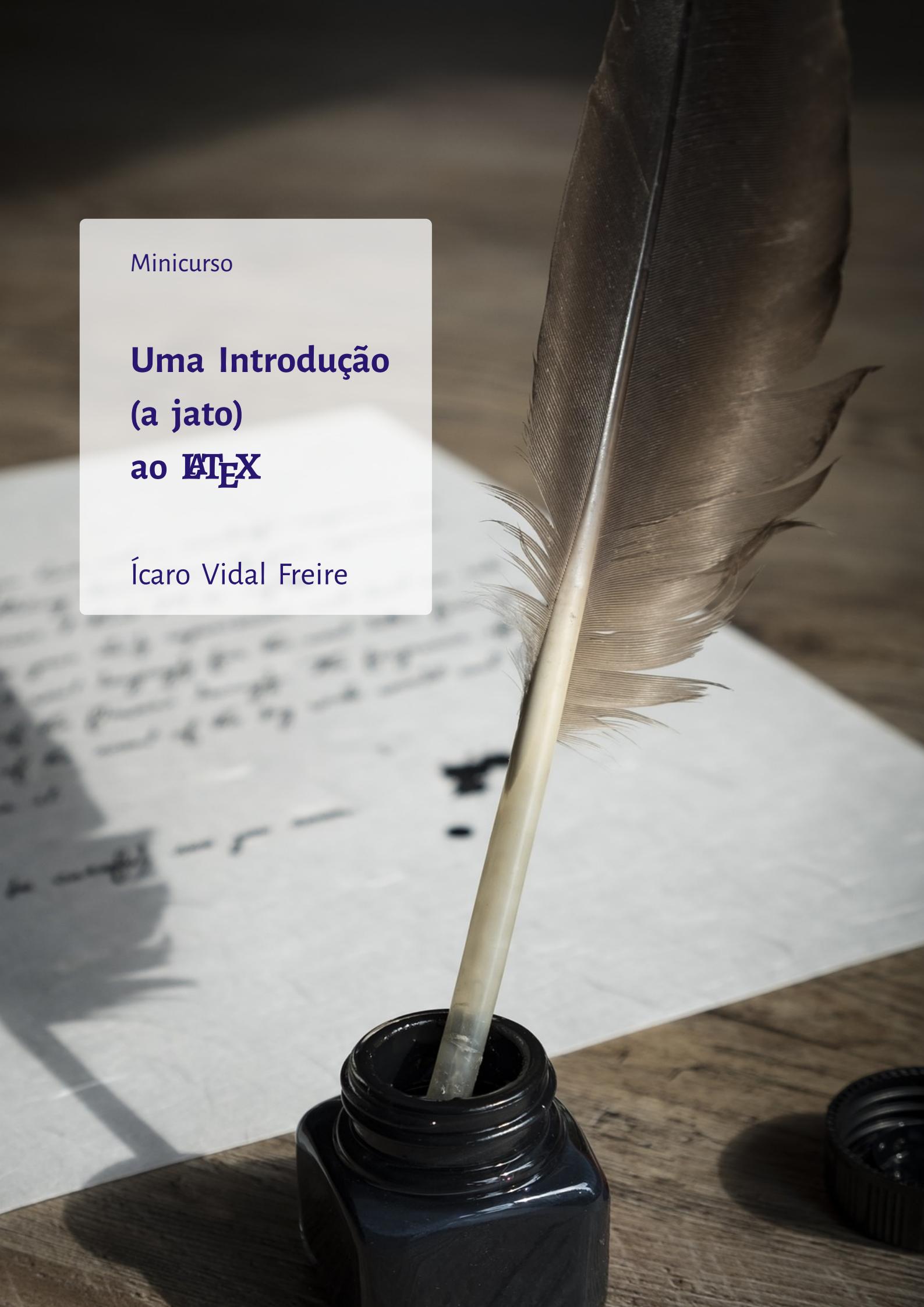


Minicurso

Uma Introdução (a jato) ao \LaTeX

Ícaro Vidal Freire



Sumário

I A Arte de Nomear as Coisas	5
1 A Seiva da Árvore ...	5
1.1 Diferenciando os nomes	5
1.2 O surgimento do \TeX	6
1.3 Mais e mais nomes: compiladores, interpretadores e formatos	6
1.4 Como colocar o \TeX em meu computador: as Distribuições	9
2 Sim ...mas como eu uso?	9
2.1 Como fazemos a “compilação”?	11
2.2 Editores para \TeX	12
2.2.1 O \TeX é outro Word?	12
2.2.2 Indicações de editores para \TeX	12
3 Overleaf: um editor <i>online</i> notável	13
II Aprendendo a Linguagem LaTeX	17
4 Uma Linguagem	17
4.1 Um pouco sobre a estruturação do LaTeX	18
4.1.1 Comandos	18
4.1.2 Classe do Documento	18
4.1.3 Mais sobre Comandos	20
4.1.4 Ambientes	21
4.1.5 Pacotes	22
4.2 Arrumando as coisas	26
III Começando a Escrever: O Modo Texto	29
5 Texto e Código	29
6 Algo (aquém do mínimo) sobre Fontes	30
6.1 Tamanho da Fonte	31
6.2 Ênfase na Fonte	32
7 Alinhamento	33

8 Algo sobre Listas	34
8.1 Description	34
8.2 Itemize	34
8.3 Enumerate	35
9 Figuras e Tabelas: uma noçãozinha de nada	36
9.1 Elementos Flutuantes	36
9.2 Tabelas	37
9.3 Figura	39
10 Rodapé e Minipage	43
10.1 Rodapé	43
10.2 Minipage	43
IV A Beleza do L^AT_EX: o Modo Matemático	45
II Teorema	45

Bate-papo Inicial

Se você optou por ler esse material, vá até o fim! O \TeX pode ser para você uma ferramenta extremamente agradável para produção tipográfica, a nível profissional, de suas futuras notas de aula; monografias, dissertações ou teses; listas de atividades; partituras; marcações no xadrez; etc.

Não pretendo abordar muita coisa, confesso.

Fui incumbido de ministrar, em 4h, uma introdução ao $\text{\TeX} 2\varepsilon$.

Esse texto não é um tutorial! Apenas um bate-papo sobre coisas que acho importante aprender desde o início da caminhada do aprendizado com o \TeX .

Falta terminar!



A Arte de Nomear as Coisas

1 A Seiva da Árvore ...

Lembro-me da primeira vez que vi o nome \LaTeX ...

Foi em uma chamada de um minicurso de alguma “semana de matemática” da universidade que fiz graduação. O título era algo assim: “Introdução ao \LaTeX ”.

Eu estava a um semestre de concluir a graduação e pensei: “Rapaz ...acho que eles erraram esse minicurso. Pra quê estudar látex em Matemática? Isso está mais para Geografia.”

Sim! Eu pensei que estavam falando daquela substância espessa e branca que sai de algumas plantas (seringueira, por exemplo)!

Então, vamos deixar as coisas claras: não estamos falando de látex, mas de \LaTeX .

Como pronunciar corretamente?

Aliás, como se pronuncia a palavra \LaTeX ?

Existem, pelo menos, duas maneiras de pronunciarmos, corretamente, essa palavra: “LeiTéc” ou “LaTéc”, ou seja, o som de \TeX (“Téc”) é o mesmo que na palavra “tecnologia”. Particularmente, adoto a segunda opção. Evite, por amor a Deus, falar “Látecks”. 😊

Falando nisso, essa palavra \TeX não está destacada de forma aleatória! Ela foi idealizada como sendo a junção de três letras gregas: τ , ϵ e χ .¹ Esse núcleo grego gera palavras como *arte* ($\tauέχνη$) ou mesmo *tecnologia* ($\tauεχνολογία$). Daí vem o espírito da palavra \TeX : unir uma arte (a tipografia) com a tecnologia (programação) para produzir documentos com uma beleza realmente ímpar.

Na realidade, para falarmos com propriedade sobre \TeX , precisamos tangenciar o \TeX .

1.1 Diferenciando os nomes

Tudo começou quando **Donald Ervin Knuth** queria qualidade nos elementos tipográficos de seus livros, principalmente na escrita matemática. Ele então criou, em meados da década dos anos 70, um sofisticado programa para a composição

Tabela 1 . Sumário da PART I

Seção 1. A seiva da árvore

Seção 2. Sim...mas como eu uso?

Seção 3. Overleaf: um editor online notável

Figura 1 . Isso é látex, não \TeX



Fonte: Globo Rural

¹ τ (tau), ϵ (épsilon) e χ (chi)

tipográfica de textos científicos e uma alternativa quase necessária para edição de textos com conteúdo matemático. Nasceu o \TeX . Em suas próprias palavras:

\TeX é destinado para a criação de belos livros - e, especialmente, para os livros que contêm grande quantidade de matemática.

Todavia, ao que parece, essa linguagem de programação não era tão acessível a meros mortais como nós, por conter diversos parâmetros relativos ao formato final do texto ...Bom, vamos falar a verdade: dava muito trabalho para digitar o que você queria simplificar. Isso porque o \TeX era formado por objetos denominados “primitivos” e toda estruturação do texto deveria ser feita por meio deles. O próprio Knuth criou um conjunto de macros, ou seja, um mapeamento de sequências de comandos primitivos, frequentemente utilizados, para simplificar a escrita de comandos \TeX em seus livros. A esse conjunto de macros ele deu o nome de *Plain \TeX* .

Obviamente, o *Plain \TeX* é um conjunto de macros bem simples e que poderia ser expandido. E foi isso que aconteceu.

1.2 O surgimento do \TeX

Leslie B. Lamport, facilitou nossa vida! Por volta dos anos 80, ele criou um conjunto de macros para o \TeX , muito bem estruturado, e com ideias interessantes (classes e pacotes, por exemplo) que somaram substancialmente à causa do \TeX , formando assim o \LaTeX . Inclusive, o “La”, de \LaTeX ; vem do “La”, de **Lamport**.¹

Logo, nunca se esqueça disso:

O \LaTeX veio para facilitar sua vida!

1.3 Mais e mais nomes: compiladores, interpretadores e formatos

Bom ...já sabemos que \TeX e \LaTeX são coisas diferentes, entretanto, interrelacionadas: o segundo é a forma mais utilizada, atualmente, para interagir com o primeiro.

Nesse ponto, seria interessante diferenciarmos *engines* (motores/interpretadores) de *formats* (formatos).

O \TeX é um interpretador (*engine*); o \LaTeX , um formato (*format*).

Os *interpretadores* são os arquivos binários executáveis (ou seja, o programa em si); já os *formats* são macros (comandos ou instruções), baseadas em \TeX , que usamos para escrever nossos documentos (a grosso modo, são linguagens ou atalhos para sequências de comandos ou estruturas em \TeX).

Não é difícil perceber que o interpretador \TeX é bem antigo e que outros tênhiam surgido ao longo desses anos. De fato, à época do \TeX , nem existia ainda

Saiba mais

Uma *macro* (abreviação para macroinstrução), em ciência da computação, é uma regra ou padrão que especifica como uma certa “sequência de entrada” deve ser mapeada para uma substituição de “sequência de saída” de acordo com um procedimento definido.

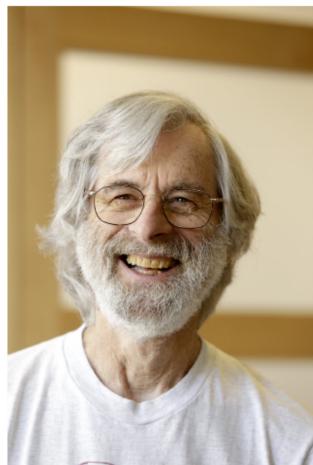
¹ $\text{\LaTeX} = \text{Lamport} + \text{\TeX}$

Saiba mais

Um outro formato que vimos é o *Plain \TeX* ; e, atualmente, um formato que se destaca é o **ConTeXt**.



(a) Donald E. Knuth



(b) Leslie B. Lamport

Figura 2 . Os culpados!

arquivos com extensão `.pdf` – a saída dos documentos era em DVI. NotesTeX. Quando surgiu o PDF, um interpretador ficou bem conhecido: pdfTeX – cuja saída poderia ser em DVI que poderia ser convertida em PDF. O pdfTeX talvez seja o mais usado dentre os interpretadores que vamos falar, pois já vem selecionado, por padrão, em muitos editores dedicados ao \TeX^2 ; as pessoas, simplesmente, não mudam a configuração padrão se realmente não for necessário. Aliás, nosso sistema relaciona o \TeX com o pdfTeX usando um “atalho”, denominado pdf \TeX .

Portanto, o pdf \TeX (que geralmente é o nome que aparecerá em nossos editores) é o *interpretador* pdfTeX com o *formato* \TeX .

Atualmente, dois interpretadores que se destacam são XeTeX e LuaTeX. Além de serem mais rápidos, trazem implementações notáveis como seleção de fontes do próprio sistema (o pdfTeX não faz isso) ou interação com a linguagem de programação (brasileira) Lua (veja a Figura 3). A saída de cada um deles é o PDF, diretamente.

Em nossos sistemas, ao usarmos XeTeX com \TeX usamos o atalho Xe \TeX . Da mesma forma, LuaTeX com \TeX é simplificado por Lua \TeX .

Obviamente, nesse minicurso, estamos interessados apenas no formato \TeX . Também, por mera preferência, usaremos como interpretador o LuaTeX. Portanto, “compilaremos” nossos arquivos com `lualatex`.

Obtamos por usar `lualatex`, pois é o sucessor natural do `pdflatex`.²

**Figura 3 .** Poderosa linguagem de programação (clique na imagem para saber mais)

Saiba mais

De acordo com Joseph Wright, usamos a palavra “compilação” herdada da computação, mas, uma palavra mais adequada seria “composição tipográfica” (WRIGHT, 2021).

² Confira essa informação no site <http://www.luatex.org/>

²falaremos sobre editores para \TeX mais abaixo.

**Mudanças à vista ...**

Isso pode mudar algumas coisas para quem é acostumado a usar `pdflatex`, mas explicaremos as diferenças ao longo desse minicurso.

Geralmente chamamos esses “atalhos” de **compiladores**³.

Veja a Tabela 2 para comparação entre *compiladores*, *engine* e *format*.

³ Para mais detalhes, veja Pégourié-Gonnard (2013).

Compilador	Interpretador	Formato
<code>tex</code>	<code>T_EX</code>	plain T _E X
<code>pdftex</code>	<code>pdfT_EX</code>	plain T _E X
<code>xetex</code>	<code>X_ET_EX</code>	plain T _E X
<code>latex</code>	<code>pdfT_EX</code>	E _T X
<code>pdflatex</code>	<code>pdfT_EX</code>	E _T X
<code>xelatex</code>	<code>X_ET_EX</code>	E _T X
<code>lualatex</code>	<code>LuaT_EX</code>	E _T X
<code>texexec</code>	<code>pdfT_EX</code>	ConTeXt
<code>texexec --xtx</code>	<code>X_ET_EX</code>	ConTeXt
<code>context</code>	<code>LuaT_EX</code>	ConTeXt

Tabela 2 . Dando “nome aos bois”

Como esse texto é uma tentativa de introdução ao E_TX, não chegaremos nem perto do que gostaríamos de expor sobre essa simbiose entre uma linguagem de programação (Lua) integrada harmoniosamente com uma linguagem de marcação (E_TX). Basicamente, o código do T_EX foi reescrito em Lua (uma linguagem de programação com muita relevância internacional, desenvolvida por brasileiros, na PUC-RJ), formando assim o LuaT_EX. Houve, intencionalmente, a projeção para que esse *interpretador* fosse compatível com versões anteriores do pdfT_EX, tornando o LuaT_EX seu substituto natural, visto que é mais rápido e moderno.

Para encerrarmos essa subseção, é importante destacar a estabilidade do E_TX. Basicamente temos duas versões: uma antes de 1993, a saber E_TX 2.09; e, a de 1994 até HOJE, a saber E_TX 2_E. Isso é interessante, pois os códigos são praticamente preservados ao longo do tempo: você poderia rodar um código em E_TX de 20 anos atrás sem muitos problemas.

Todavia, quando falamos em E_TX, hoje, estamos nos referindo às funcionalidades trazidas na versão E_TX 2_E.

As atualizações são constantes, mas sem muitas mudanças estruturais. Todavia, está em desenvolvimento uma terceira versão do E_TX, a saber, E_TX3 (veja a Figura 5).

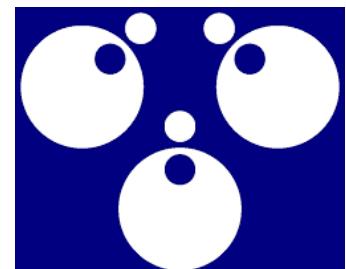


Figura 4 . Saiba mais sobre o LuaT_EX acessando o link: <http://www.luatex.org>



Figura 5 . Saiba mais sobre esse fantástico projeto no link: <https://www.latex-project.org>

1.4 Como colocar o \TeX em meu computador: as Distribuições

Para instalar localmente (em nosso computador) os *interpretadores*, precisamos das **distribuições**.

As principais são:

Distribuições	Sistema	Download/Instalação
MiK \TeX	Windows ou GNU/Linux ou Mac OS	https://miktex.org/download
\TeX Live	GNU/Linux ou Windows	https://www.tug.org/texlive/acquire-netinstall.html
Mac \TeX	Mac OS	https://www.tug.org/mactex/

A distribuição Mac \TeX contém todo o \TeX Live e adições específicas para Mac OS.

O \TeX Live completo precisa de, aproximadamente, 5GB de espaço em disco. Ele instala TODOS os pacotes disponíveis para \TeX . Como hoje em dia a capacidade de disco é relativamente grande, a instalação completa pode ser interessante, caso você não queira depender mais da internet para instalação de futuros pacotes.

Mas, se você quiser usar apenas o necessário e manter uma distribuição mímina em seu computador, instalando futuros pacotes à medida que for precisando deles, talvez a distribuição MiK \TeX seja a mais adequada.



Atenção!

O processo de instalação depende do sistema operacional e não será tratado nesse texto, visto que usaremos distribuições ONLINE dos interpretadores de \TeX .

Saiba mais

Para GNU/Linux existem outras opções de instalação que economizam espaço. Tudo dependerá da necessidade de cada um:
`texlive-latex-base`
`texlive-latex-recommended`
`texlive-pictures`
`texlive-fonts-recommended`
`texlive`
`texlive-plain-generic`
`texlive-latex-extra`
 Para mais informações veja essa discussão no [TeX SE](#)

2 Sim ...mas como eu uso?

Certo ...já temos uma distribuição \TeX , e agora? Basicamente, precisamos:

- *escrever*, usando o formato \TeX , o que desejamos num arquivo de texto com extensão `.tex`;
- *compilar*, ou seja, compor tipograficamente o arquivo `.tex`, produzindo um arquivo `.pdf`; e, nesse ponto, usaremos o compilador `lualatex`;
- *visualizar* o arquivo `.pdf`; nesse ponto, o visualizador de PDF é de escolha pessoal.

É uma boa prática salvar os arquivos `.tex` com nomes **sem acentuação** ou **caracteres especiais** do teclado (%,\$,* , etc.). E, se o nome do arquivo for composto por mais de uma palavra, é aconselhável também **não deixar espaço entre elas**.

Por exemplo, suponha que você esteja escrevendo um artigo sobre *números complexos*. Seu arquivo principal não deve ser nomeado assim:

Artigo! Números Complexos.tex

Você deve retirar o *ponto de exclamação* e o *acento agudo*, bem como retirar os espaços entre as palavras. Nesse último ponto, pode-se usar o *camelCase*, ou *snake_case* ou separarar as palavras-com-traço. São aceitáveis qualquer das seguintes possibilidades (note que há uma preferência por letras minúsculas):

artigoNumerosComplexos.tex
 artigo-numeros-complexos.tex
 artigo_numeros_complexos.tex
 artigo_numeros-complexos.tex

Particularmente, prefiro essa última opção: onde se mistura o *snake_case* com os traços. Tento manter algum padrão antes do *underline* e o nome do documento que estou escrevendo, separo por traços, caso seja necessário. Essa abordagem pode ser interessante se você estiver trabalhando com um arquivo principal (onde geralmente ocorre a compilação final) e muitos outros arquivos que serão incluídos no principal.

É uma forma de escrever ...Há quem prefira escrever tudo em um único arquivo .tex!

Falaremos sobre manipulação de vários arquivos mais à frente, porém, apenas para exemplificar a ideia, suponha que esse meu artigo (*artigo_numeros-complexos.tex*) sobre números complexos seja composto de cinco seções: introdução histórica; formas de representar um número complexo; raízes de um número complexo; o Teorema Fundamental da Álgebra; e, conclusões. Então, desejamos escrever essas seções em arquivos separados e “incluir-los” no arquivo principal, paulatinamente, por meio de compilações SOMENTE no arquivo principal.



Fica a dica!

Muitos nomeiam esses “arquivos principais”, ou seja, onde ocorre a compilação, de *main.tex* (ou *master.tex*), que significa, em inglês, “principal”, “importante”, etc. (“senhor”, “dominador”, etc., para *master*).

Para esse minicurso, à jato, começaremos com um arquivo principal, nomeado por “*main.tex*”, no qual escreveremos o conteúdo desejado inserindo as partes que o compõem, aos poucos.

Saiba mais

Uma forma de nomear esses arquivos seria, respectivamente:
01_intro-historica.tex;
02Representacoes.tex;
03_raizes.tex;
04_teo-fundamental.tex;
05_conclusao.tex.

Notem que deixei a numeração no início, separando-a com *underline* e nomeei os arquivos de maneira que lembre-me a seção onde me encontro.

2.1 Como fazemos a “compilação”?

Bom ...ainda aprenderemos como escrever em \TeX , não se preocupem, mas, antes, vamos aprender como compomos tipograficamente o texto.

Já vimos que precisamos escrever o conteúdo do texto em um arquivo de extensão `.tex` e compilarmos com `lualatex` ...Mas, como fazemos isso?

Ora, a escrita do texto pode ser feita em qualquer **editor de texto** de sua preferência. Pode-se usar um “bloco de notas” (para usuários de Windows), ou o “evince” (para usuários de GNU/Linux), por exemplo. Apenas deve-se lembrar em salvar o arquivo com extensão `.tex`.

Também é aconselhável manter o arquivo `main.tex` em algum diretório (pasta) nomeado adequadamente. Isso se deve ao fato de que, ao compilarmos, arquivos secundários são gerados (`.aux`, `.log`, etc.), o que pode gerar certa “bagunça”.

Então, suponha que você tenha escrito um belo texto, cheio de equações matemáticas, num arquivo denominado `main.tex`, salvo em um diretório por nome “`artigo_tcc/`”. Feito isso, abra o **terminal** (no Windows seria o “Prompt de Comando”) dentro do diretório “`artigo_tcc/`” e digite o seguinte comando:



Usando o terminal para compilar

```
1 lualatex main.tex
```

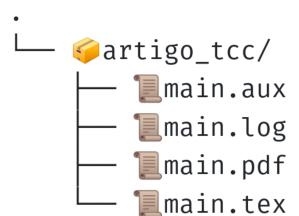


Figura 6 . Arquivos gerados numa compilação simples

Pelo menos três arquivos serão produzidos: um arquivo em `.pdf`, que é a saída desejada; um arquivo `.aux`, essencial em referências cruzadas, por exemplo; e, um arquivo `.log`, o qual é um registro detalhado de tudo o que ocorreu na compilação, inclusive possíveis erros. Pode aparecer mais arquivos durante a compilação! Tudo dependerá de quão complexo é seu texto (se possui sumário; index; lista de tabela; lista de figura; glossário; etc). A Tabela 3 nos mostra alguns desses arquivos:

Tabela 3 . Alguns dos possíveis arquivos gerados durante a compilação

Extensão	Descrição
<code>.log</code>	registro detalhado sobre a compilação, inclusive erros
<code>.aux</code>	registra processos intermediários, como referências cruzadas
<code>.toc</code>	serve para produção do índice
<code>.lof</code>	serve para produção da lista de figuras
<code>.lot</code>	serve para produção da lista de tabelas

Não tenha medo do terminal! Ele é seu amigo! Trabalhar usando “linha de comando”, pelo terminal, é uma maneira de você comunicar-se com o computador de maneira rápida, direta e livre de distrações. O comando `lualatex main.tex` simplesmente diz: “Olha ...componha tipograficamente meu texto que está no arquivo `main.tex`, usando o interpretador \TeX , no formato \TeX ”.

Mas, se você quiser mais funcionalidades na digitação (autocomplete, identação automática, etc.) que um “bloco de texto” não oferece; bem como compilar sem usar o terminal, você precisará de um **editor de texto** especializado ou uma IDE (*Integrated Development Environment*).

2.2 Editores para \LaTeX

2.2.1 O \LaTeX é outro Word?

Um detalhe que precisamos ter em mente: a natureza dos editores para \LaTeX é geralmente diferente dos editores de texto como *Microsoft Word* ou *LibreOffice Writer*. Esses últimos editores são denominados WYSIWYG (acrônimo para a frase **What You See Is What You Get**). Em uma tradução literal livre significa “O que você vê é o que você tem”, ou seja, o que você vê na tela de seu editor, é o que aparecerá na impressão. No \LaTeX não é bem assim ...Você vê comandos misturados com texto normal, porém o resultado sairá diferente daquilo que estará vendo em seu editor! Ficará mais bonito o resultado, não se preocupe. =)

Poderíamos listar argumentos sobre a superioridade do \LaTeX , comparando-o ao *Word* (por exemplo), mas isso não será feito. Cada um tem suas peculiaridades: vantajosas ou não. Tudo dependerá da finalidade do uso! Particularmente, tenho encontrado no \LaTeX um ambiente próprio para escrita matemática de forma rápida, bonita e gratuita! Textos longos, cheios de capítulos, ou documentos personalizados — como esse texto, exigiria um esforço e tempo tais, que nem passa por minha mente escrevê-los no *Word* (você já tentou criar um simples sumário no *Word*? Ou uma simples equação como esta: $\sum_{n=0}^{\infty} x^n = \frac{1}{1-x}$, em menos de 10 segundos?)

Com isso em mente, é importante a aquisição de uma linguagem básica, mas, antes disso, vamos falar um pouco sobre os editores para \LaTeX .

2.2.2 Indicações de editores para \LaTeX

As possibilidades de editores são realmente vastas. Portanto, comparar as funcionalidades dos principais editores é algo fundamental. Você pode olhar uma comparação entre eles nesse link:

https://en.wikipedia.org/wiki/Comparison_of_TeX_editors

Podemos usar editores mais **genéricos** ao \LaTeX ou editores **dedicados**.

No que se refere aos editores mais genéricos, ou seja, que podem ser usados para outras linguagens de marcação (ou programação), podemos usar *plugins* para usarmos o \LaTeX essa realidade muda. Veja a Tabela 4 para algumas sugestões.

Agora, no que se refere aos editores dedicados, a Tabela 5 exibe algumas possibilidades.

Obviamente, para o iniciante, é indicado um editor dedicado ao \LaTeX .

Saiba mais

“WYSIWYG ...O termo é usado para classificar ferramentas de edição e desenvolvimento que permitem visualizar, em tempo real, exatamente aquilo que será publicado ou impresso” ([TECMUNDO](#), acesso em 03/02/2022).

Se você quiser saber como pronunciar corretamente essa sigla, acesse: [How to pronounce WYSIWYG](#).

Tabela 4 . Editores Genéricos

Editor	Plugin
GNU Emacs	AUC \TeX
Atom	atom-latex
Neovim	VimTeX
VSCode	\TeX Workshop

Tabela 5 . Alguns Editores Dedicados ao \TeX

Editor	Algumas características
TeXstudio	Destaca-se por ser multiplataforma e com muitos recursos.
TeXnicCenter	É voltado para usuários de Windows. Integra-se muito naturalmente ao sistema (e ainda indica o Sumatra para visualização do PDF).
LyX	Esse é um editor diferente dos citados anteriormente; pois renderiza as equações e estruturas do texto de forma quase imediata. É o mais próximo, para \TeX , de editores WYSIWYG.

Particularmente, uso o VSCode com o *plugin* do \TeX Workshop. Esse *plugin* faz o processo de compilação de uma forma um pouco diferente: ele usa uma ferramenta chamada **latexmk** que nada mais é do que um *script* que automatiza muitos processos. Por exemplo, quando o documento é complexo; com referências cruzadas; sumário; glossário; lista de tabelas; lista de figuras; etc. é necessário mais de um processo de compilação. Esse *plugin* faz isso tudo parecer mágica. Ele vem, por padrão, configurado para pdflatex, mas pode ser modificado o interpretador de maneira bem simples.

Uma ferramenta parecida com o **latexmk**, mas que precisa de diretivas (você precisa “dizer” para ele os passos que deseja executar) é o **arara**. Ele foi desenvolvido por um brasileiro (Paulo Cereda) e possui ferramentas de limpeza de arquivos auxiliares; rapidez na segunda compilação; produção de arquivo *draft* (um rascunho que não gera o conteúdo pdf, mas apenas verifica a sintaxe do documento – o que torna as coisas muito mais rápidas); etc. Falaremos sobre ele no Apêndice ??, pois vale muito a pena conhecer essa ferramenta!



3 Overleaf: um editor *online* notável

Como nosso minicurso é *online* e possui tão curto tempo, não seria possível um auxílio na instalação de uma distribuição \TeX , editor dedicado, nem tampouco um visualizador de PDF de cada um de vocês, localmente.

Usaremos, então, uma plataforma que reúne tudo isso de forma *online* e gratuita (para aquilo que desejamos): **Overleaf**. Vamos conhecer alguns recursos disponíveis do **Overleaf**, antes de iniciarmos a aquisição da linguagem \TeX .

O primeiro passo é acessar o *link* e fazer um registro (*Register*) de uma conta (é possível fazer o *login* com uma conta Google), logando na plataforma *online* (Veja Figura 7).

Ao entrar no **Overleaf**, aparecerá uma tela como é mostrado na Figura 8.

Saiba mais

Existem outros editores online que poderíamos usar: **Papeeria** ou **Authorea**, por exemplo.

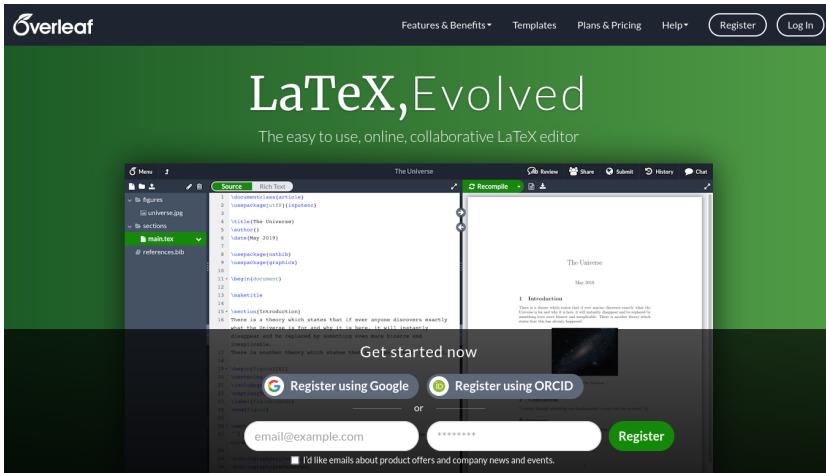


Figura 7. <https://www.overleaf.com/>

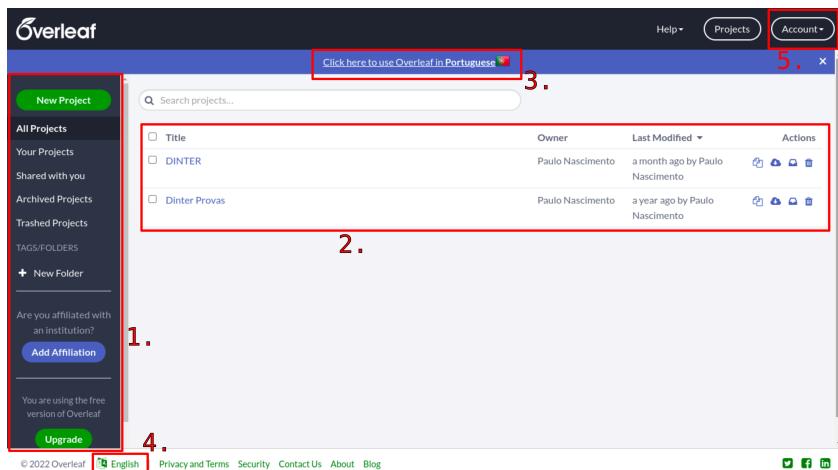


Figura 8. Visão inicial do Overleaf

Vamos explicar o que significa cada nicho destacado:

1. Aqui é onde iniciaremos um **Novo Projeto** (*New project*), que poderá ser em Branco; ou de algum modelo que o próprio **Overleaf** disponibiliza; ou do GitHub; etc.;
2. Nesse nicho aparecerão todos os seus projetos (no meu caso, tenho 2 em andamento);
3. Você pode modificar o idioma por aqui;
4. Também é possível fazermos a modificação do idioma por aqui;
5. Você pode sair do **Overleaf** clicando nesse botão e, em seguida, Sair (*LogOut*)

Escolha o item “3” ou “4” para modificar o idioma para Portugues. Em seguida, abra um projeto em branco, seguindo o caminho:

Novo Projeto → Projeto em Branco

Aparecerá uma janela para que você coloque o título do projeto. Escreva:

minicurso-EMAT_LaTeX

Entraremos naquilo que vamos denominar Área de Trabalho do Overleaf (algo como a Figura 9).

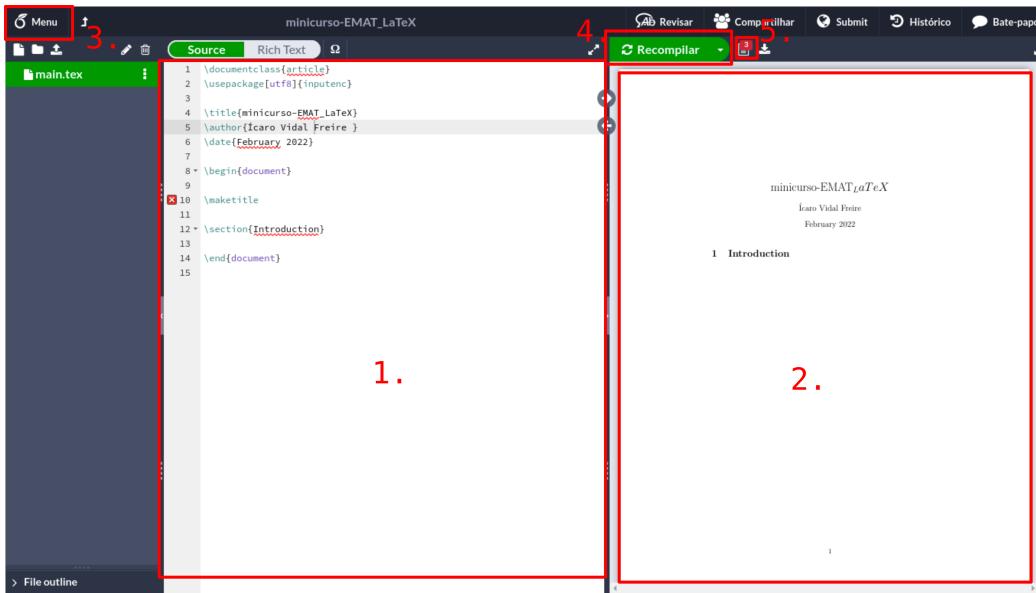


Figura 9 . Visão da Área de Trabalho dos Projetos, no Overleaf

Vamos modificar o compilador (lembrem-se que usaremos o `lualatex`; não o `pdflatex`, que é o padrão do Overleaf), mas, antes, vamos conhecer um pouco essa área de trabalho:

1. Digitamos os códigos nesse espaço! Aqui é onde escreveremos a linguagem do L^ET_X. Notem que o Overleaf já usou o seu nome e o nome do projeto para preencher algumas coisas nessa linguagem;
2. A saída do pdf é mostrada aqui. A renderização é automática, o que facilita muito o aprendizado para iniciantes;
3. Nesse espaço modificamos muita coisa no Overleaf, especificamente, modificamos algumas configurações, inclusive o compilador. Também podemos fazer o *download* do PDF ou do Código por aqui.
4. No botão *Recompile* existem muitas opções para renderizar o documento.
5. É aqui que você pode configurar para compilação automática; ou produzir um documento *draft* (rascunho); ou para checagem das sintaxes;

6. Por fim, aqui mostra mensagens de erros ou alertas. Inclusive, há três mensagens de erros por lá! Veremos que a mensagem, na realidade, resume-se a um único problema: Ao aproveitar o título de nosso projeto e escrevê-lo no título do documento, o Overleaf usou um **caractere especial** no modo Texto, mas que é exclusivamente reservado ao Modo Matemático 🤯.



Atenção!

Em Menu (ver item 3 da Figura 9), altere o compilador modificando na seção *Settings*, item *Compiler*, de pdfLaTeX para LuaLaTeX.

O nosso fluxo de trabalho será:

- **escrever** o texto e os códigos na área adequada;
- **compilar** (compor tipograficamente) o documento;
- **depurar** possíveis erros;
- **admirar** o resultado 😊.

Algumas outras peculiaridades do Overleaf veremos quando estivermos praticando a linguagem do L^AT_EX nos exercícios! Então, agora, resta-nos aprender os rudimentos dessa linguagem.

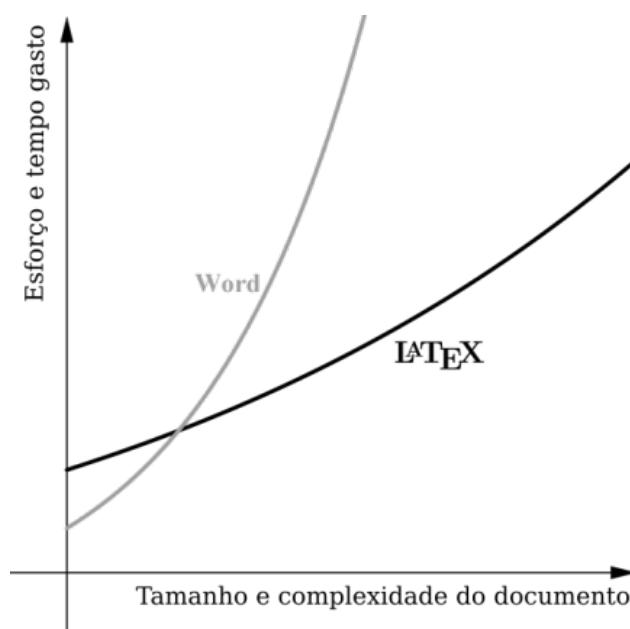
Aprendendo a Linguagem LaTeX

4 Uma Linguagem

A aquisição de uma linguagem não é algo imediato (para a maioria das pessoas); e, o \LaTeX é uma linguagem, conhecida como *linguagem de marcação*. Isso implica que há uma estrutura e simbologia características, ou seja, é necessário um investimento em tempo e esforço para uma certa fluidez na comunicação ou escrita.

Vimos na Subseção 2.2.1 que o \LaTeX é uma linguagem que envolve *códigos* e *texto*. Sua estruturação é diferente de sistemas WYSIWYG (como o Word) e, à primeira vista, demanda um “esforço técnico” maior no início do aprendizado, mesmo para documentos simples. Mas, à medida que a complexidade do documento aumenta, o esforço empregado ao usar o \LaTeX é menor, quando comparado à sistemas WYSIWYG, para produzir o mesmo documento. A Figura 10 mostra um vislumbre dessa ideia, embora seja apenas hipotética.

Figura 10 . Complexidade do Documento vs Esforço empregado

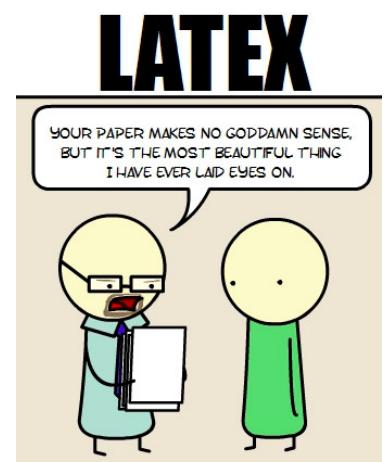


Fonte: <https://latextcfp.wordpress.com/>

Além disso, é notável a beleza da composição tipográfica final, em especial (mas não necessariamente) quando o documento envolve equações matemáticas. Como vimos, essa beleza é fruto das mais sofisticadas técnicas empregadas nos

Tabela 6 . Sumário da PART II

| **Seção 4 .** Uma Linguagem



Seu artigo não faz nenhum sentido para mim, mas é a coisa mais linda que eu já vi! (Tradução livre e suavizada 😊)

Fonte: [The magic of LaTeX](#).

algorítimos que produzem a saída 0 ou 1, representando, respectivamente, “sem tinta” ou “com tinta”.

4.1 Um pouco sobre a estruturação do LaTeX

Antes de aprendermos os símbolos pertinentes dessa linguagem, é interessante conhecermos um pouco de sua estruturação.

4.1.1 Comandos

Um comando no \TeX sempre começa com uma barra invertida (“\”). Tal comando pode estar inserido em modo texto ou em modo matemático.⁴ Por exemplo, quando escrevemos \LaTeX , com o “L”, “T” e “X” maiúsculos, o resultado depois da compilação é: \TeX . Mas, se digitarmos \Latex , uma mensagem de erro será mostrada, visto que esse comando não vem por padrão definido.

Tal exemplo já nos mostra uma outra característica da estruturação da linguagem que estamos conhecendo: ela é *case sensitive*, ou seja, é sensível à letras maiúsculas e minúsculas. O comando \LaTeX é diferente do comando \ latex , que é diferente do comando \Latex .⁵

⁴ Existe toda uma sintaxe para o modo matemático, que vamos deixar para abordar em outra seção.

⁵ $\text{\LaTeX} \neq \text{\Latex} \neq \text{\ latex}$



Atenção!

Inclusive, essa é uma das grandes fontes de erro de quem está começando no \TeX : a digitação de um comando de forma errada é muito comum!

Também é importante ressaltar que comandos podem vir acompanhados de **argumentos obrigatórios**, que colocamos entre chaves (“{}”); ou de **argumentos opcionais**, que colacamos entre colchetes (“[]”).

Por exemplo, o comando \documentclass define a classe do documento. O argumento da classe é obrigatório, ou seja, precisamos “indicar” ao \TeX se estamos usando um artigo (*article*); um livro (*book*); um relatório (*report*); uma apresentação (*beamer*); etc. Além disso, existem vários argumentos opcionais, dentre os quais descato: o tamanho da fonte, que por padrão é 10pt⁶; o formato do papel, por exemplo, A4; e a opção de *layout* para estruturar as margens à impressão em frente e verso: *twoside*. Veja como ficaria o comando para a classe *article*; com tamanho da fonte de 12pt; definida para um formato de paper A4; e, com a opção de imprimir em frente e verso:⁷

```
\documentclass[12pt, a4paper, twoside]{article}
```

4.1.2 Classe do Documento

Notem que uma classe possui características intrínsecas: um artigo não possui capítulos, mas seções; o que é diferente de um livro, que possui essas duas

⁶ são três opções possíveis nas classes *standard*: 10pt, 11pt e 12pt.

⁷ Note que o comando opcional *twoside* não “diz” para a sua impressora imprimir em “frente e verso”. Apenas configura as margens das páginas para essa opção.

estruturas, por exemplo. Existem classes que já vem, por padrão, nas distribuições TeX que conhecemos e inúmeras outras implementadas pela comunidade de usuários do \TeX .

A Tabela 7, mostra algumas dessas classes nativas (também chamadas de padrão ou *standard*).



Atenção!

Por conta da limitação de nosso tempo, vamos tangenciar alguns aspectos da linguagem \TeX , usando apenas a classe *article*.

Você já deve ter imaginado que existe uma infinidade de classes para o \TeX , além das nativas. De fato, há uma comunidade ativa que contribui com seus talentos e disponibilizam, em sua grande maioria, de forma gratuita muitas funcionalidades/ferramentas que facilitam a vida de usuários do \TeX . Dentre essas funcionalidades/ferramentas, estão as classes. Mas, existem **pacotes**; aplicativos em java (como o arara); etc. Geralmente, as classes modificam a estrutura geral de um documento. Como vimos, um artigo possui uma estrutura diferente de um livro, por exemplo. O que define essas estruturas são as classes.



Atenção!

Cada classe possui sua sintaxe específica para alguns comandos, porém alguns outros comandos são gerais e servem para a maior parte das classes. Estudaremos apenas esses últimos. Caso você queira trabalhar com uma classe diferente da “nativa”, deve verificar o manual para os detalhes.

Dentre as classes que não são nativas ao \TeX , gostaria de destacar:

abnTeX2 Se você pensa em escrever seu Trabalho de Conslusão de Curso, usando \TeX , você precisa ler a documentação e conhecer bem essa classe. Ela, basicamente, fornece um modelo canônico (*template*) para várias estruturas de cunho científico, dentro do padrão da ABNT (Associação Brasileira de Normas Técnicas). Você não precisará se preocupar com as configurações de todas aquelas normas em seu texto! Conhecendo os comandos da classe, você só se preocupará com o conteúdo, não com a estrutura. Inclusive, estou desenvolvendo uma classe para monografia do curso de Licenciatura em Matemática da UFRB, baseada na classe abnTeX2. Ela está disponível no link: [tccUFRB.cls](#)

beamer Esta é uma classe para apresentações em *slides* (chamado também de *frame*, pelos usuários). A qualidade das apresentações é surpreendente. Principalmente quando a apresentação envolve fórmulas matemáticas (não distorce

Tabela 7 . Algumas das classes *standard* do \TeX

Classe	Finalidade
<i>article</i>	artigo
<i>book</i>	livros
<i>report</i>	relatórios
<i>letter</i>	carta

A extensão do arquivo para uma classe é .cls

Saiba mais

É bem engraçada, e realista, a descrição do nome abnTeX:
ABsurd Norms for TeX.
 Por que será, né? 😊

como acontece por aí ...)! Existem uma infinidade de *templates* (oficiais ou não) disponíveis que você, certamente, encontrará um que atenda às suas necessidades. O manual do beamer pode ser visto no *link*: [The BEAMER class](#). Um *template* de tema claro, extra oficial, que recomendo para beamer é o [Metropolis](#); e, um escuro (*dark*): [The Nord Beamer Theme](#).

Um pequeno manual, em espanhol, que considero muito prático e eficiente (principalmente se você não quiser ler as 234 p do manual oficial) é: [Presentaciones en L^AT_EX con Beamer](#).

exam Essa é uma classe para confecção de listas de atividade. Possui um ambiente para soluções; comandos para pontuação nas questões; cria cabeçalhos e rodapés estilizados; etc. Se você é professor e usa fórmulas em suas listas de atividades, recomendo fortemente ler a documentação dessa classe exam.cls. Você encontrará o manual nesse *link*: [Using the exam document class](#).

Baseada nessa classe, desenvolvi (e estou no processo de atualização) a classe [ativmatUFRB.cls](#). Ela é uma classe não oficial para listas de atividades do curso de Licenciatura em Matemática, da UFRB. Possui um cabeçalho estilizado com o logo da universidade, bem como comandos e informações que considero úteis num documento desse tipo.

4.1.3 Mais sobre Comandos

Uma pergunta que talvez possa surgir nesse momento: *se cada comando começa com a "barra invertida", qual o comando para representar a própria barra invertida?* 😊 É natural pensarmos em “\\”, mas a pergunta surgiria novamente: *como representar “\\”?* 🤔

E seguiria uma sequência infinita de perguntas, não?! 🤔

Bom, para que essas questões possam ser resolvidas, você deve ter em mente que alguns símbolos são reservados, ou seja, há um conjunto de definições internas ao próprio L^AT_EX. Inclusive, símbolos usados em estruturas internas de comandos. É o caso de “\”: ele é usado para iniciar qualquer comando no L^AT_EX. Se você quiser representá-lo textualmente, pode usar \textbackslash.

A Tabela 8 mostra alguns desses caracteres especiais.

Se você quiser exibir qualquer um deles, (excetuando a *barra invertida*, claro) basta colocar a barra invertida antes dos símbolos! Por exemplo, se deseja exibir “10%”, deve digitar: 10\%.

Ainda falando sobre comandos ... se você for utilizar algum comando interno, ou mesmo algum criado por você (sim isso é possível, mas não veremos nesse microcurso 😞), deve ter um cuidado especial com o espaçamento de alguma palavra depois do referido comando. Por exemplo, se formos escrever

Veremos que, no modo Matemático, esse mesmo comando pode ser representado simplesmente por \backslash.

O L^AT_EX é sensacional!

Tabela 8 . Caracteres Especiais no \TeX

Caractere	Descrição
\	inicia um comando
%	para comentar código
~	evita separação de palavras
{ }	delimita códigos
&	separa colunas numa tabela
#	exibe n° de parâmetros num comando
\$	usado no modo matemático
^	usado para potências
_	usado para subíndices

não devemos fazer “O \TeX é sensacional!”, visto que a saída produzida por esse último comando é:

O \TeX é sensacional!

Devemos, portanto, colocar “{}” ou “\” ao final do comando. Isso faz com que haja um pequeno espaço entre ele e a próxima palavra. Em resumo, você deve fazer alguma das seguintes maneiras:

- (i) O \LaTeX{} é sensacional!
- (ii) O \LaTeX\ é sensacional!

Gostaria de destacar o uso do comando % para comentários de código! Use-o sempre para lembrar-se de algo ou para organização. Ao longo desse texto, esse comando será usado constantemente.

4.1.4 Ambientes

Um agrupamento de comandos peculiar são os ambientes (*environments*). Pode ser uma tabela; um ambiente matemático; um ambiente para texto; etc. Sempre começará com algum `\begin{algum-nome}` e, **obrigatoriamente**, terminar com seu respectivo `\end{algum-nome}`.



Atenção!

Pode haver a necessidade de ambientes aninhados! Por isso, tome cuidado para não esquecer de “fechar” um ambiente (com `\end`), inclusive na ordem correta! Uma dica é você dar pelo menos dois espaços no início do ambiente interno ao principal.

Por exemplo, suponha que você queira escrever o ambiente nome-2, dentro do ambiente nome-1. Então, poderia organizar assim:

Organizando os ambientes

```
1 \begin{nome-1}
2   \begin{nome-2}
3     Escreva aqui ...
4   \end{nome-2}
5 \end{nome-1}
```

Um ambiente notável é o document. Sem ele não podemos escrever nossos documentos. Escrevemos logo depois do comando \documentclass. O código abaixo mostra uma estrutura mínima de um texto em \TeX . Note que usei o comando % para delimitar partes importantes da estrutura no \TeX . Isso é opcional; entretanto, deixa seu código mais arrumado para futuras consultas. 😊

Estrutura mínima para escrever um texto

```
1 % classe do documento =====
2 \documentclass{article}
3
4 % início do documento =====
5 \begin{document}
6 %
7
8 Olá! Esse é um pequeno texto em \LaTeX.
9
10 % fim do documento -----
11 \end{document}
```

4.1.5 Pacotes

Obviamente, você não precisa “inventar a roda” para desenvolver implementações no \TeX . Muito do que precisamos numa escrita de um artigo que envolva expressões matemáticas, já foi pensado e desenvolvido por alguém. Essas pessoas, empenharam tempo e dedicação para usar os comandos primitivos do $\text{T}\bar{\text{E}}\text{X}$ ou \TeX e escreverem seus pacotes.

Por exemplo, suponha que desejamos escrever expressões matemáticas que envolvam Matrizes; Integrais; símbolos da Teoria dos Conjuntos; Álgebra; etc. Não seria conveniente que a escrita desses comandos seja não conflituosa; integrativa e mantenha uma certa harmonia entre eles? Bom, a $\mathcal{AM}\mathcal{S}$ - \TeX desenvolveu um conjunto de pacotes essenciais para isso.

Saiba mais

A sigla AmS vem de *American Mathematical Society*, ou seja, “Sociedade Americana de Matemática”.

**Atenção!**

Nessa pseudo-introdução ao \LaTeX , os ambientes, fontes e expressões matemáticas serão compostas por meio desses pacotes.

Gostaria de destacar, dentre os pacotes da $\mathcal{AM}\mathcal{S}$ - \LaTeX , os seguintes:

amsmath. É o principal pacote para expressões e ambientes para escrita matemática. Nele podemos enumerar equações; construir matrizes; funções; etc. Realmente é bem vasto e essencial para escrever qualquer texto sério em Matemática. Você pode saber mais sobre ele no link: <https://ctan.org/pkg/amsmath>.

amssymb. Esse pacote carrega outro muito importante, amsfonts; e, juntos, disponibilizam um conjunto de símbolos e fontes para escrita em Matemática. Dá para escrever \subset , \emptyset , \triangle , \mathbb{N} , \mathcal{F} , etc. Você pode saber mais sobre ele no link: <https://ctan.org/pkg/amsfonts>

amsthm. Esse pacote fornece comandos para formatar o estilo, numeração, etc., das Definições, Lemas, Teoremas; Observações, etc. Você pode saber mais sobre ele no link: <https://ctan.org/pkg/amsthm>

amscd. Pacote com uma linguagem muito simples para escrever diagramas comutativos.⁸ Você pode saber mais sobre ele no link: <https://ctan.org/pkg/amscd>

Falaremos mais sobre esses pacotes de axílio matemático mais à frente; porém, é importante destacar que ao longo do tempo, outros pacotes surgiram para complementá-los. É o caso, por exemplo, do **mathtool** que, além de carregar o amsmath, ainda disponibiliza correções e novas implementações para o mesmo. Também é notável o pacote **tikz-cd** para os diagramas comutativos: simplifica e expande (em muito) o pacote da amscd.

Para carregar um pacote ao nosso documento, usamos o comando (Note que alguns pacotes podem oferecer opções para configurações.):

```
\usepackage[opções]{nome-do-pacote}
```

Além disso, também podemos agrupar pacotes. Por exemplo, os pacotes citados, da $\mathcal{AM}\mathcal{S}$ - \LaTeX , podem ser agrupados assim:

```
\usepackage{amsmath, amssymb, amsthm}
```

Você deve imaginar a quantidade exorbitante que existe de pacotes, disponíveis para as mais variadas situações no \LaTeX : desde cifras, tablaturas ou partituras (**TeX**

⁸ Um exemplo de diagrama comutativo:

$$\begin{array}{ccc} H & \xleftarrow{f} & H \\ \downarrow \psi & & \downarrow \psi \\ H_1 & \xleftarrow{\psi^*(f)} & H_1 \end{array}$$

for Musicians); songbook (**leadsheets**); xadrez (**TeXmate**); sudoku (**sudokubundle**); labirintos (**labyrinth**); etc. (Veja a página sobre **games**); até construção de gráficos (**pgfplots**); objetos em 3D (**tikz**); etc. (<https://texample.net>).

Você pode ver a lista de pacotes disponíveis no *link*:

<https://ctan.org/pkg>

Certamente, não precisamos de todos eles num texto! Tudo dependerá da necessidade e, com o tempo, saberemos selecionar os mais adequados aos nossos objetivos. Todavia, alguns são notáveis, quer por sua simplicidade, quer por suas funcionalidades. Falaremos sobre eles ao longo desse texto, mas gostaria de descrever parte deles:

fontspec É um pacote essencial para quem usa Lua \TeX ou X \TeX . Com ele podemos selecionar as fontes instaladas em nosso sistema operacional, bem como oferece uma codificação adequada à saída das mesmas.⁹

babel Ele gerencia as regras tipográficas de mais de 250 idiomas! Quando selecionamos a opção para nosso idioma, esse pacote faz, quando necessário, a hifenação correta das palavras, bem como traduz as estruturas tipográficas características do texto em inglês, na maioria das classes disponíveis. Carregamos esse pacote assim:

```
\usepackage[brazil]{babel}
```

microtype Pacote bem sutil que ativa protusão de caracteres e expansão da fonte, além do ajuste entre as palavras, melhorando o espaçamento entre as mesmas. Em outras palavras, esse pacote organiza as palavras nos parágrafos de forma harmoniosa, evitando desalinhamento de texto.

geometry Fenomenal pacote para ajuste de margens ou papel! Apenas como exemplo, para ajustar as margens no padão da ABNT, de um papel A4, podemos fazer:

```
\usepackage[a4paper,top=3cm,left=3cm,bottom=2cm,right=2cm]{geometry}
```

booktabs Estilo muito elegante para formatação de tabelas.

graphicx Essencial para inserção das figuras no texto. Nativamente suporta as extensões **pdf**, **png** e **jpg**. Também podemos configurar para que o \TeX insira apenas as figuras contidas numa pasta, de nome, **figs**, por exemplo. O carregamento, do pacote é feito como: `\usepackage{graphicx}`; e a configuração para inserir figuras do diretório **/figs** como:

```
\graphicxpath{./figs/}}
```

Saiba mais

O *The Comprehensive TeX Archive Network* (CTAN) é um verdadeiro repositório para todo tipo de material em \TeX . Atualmente possui 6220 pacotes!

⁹ Se fôssemos usar o $\text{pdf}\text{\TeX}$, precisaríamos do pacote **fontenc**, com codificação T1; além do empacotamento das fontes que desejássemos usar!

Saiba mais

Os caras são realmente criativos! A palavra **babel** significa “confundir” (a tradução vem do hebraico). Há um relato na Bíblia, no Capítulo 11 do livro de Gênesis, sobre a história de uma torre que seria construída pelos seres humanos, quando todos falavam a mesma língua: a chamada Torre de Babel. Se observarmos o contexto do livro, os seres humanos haviam se rebelado contra Deus e, juntos, construiriam essa torre que “tornaria célebre” o nome da raça humana corrompida. Deus, então, dispersa esses homens “confundindo” (**babel**) suas linguagens! Aí o pessoal do pacote denomina-o por **babel**, para retornar à linguagem materna de cada indivíduo.



biblatex-abnt É um pacote que usa o estilo BibLaTeX (estilo amplo e moderno para compor as bibliografias) para produzir Bibliografia, compatível com as normas da ABNT. Usamos o comando `\usepackage[style=abnt, justify]{biblatex}` para carregarmos o pacote com as normas da ABNT. Mas, escrevemos a bibliografia (com comando específicos dessa classe, portanto é importan- tíssimo ler o manual da mesma) num arquivo separado, de extensão `.bib`, adicionando-o com o comando `\addbibresource{referencias.bib}` (por nome *referencias*, por exemplo) e exibindo-a com `\printbibliography` (no arquivo principal).

Agora que comentamos sobre alguns pacotes, vamos organizá-los em nosso arquivo `main.tex`. Onde colocamos esse comando?

O espaço entre o comando `\documentclass[]{} e \begin{document}` é denominado de **préâmbulo**. Nele colocamos as configurações prévias que desejamos fazer em nosso texto (escolha de fontes; informações sobre título, autor e data; comandos criados por você; etc.) e os pacotes que usaremos.¹⁰

Um arquivo básico seria:



main.tex (estilizado)

```

1 % classe do documento =====
2 \documentclass[12pt]{article}
3
4 % préâmbulo =====
5 % pacotes carregados -----
6 \usepackage{fontspec}
7 \usepackage[brazil]{babel}
8
9 \usepackage[a5paper]{geometry}
10
11 \usepackage{mathtools, amsthm, amssymb}
12 \usepackage{booktabs}
13
14 \usepackage{graphicx}
15 \graphicspath{{./fig/}}
16
17 \usepackage{microtype}
18
19 \usepackage[style=abnt, justify]{biblatex}
20 \addbibresource{referencias.bib}
21 % configurações -----
22
23 % início do documento =====
24 \begin{document}
25 %
26
27 Olá! Esse é um pequeno texto em \LaTeX.
28
29 % fim do documento -----
30 \end{document}

```

¹⁰ Dependendo da distribuição do TeX que você possua em seu computador (caso não use o Overleaf, nem todos os pacotes estão instalados). Entretanto, ao carregá-los, geralmente, a instalação é automática.

Não coloquei as configurações ainda, pois faremos isso ao longo do texto. Da mesma forma, ainda não foi colocado o comando para exibir a bibliografia, pois falaremos em uma seção específica para ela. Note, também, que selecionei o tamanho de papel A5 para o texto, deixando as margens no padrão nativo do \LaTeX (fica melhor para visualização na tela, por estarmos num minicurso online).

4.2 Arrumando as coisas

Bom, a proposta é deixar o arquivo principal, `main.tex`, o mais “limpo” possível.

Com efeito, se escrevermos todo o texto, mais as configurações e mais os pacotes; tudo isso no mesmo arquivo, é possível que haja uma certa desorganização. Ou, por uma desventura da vida, você apagar de forma descuidada o arquivo, perdendo tudo o que escreveu!

Para que situações como a descrita sejam minimizadas, o ideal (no meu entendimento) é separarmos o conteúdo e incluir essas partes no arquivo principal. Fazemos essa inclusão por meio do comando:

```
\input{nome-do- arquivo.tex}
```

Assim, vamos criar, para cada parte de nosso documento, um arquivo de extensão `.tex` e incluí-lo no `main.tex` por meio do comando `\input{}`.¹¹

Lembrem-se que estamos dentro de um diretório/pasta de nome, por exemplo, `microcurso-EMAT_LaTeX`. Então, para organizar mais as coisas, vamos criar alguns subdiretórios: `figs/`, nele colocaremos TODAS as figuras utilizadas no documento; `tex/`, nele colocaremos os arquivos `.tex` que incluiremos no arquivo principal.

Inicialmente, vamos criar dois arquivos: `pacotes.tex` e `intro.tex`; e, à medida que nosso texto for aumentando, criaremos os outros, inclusive o da bibliografia (que colocaremos no diretório `bib/`).

No arquivo `pacotes.tex`, simplesmente, colocamos os pacotes que desejamos carregar (não colocamos `\documentclass` ou `\begin{document}`):

¹¹ Futuramente, você pode desenvolver um pacote e salvá-lo numa extensão própria: `.sty`, que evita possíveis arquivos auxiliares na “compilação”.



pacotes.tex

```

1 \usepackage{fontspec} %-----> para codificação saída e escolha de fonte
2 \usepackage[brazil]{babel} %-----> suporte para idioma Brasileiro
3
4 \usepackage[a5paper]{geometry} %-----> configurações de margens
5
6 \usepackage{mathtools, amsthm, amssymb} %-> para expressões matemáticas e Cia
7 \usepackage{booktabs} %-----> estilo elegante para tabelas
8
9 \usepackage{graphicx} %-----> para inclusão de gráficos
10 \graphicspath{./fig/} %-----> local onde ficarão as figuras

```

**pacotes.tex (cont.)**

```

11
12 \usepackage{microtype} %-----> para protusão de caracteres
13
14 \usepackage[style=abnt, justify]{biblatex} %--> estilo para BibLaTeX para normas ABNT
15 \addbibresource{referencias.bib} % adiciona o arquivo .bib

```

Já no arquivo `intro.tex`, colocaremos nosso texto introdutório:

**intro.tex**

```
1 Olá! Esse é um pequeno texto em \LaTeX.
```

Agora, no arquivo principal, onde fazemos a composição tipográfica, devemos incluir os arquivos que estão em suas respectivas pastas.¹² Para carregálos corretamente, usamos os **caminhos relativos**. Por exemplo, para incluir o arquivo `pacotes.tex`, que se encontra na pasta `tex/`, devemos usar o comando (note que não precisa escrever a extensão do arquivo):

`\input{tex/pacotes}`

Assim, nosso arquivo principal ficaria:¹³

**main.tex**

```

1 % classe do documento =====
2 \documentclass[12pt]{article}
3
4 % preâmbulo =====
5 % pacotes carregados -----
6 \input{tex/pacotes}
7
8 % configurações -----
9
10 % início do documento =====
11 \begin{document}
12 %
13 \input{tex/intro} %-----> texto introdutório
14 %
15 %
16 \%printbibliography
17 %
18 % fim do documento -----
19 \end{document}

```

¹² No Overleaf a compilação no arquivo `main.tex`, independentemente do arquivo que você estiver editando. Basta, apenas, salvar (Ctrl + S).

¹³ Notem que deixei comentado o comando para exibir a bibliografia. Depois de aprendermos a adicionar os comandos no arquivo `.bib`, desmarcaremos o comentário.

E a organização visual seria algo como a Figura 11

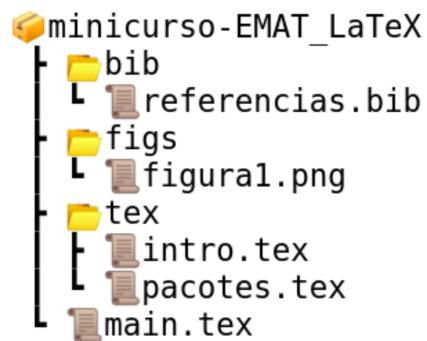


Figura 11. Visualização dos diretórios

Agora sim, estamos aptos para escrever nosso texto!

Começando a Escrever: O Modo Texto

5 Texto e Código

A escrita do código em \TeX possui algumas peculiaridades. Muitas delas, dependem de subjetividade e, portanto, fica à critério de cada um. Vou falar apenas a minha experiência.

Quando iniciei no \TeX fazia códigos na mesma linha, aglutinando comandos. Não me preocupava se num futuro eu teria dificuldade em entender o que escrevi, nem do trabalho que é encontrar um erro num código bagunçado! Existem boas práticas e, desde cedo, aconselho vocês seguirem.

Na realidade, já estamos fazendo isso! De fato, definimos uma estrutura de diretórios e organização dos arquivos `.tex`; indicamos como nomear arquivos; usamos comentários para separar partes do código; etc. Mas, a escrita, em si, deve contemplar alguns aspectos também! Ao meu ver, pense que todo o seu texto seja um código! Uma boa prática, então, é manter cada linha de seu texto com, no máximo, 80 caracteres. Além disso, eu costumo colocar cada parágrafo em uma linha de código. Isso é possível, pois o \TeX não considera mais de um espaçamento entre palavras! Um novo parágrafo só é formado de você deixar uma linha em branco entre ele e o anterior.

Por exemplo:¹⁴

Espaçamento entre palavras

```
Note que, mesmo eu escrevendo
espaçadamente, isso não influencia no
resultado!
Mesmo dando ``Enter'' \ldots
```

Parágrafo? Só se houver uma linha em branco entre eles!

Note que, mesmo eu escrevendo espaçadamente, isso não influencia no resultado! Mesmo dando “Enter” ...

Parágrafo? Só se houver uma linha em branco entre eles!

Os espaçamentos horizontais podem ser obtidos com o comando `\hspace{x cm}`, onde `x` representa o comprimento desejado.¹⁵ Os verticais seguem a mesma ideia: `\vspace{}`. É interessante notar que, se for usado números negativos, é feita uma contração no espaçamento.

Tabela 9 . Sumário da PART III

- Seção 5.** Texto e Código
- Seção 6.** Algo sobre Fontes
- Seção 7.** Alinhamento
- Seção 8.** Algo sobre Listas
- Seção 9.** Figuras e Tabelas: uma noçaozinha de nada
- Seção 10.** Rodapé e Minipage

Saiba mais

Complementando essa ideia de boas práticas na escrita do código, o texto *An essential guide to \TeX 2ε usage*, traz uma boa noção de comandos e pacotes obsoletos. É uma boa prática, sempre que possível, usar comandos mais novos e que, muitas vezes, foram criados para resolver problemas de anteriores!

¹⁴ Vejam a forma como colocamos aspas na palavra “Enter”! No início da palavra, dois acentos graves (crases); e, ao final da palavra, dois apóstrofos. Além disso, para os “...”, usamos \ldots.

¹⁵ Merece destaque o comando `\hfill`. Ele espaça, horizontalmente, duas palavras até o final da linha em curso.

Espaçamentos horizontais e verticais			
<pre>Meu \hspace{2cm} Deus!\\ Meu \hspace{-0.5cm} Deus!\\ Primeira \hfill Segunda\\ \vspace*{0.7cm}\\ Teste para vertical</pre>	Meu	Deus!	
	MeDeus!		
	Primeira		Segunda
			Teste para vertical

Notem que, para a “quebra de linha”, ou seja, escrever na linha subsequente sem iniciar outro parágrafo, usamos “\\”.

Para traços e hífens o \TeX faz quatro diferenciações: hífen nas palavras; traço entre números (para datas); travessão e sinal da subtração. Veja a diferença entre eles:

Traços, hífens e Cia	
<pre>Bem-vindo!\\ pg 12--36\\ Russia ---convervadora?\\ \$-1 < 0\$</pre>	<pre>Bem-vindo! pg 12–36 Russia —convervadora? –1 < 0</pre>

6 Algo (aquém do mínimo) sobre Fontes

Falar sobre fontes para o \TeX , precisaríamos de uns 5 minicursos como esse. Mas, gostaria de tangenciar alguns aspectos básicos, preparando uma estrutura mínima de conhecimento sobre o tema.

Num texto como esse, usa-se muitos tipos de fontes: existe a fonte para o corpo do texto (geralmente serifada); fontes para os títulos das seções, subseções, etc (geralmente sem serifa); fontes para códigos (geralmente monoespaçadas); e, fontes para expressões matemáticas!

Por padrão, a família de fontes utilizadas no \TeX é a *Computer Modern*. Ela vem com suporte para matemática! Nem todas as fontes possuem esse suporte! Tenha cuidado ao escolher uma fonte sem suporte para matemática, pois pode ficar bem distonante as expressões com o corpo do texto.

Há um catálogo de fontes disponíveis para uso no \TeX (The \TeX Font Catalogue):

<https://tug.org/FontCatalogue/>

**Atenção!**

Inicialmente, o pdf \TeX usava fontes empacotadas (como se fosse um pacote do \TeX). Entretanto, com os atuais interpretadores, como Lua \TeX , podemos escolher fontes instaladas em nossa máquina (por meio de pacotes como `fontspec`)!

Tenha cuidado ao selecionar fontes do catálogo acima: algumas delas foram pensadas para uso com pdf \TeX ! Verifique se há suporte para fontes OpenType (`.ttf` ou `.otf`).

Apenas por curiosidade, nesse texto estou usando:

Fonte	Descrição
Alegreya	para o corpo do texto (serifada);
Alegreya Sans	para destaques ou títulos (sem serifa);
Ubuntu Mono	para os códigos (monoespaciadas);
eulervm	pacote para fontes das expressões matemáticas.

Stephen G. Hartke, em 2006, escreveu um texto onde combina muitas fontes matemáticas de forma harmoniosa. Você pode tomar como referência algumas coisa de lá, mas lembre-se que deve adaptar para Lua \TeX ! O nome do texto é *A Survey of Free Math Fonts for T_EX and \TeX* e pode ser lido no *link*:

http://tug.ctan.org/info/Free_Math_Font_Survey/survey.pdf

Se você quiser conhecer um mínimo sobre tipografia, consulte esse link:

<https://trms.me/typography/>

Escolhida a fonte, podemos agora conhecer como modificar o tamanho e o estilo.

6.1 Tamanho da Fonte

São 10 (dez) os tamanhos disponíveis por padrão. Com eles modificamos **localmente** o tamanho de uma palavra, ou conjunto delas. Logicamente, esses tamanhos irão variar dependendo da escolha do tamanho da fonte do corpo do texto, feita no preâmbulo do documento (`documentclass`).

De uma forma geral, podemos resumir os comando na Tabela 10.

Observe que, para a fonte Alegreya, a diferença entre `\huge{}` e `\Huge{}` é imperceptível. Mas, dependendo da fonte escolhida, isso muda drasticamente.

Saiba mais

“A tipografia é a arte e o processo de criação na composição e impressão de um texto, física ou digitalmente. Assim como no design gráfico em geral, o objetivo principal da tipografia é dar ordem estrutural e forma à comunicação escrita.” (Wikipédia)

Tabela 10 . Tamanhos das Fontes em porções do texto

Comando	Saída
<code>\tiny{Matemática}</code>	Matemática
<code>\scriptsize{Matemática}</code>	Matemática
<code>\footnotesize{Matemática}</code>	Matemática
<code>\small{Matemática}</code>	Matemática
<code>\normalsize{Matemática}</code>	Matemática
<code>\large{Matemática}</code>	Matemática
<code>\Large{Matemática}</code>	Matemática
<code>\LARGE{Matemática}</code>	Matemática
<code>\huge{Matemática}</code>	Matemática
<code>\Huge{Matemática}</code>	Matemática

Se por algum motivo você quiser uma fonte ainda maior do que as expostas na Tabela 10, poderá usar (com extrama parcimônia) o comando

`\resizebox{tamH}{tamV}{Texto}`

Onde tamH e tamV são as dimensões (“tamanho”) da letra — que são dadas em alguma unidade de medida; e, Texto é o local onde você digitará a palavra que deseja modificar. Deve-se ter o cuidado com as escolhas das dimensões para que a palavra não fique distorcida. Se você quiser manter uma proporção entre a “largura” e “altura” da letra, defina uma dimensão e, no lugar da outra, coloque uma exclamação “!”.

Comandos	Saída
<code>\resizebox{1cm}{2cm}{Math}</code>	
<code>\resizebox{10pt}{20pt}{Math}</code>	
<code>\resizebox{!}{1cm}{Math}</code>	
<code>\resizebox{1cm}{!}{Math}</code>	

6.2 Ênfase na Fonte

Existem diversas ênfases nas fontes para usarmos no \TeX . Podemos colocar uma palavra em *italico*; **negrito**; “máquina de escrever”; SMALL CAPS, que seria

uma espécie de maiúsculas num tamanho menor; e, sublinhado.

A Tabela 11 exibe os comandos referentes:

Tabela 11. Ênfases na fonte

Comando	Saída
<i>italico</i>	<i>italico</i>
negrito	negrito
máquina de escrever	máquina de escrever
SMALL CAPS	SMALL CAPS
<u>sublinhado</u>	<u>sublinhado</u>



Atenção!

Existem muitos tópicos que não aborde, claro. Inclusive simplifiquei muitas coisas. Por isso, incentivo a leitura do texto [\TeX 2ε font selection](#) para um maior entendimento.

7 Alinhamento

Por padrão, o \TeX deixa o texto justificado. Entretanto, podemos deixar porções do texto de forma *centralizada*, *alinhada à esquerda* ou *alinhada à direita*. Para isso usamos os respectivos ambientes: `center`, `flushleft` e `flushright`.

Exemplo de Alinhamento do Texto

```
\begin{flushleft}
    texto à esquerda
\end{flushleft}
\begin{center}
    texto no centro
\end{center}
\begin{flushright}
    texto à direita
\end{flushright}
```

texto à esquerda

texto no centro

texto à direita

8 Algo sobre Listas

Vamos analisar, rapidamente, três tipos de listas no \LaTeX :

enumerate para listar itens de forma enumerada;

itemize para listar pontos ou observações;

description para descrever os itens.

Todas essas listas são “ambientes”, portanto devem iniciar com `\begin` e terminar com `\end`. Além disso, cada item listado deve ser precedido do comando `\item`. Ainda podemos mesclar uma lista em outra! Tudo dependerá de sua necessidade.

8.1 Description

Veja como foi produzido a lista no início dessa seção:

Descrevendo itens de uma lista

```
\begin{description}
\item[enumerate] para listar itens de forma enumerada;
\item[itemize] para listar pontos ou observações;
\item[description] para descrever os itens.
\end{description}
```

enumerate para listar itens de forma enumerada;

itemize para listar pontos ou observações;

description para descrever os itens.

8.2 Itemize

Algumas verdades:

Exemplo para itemize

```
\begin{itemize}
\item Esse curso foi mui
    rÁpido.
\item SÁo muitos comandos.
\item Mas, \LaTeX\ é legal.
\end{itemize}
```

- Esse curso foi mui rápido.
- São muitos comandos.
- Mas, \TeX é legal.

8.3 Enumerate

Como exemplo, veja um argumento falacioso:

Enumerando itens

```
\begin{enumerate}
    \item Cream cracker é feito de água e sal.
    \item O Mar é feito de água e sal.
    \item Logo, o Mar é um biscoitão.
\end{enumerate}
```

1. Cream cracker é feito de água e sal.
2. O Mar é feito de água e sal.
3. Logo, o Mar é um biscoitão.

É interessante esse ambiente para preparação de listas de atividades, mas devemos usar o pacote enumitem. Em seguida, use a sintaxe:



Sintax para listas

```
1 \begin{enumerate}[opções]
2     \item escreva
3     \item escreva
4     ...
5 \end{enumerate}
```

Em “opções” coloque o que você quer que se repita em cada início de item. Por exemplo, suponha que você queira “Q1”, “Q2”, etc, sem precisar ficar digitando diretamente .

Portanto, podemos fazer:

Saiba mais

Caso você queira melhorar o estilo exposto nas “Questões”, aconselho estudar o pacote enumitem, disponível no link: <https://www.ctan.org/pkg/enumitem>.

Simples Lista de Atividade

```
\begin{enumerate}[\textbf{Q1}]
    \item Seja $X$ um conjunto.
    \item Supondo que $Y$ também seja um conjunto.
    \item Sabe-se que $Z$ não é conjunto dessa vez.
\end{enumerate}
```

Q1 Seja X um conjunto.

Q1 Supondo que Y também seja um conjunto.

Q1 Sabe-se que Z não é conjunto dessa vez.

Veja que escrevemos “Q1”, em negrito, no argumento opções. Só precisa isso para o comando identificar que você quer uma repetição. Se você quisesse a repetição de “Q2” só precisaria mudar para “Q2”.

ção, em cada item, no estilo “(i), (ii), (iii), etc”, bastava colocar em opções: “(i)”. Esse mesmo raciocínio serve para estilos: “(I), (II), etc”; “(a), (b), (c), etc”.

9 Figuras e Tabelas: uma noçãozinha de nada

Daremos uma breve abordagem nesse tópico tão rico e que costuma dar uma dor de cabeça para muitos. 😊

É notória a potencialidade do \LaTeX nesses tópicos de inserção de Tabelas ou Figuras, visto que a enumeração é automática e você pode referenciar, ao longo do texto, todas esses elementos sem perder a contagem!!

Mas, para isso, a Tabela ou Figura deve estar em um ambiente próprio. O ambiente para Tabela é o *table*; e, o ambiente para Figura é o *figure*. Vamos analisá-los como um todo, para depois falarmos dos pormenores.

9.1 Elementos Flutuantes

Um Elemento Flutuante pode ser, por exemplo, uma Tabela ou uma Figura que se encontra disposta no texto. Geralmente, o ambiente para esse elemento é dado pela sintaxe:



Sintaxe para ambientes flutuantes

```
1 \begin{ambiente}[opções1]
2   opções2 + Elemento
3 \end{ambiente}
```

Em *opções1* podemos colocar *h*, *t*, *b*, *p* (*here*, *top*, *bottom* e *page*) para que o elemento flutuante se posicione, respectivamente, “aqui”, no local onde está; no “topo da página”, no “final da página”, e em “outra página”. Caso você mantenha uma preferência na ordem, use uma exclamação antes do comando. Por exemplo, “!*htbp*”, diz ao \LaTeX que você quer posicionar o elemento preferencialmente “aqui” (onde ele está); mas se não for possível, coloque no topo da página; mas se não der, etc.¹⁶



Atenção!

Veja que o \LaTeX pode aceitar ou não sua preferência. Geralmente, o programa dispõe o elemento flutuante de forma tipograficamente harmoniosa.

¹⁶ Caso você insista em colocar o elemento flutuante exatamente onde ele se encontra, use o pacote **float** e no lugar da *opção1* coloque simplesmente **H** (maiúsculo).

Resumidamente temos a Tabela 12.

Tabela 12. Resumo dos parâmetros para o ambiente `table`

Parâmetros	Descrição
<code>h</code>	Situa a figura exatamente onde se encontra no texto do editor. Nem sempre é possível executá-la, visto que a escala da figura pode não conter no espaço da página!
<code>t</code>	Posiciona a figura na parte superior da página
<code>b</code>	Posiciona a figura na parte inferior da página
<code>p</code>	Posiciona a figura numa próxima página

Em `opções2`, podemos usar os comandos da Tabela 13.

Tabela 13. Opções para Elementos Flutuantes

Comando	Função
<code>\centering</code>	deixa o elemento flutuante centralizado.
<code>\caption{}</code>	Legenda do elemento flutuante.
<code>\label{}</code>	Rótulo, uma marcação para possível referência (citação) no texto.

Você pode colocar as `opções2` antes ou depois do *Elemento*, contudo o comando `\label` sempre deve vir depois do comando `\caption`, para evitar erros na enumeração dos elementos no texto.¹⁷

No lugar de *Elemento*, deve-se colocar o ambiente que você deseja; no caso, a *Figura* ou a *Tabela*. Vamos estudar esses ambientes separadamente.

9.2 Tabelas

Esse é um outro tópico que precisaríamos de, pelo menos, uns 4 minicursos como esse. Portanto, só abordarei o mínimo de um tópico específico. Vou usar o pacote **booktabs**, que oferece um estilo de tabela bem elegante e de sintaxe simples.

Para criarmos essa tabela, usamos o ambiente `tabular`. Sua sintaxe é:

¹⁷ o `\label` sempre toma como referência algum elemento anterior a ele. Assim se você coloca essa marcação antes do `\caption` ele pegará a referência anterior a esse elemento flutuante!

 **Sintaxe para tabela**

```

1 \begin{tabular}{opções}
2   \toprule %-----> linha de cima
3   A & B & ... & n \\ %--> quebra de linha
4   \midrule %-----> linha do meio
5   escreva & escreva & ... & escreva \\
6   ... ... ... ...
7   escreva & escreva & ... & escreva \\ %--> não esquecer
8   \bottomrule %-----> linha do fim
9 \end{tabular}

```

Em **opções** devemos preencher com as letras: **c**, **l** ou **r** (center, left, right — centro, esquerda, direita). Esse preenchimento é de acordo com a quantidade de colunas de sua tabela.¹⁸ Por exemplo, suponha que você queira construir uma tabela com cinco colunas: sendo a primeira coluna alinhada à esquerda, a última coluna alinhada à direita e as demais colunas alinhadas no centro. Então, deve colocar em **opções**: **lcrrc**.

Os comandos **\toprule**, **\midrule** e **\bottomrule** são linhas horizontais estilizadas. Note que se você quiser colocar uma linha estilizada entre as linhas da tabela, deve usar **\midrule**. Não esqueça de quebrar a linha antes do comando **\bottomrule**.

O símbolo **&** é utilizado para separar os elementos das colunas e “****” quebra a linha na tabela.

Veja um simples exemplo:

Uma simples, mas elegante tabela

```
\begin{tabular}{ccc}
\toprule
\textbf{Número} && \textbf{Raiz}\\
\midrule
1 && 1\\
4 && 2\\
100 && 10\\
603729 && 777\\
\bottomrule
\end{tabular}
```

Número	Raiz
1	1
4	2
100	10
603729	777

Agora que já sabemos como construir uma simples tabela, podemos nos perguntar: como colocar a legenda nessa tabela?

Para isso vamos inserir o ambiente *tabular* dentro do ambiente *table*!

Queremos produzir algo como a Tabela 14.

Os comandos ficam assim³:

¹⁸ “c, l, r” são minúsculos!

Tabela 14 . Raíz Quadrada de *n*

Número	Raiz
1	1
4	2
100	10
603729	777

³Mesmo colocando três colunas, deixamos uma sem usar. Isso serve para dar um maior espaço entre as colunas que contém o texto. Tudo dependerá da necessidade!



Exemplo com ambiente para Tabelas

```

1 \begin{table}![htbp]
2   \centering
3   \caption{Raiz Quadrada de \textit{n}}
4   \label{tab:raizq}
5   \begin{tabular}{ccc}
6     \toprule
7     \textbf{Número} && \textbf{Raiz} \\
8     \midrule
9     1 && 1 \\
10    4 && 2 \\
11    100 && 10 \\
12    603729 && 777 \\
13    \bottomrule
14  \end{tabular}
15 \end{table}

```

Perceba que a enumeração segue a sequência das tabelas do texto.¹⁹

Outra coisa: o comando `\label` não apareceu no resultado final da Tabela 14. De fato, esse comando é uma marcação, um rótulo! Se em algum momento do texto quisermos citar a tabela em questão, basta usar o comando `\ref{marcação}`. Onde em `marcação` devemos colocar **exatamente** o rótulo que estabelecemos. Assim, para citarmos a tabela do exemplo em questão fazemos: `\ref{tab:raizq}`.

Oi
<code>\ldots considere a Tabela~\ref{tab:raizq}</code>
<code>...considere a Tabela 14</code>

¹⁹ O simbolo “~” antes de “`\ref`” serve para que não haja quebra de linha entre esses elementos.

Já pensou no final de uma linha aparecer o nome “Tabela” e no início da outra o número “1.4”? De uma forma geral, se você quiser que certas palavras não se separem ao longo de uma linha no texto, coloque o simbolo ~ entre elas!

9.3 Figura

O ambiente apropriado para incluir uma figura é o *figure*. É muito semelhante a estrutura do ambiente *figure* com relação ao ambiente *table*. São os mesmos parâmetros descritos na Tabela 12.

Todavia, para inserir uma figura no texto, usamos a sintaxe:

```
\includegraphics[opções]{nome}
```

Em `nome` devemos inserir, exatamente, o nome que salvamos a figura. Já em `opções`, temos os parâmetros descritos na Tabela 15

Chamamos à atenção sobre possíveis distorções nas imagens.²⁰ Os comandos “`width= x cm`” e “`height= x cm`” quando usados separadamente, estabelecem automaticamente uma proporção adequada à figura em questão. Respectivamente, fixam a *largura* e a *altura* desejada. Mas, se esses comandos forem usados conjun-

Saiba mais

Lembre-se que os formatos são JPEG, PNG ou PDF.

Prefira o PDF; pois, por ser um formato vetorial, não há distorção ao ampliar a imagem, por exemplo.

²⁰ Lembre-se que é imprescindível salvar a figura na pasta `\figs/`. Para isso, no preâmbulo, colocamos o comando `\graphicspath{./figs/}`.

Tabela 15 . Parâmetros para o comando `\includegraphics`

Parâmetros	Descrição
<code>width</code>	Define a <i>largura</i> da figura. Caso seja usado sozinho, estabelece uma proporção com a <i>altura</i> da figura.
<code>height</code>	Define a <i>altura</i> da figura. Caso seja usado sozinho, estabelece uma proporção com a <i>largura</i> da figura.
<code>scale</code>	Define uma <i>escala</i> para figura. Neste caso, há uma proporção tanto na altura, quanto na largura da figura
<code>angle</code>	Define um <i>ângulo</i> de rotação à figura, no sentido positivo.

tamente, a imagem pode realmente ficar distorcida. Use-os, simultaneamente, quando for realmente necessário.

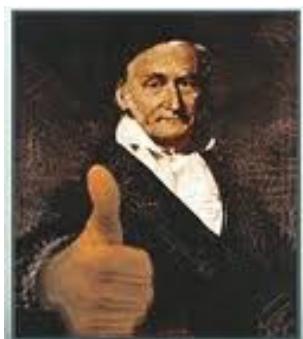


Figura 12 . Gauss dando legal com largura de 4 cm.

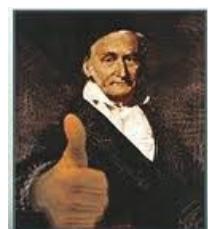


Figura 13 . Gauss dando legal com altura de 3 cm.



Figura 14 . Gauss?

A Figura 12 foi obtida usando 4 cm de largura. Para tanto usamos `width=4cm`. A Figura 13 foi obtida usando 3 cm de altura. Para tanto usamos `height=3cm`. Por fim, a Figura 14 foi obtida usando, simultaneamente, 3.3 cm de largura e 1 cm de altura, ou seja, usamos `width=3.3cm` e `height=1cm`.

Veja os comandos separada e respectivamente:

✍
Figura com Largura 4 cm

```

1 \begin{figure}[!htbp]%%<-- ambiente para figura
2 \includegraphics[width=4cm]{gauss}%%<-- Incluindo a figura
3 \caption{Gauss dando legal com largura de 4\,cm.}%legenda
4 \label{fig:LeGauss1}%%<-- rótulo
5 \end{figure}

```



Figura com Altura 3 cm

```

1 \begin{figure}[!htbp]
2   \includegraphics[height=3cm]{gauss}
3   \caption{Gauss dando legal com altura de 3\,cm.}
4   \label{fig:LeGauss2}
5 \end{figure}
```



Figura com Largura 3.3 cm e Altura 1 cm

```

1 \begin{figure}[!htbp]
2   \includegraphics[width=3.3cm,height=1cm]{gauss}
3   \caption{Gauss?}
4   \label{fig:LeGauss3}
5 \end{figure}
```

Talvez você esteja se perguntando como colocamos figura ao lado de figura não é mesmo? Mas, veremos isso quando falarmos sobre `minipage`.

Por hora, vamos analisar como colocar figura em toda margem ou em parte dela.

Ora, para que uma figura ocupe certa proporção da margem disponível à escrita, basta usar o comando `width = x\linewidth`. Onde “x” é a proporção (contração) que deseja da margem.²¹ Por exemplo, as Figuras 15 e 16, representam, respectivamente, a totalidade da margem em uso ($1\backslash linewidth$) e 30% da margem em uso.

²¹ Note que $x \in (0, 1]$.



Figura 15 . Foto ocupando toda margem em uso



Figura 16 . Foto ocupando 0.3 da margem em uso

Os respectivos comandos são:



Figuras em diferentes larguras

```

1 \begin{figure}[!htbp]
2   \centering
3   \includegraphics[width=1\linewidth]{por}
4   \caption{Foto ocupando toda margem em uso}
5   \label{fig:amargosa1}
6 \end{figure}
7
8 \begin{figure}[!htbp]
9   \centering
10  \includegraphics[width=0.3\linewidth]{por}
11  \caption{Foto ocupando 0.3 da margem em uso}
12  \label{fig:amargosa2}
13 \end{figure}

```

Caso precise rotacionar uma figura em “ α ” graus, use a opção “`angle= α` ” como parâmetro.²²

Veja as Figuras 17 e 18 para referência.



Figura 17 . Avião rotacionado em 45° .

Figura 18 . Avião rotacionado em 90° .

²² Caso queira citar uma figura específica dentro do texto, o comando é o mesmo usado na citação de tabelas:
`\ref{marcação}`.



Rotação em 45°

```

1 \begin{figure}[!htbp]
2   \centering
3   \includegraphics[width=3cm, angle=45]{aviao}
4   \caption{Avião rotacionado em  $45^\circ$ .}
5   \label{fig:avi2}
6 \end{figure}

```



Rotação em 90°

```

1 \begin{figure}[!htbp]
2   \centering
3   \includegraphics[width=3cm, angle=90]{aviao}
4   \caption{Avião rotacionado em  $90^\circ$ .}
5   \label{fig:avi3}

```



Rotação em 90° (cont.)

```
6 \end{figure}
```

Também é possível inserir uma figura ao longo de uma linha. Como exemplo, considere o símbolo do cérebro treinando .

Para tanto, usamos o código:



Figura inline

```
1 ... treinando \fbox{\includegraphics[scale=0.18]{treino}}
```

Note que para deixar o contorno de uma “caixa”, escrevemos dentro de `\fbox{}`. Caso não queira o contorno, escreva simplesmente:

`\includegraphics[scale=0.18]{treino}`

10 Rodapé e Minipage

10.1 Rodapé

Para inserir uma *nota de rodapé* use o comando: `\footnote{}`.

```
\ldots isso completa a demonstração do lema\footnote{Esse resultado vale para um caso mais geral, onde $n$ é negativo ou zero.}
```

...isso completa a demonstração do lema^a

^aEsse resultado vale para um caso mais geral, onde n é negativo ou zero.

10.2 Minipage

O ambiente `minipage`, como o próprio nome sugere, cria uma “mini página” dentro do texto. Sua sintaxe, básica, é algo como:



Sintax básica do ambiente `minipage`

```
1 \begin{minipage}[posição]{comprimento_1}
2   <conteúdo da primeira minipage>
3 \end{minipage}
4 \hfill
```



Syntax básica do ambiente `minipage` (cont.)

```
5 \begin{minipage}[posição]{comprimento_2}
6   <conteúdo da segunda minipage>
7 \end{minipage}
```

Onde `comprimento_1` e `comprimento_2` especifica a largura de cada `minipage`.²³ Eles podem ser substituídos por `k\linewidth`, com $k \in (0, 1]$; caso você queira uma proporção da largura do `minipage` com a largura da página. Além disso, em `posição`, podemos colocar os parâmetros “t”, “c”, “b” para alinhar no topo da `minipage`, no centro ou no final desse ambiente. Certamente você pode trocar o comando `\hfill` por algum espaçamento que achar conveniente.

Para exemplificar, vamos colocar figura ao lado de figura, como as Figuras 19 e a Figura 20.



Figura 19 . Jardim de Amargosa



Figura 20 . Bosque de Amargosa

Perceba nos comandos abaixo, a localização do `minipage`: dentro do ambiente `figure!` Além disso, note onde os comandos de centralização estão localizados também.

Veja os comandos:



Figura ao lado de figura com `minipage`

```
1 \begin{figure}[!htbp]
2   \begin{minipage}[b]{0.45\linewidth}
3     \centering
4     \includegraphics[width=6cm]{praca}
5     \caption{Jardim de Amargosa}
6     \label{fig:praca}
7   \end{minipage}
8   \hfill
9   \begin{minipage}[b]{0.45\linewidth}
10    \centering
11    \includegraphics[width=6cm]{bosque}
12    \caption{Bosque de Amargosa}
13    \label{fig:bosque}
14  \end{minipage}
15 \end{figure}
```

Saiba mais

Como você faria um cabeçalho básico para sua escola usando combinações de elementos que aprendemos até aqui?

A Beleza do L^AT_EX: o Modo Matemático

11 Teorema

Tabela 16 . Sumário da PART III

| [Seção 11](#). Teorema

Referências

PÉGOURIÉ-GONNARD, Manuel. **A guide to LuaLaTeX**. Edição: Comprehensive TeX Archive Network (CTAN). Guia para entender LuaLaTeX. [S.l.], 5 mai. 2013. p. 14. Disponível em: <<https://ctan.math.illinois.edu/info/luatex/lualatex-doc/lualatex-doc.pdf>>. Acesso em: 22 jan. 2022.

ROBERTSON, Will. **The fontspec package**: Font selection for XeLaTeX and LuaLaTeX. Edição: Comprehensive TeX Archive Network (CTAN). Pacote para XeLaTeX e LuaLaTeX. Versão 2.8a. [S.l.], 15 jan. 2022. p. 66. Disponível em: <<https://ctan.dcc.uchile.cl/macros/unicodetex/latex/fontspec/fontspec.pdf>>. Acesso em: 18 jan. 2022.

WRIGHT, Joseph. **Learn LaTeX**. [S.l.: s.n.], 16 dez. 2021. Disponível em: <<https://www.learnlatex.org/>>. Acesso em: 22 jan. 2022.