UNIVERSIDADE FEDERAL DO PARANÁ

PET Computação

LaTeX

Curitiba

2010

Sumário

LaTeX, aquilo que sai da árvore? 7

Considerações dos autores 7

Compilação de um arquivo `.tex` 8

1 Aprendendo na Prática 9

1.1 Começando um documento LaTeX 9

1.1.1 Estrutura básica 10

1.1.2 Conteúdo do arquivo `.tex` 11

1.1.3 Classes de documentos 13

1.2 Estruturando o texto 14

1.2.1 Título 14

1.2.2 Seções 15

1.2.3 Listas 16

1.2.4 Notas 18

1.2.5 Citações 18

1.3 Fontes e Tamanhos 19

2 A Verdadeira Beleza do LaTeX 21

2.1 Modo Matemático 21

2.1.1 Escrevendo fórmulas matemáticas 22

2.1.2 Alguns comandos importantes 25

2.2 Tabelas 28

2.3 Inserindo Figuras no LaTeX 29
2.3.1 Referenciar figuras 31
2.3.2 Legenda das figuras 31

3 Seguindo Normas **33**
3.1 Sumário 33
3.2 Índice remissivo 34
3.2.1 Construindo o índice remissivo 35
3.3 Bibliografias 36
3.3.1 Usando o BibTeX 37
3.3.2 Formato BibTeX 38
3.4 Documentos nos padrões da ABNT 39
3.4.1 Estrutura essencial 39
3.4.2 Outros elementos 40
3.4.3 Considerações finais sobre abnTeX 41

4 LaTeX Beamer **43**
4.1 O que é Beamer? 43
4.2 Como começar? 43
4.2.1 Preâmbulo 43
4.2.2 Frames 44
4.2.3 Seções 45
4.3 Fazendo uma apresentação 46
4.3.1 Overlays 46
4.3.2 Uncover, only e invisible 46
4.4 Efeitos 47
4.4.1 Blocos de Texto 47

4.4.2 Tabelas 47

Anexo A – Lista de símbolos matemáticos **49**

LaTeX, aquilo que sai da árvore?

Não. Sem piadinhas contra seringueiros. É LaTeX, não *látex*. LaTeX (de pronúncia Lá-téch) é um sistema tipográfico de altíssima qualidade, considerado por muitos tipógrafos uma obra de arte. É largamente usado pela comunidade científica, e se um dia você pensa em redigir um texto técnico ou um artigo científico, recomendamos fortemente você gastar algumas horas para entender e aprender LaTeX.

Dentre as primeiras coisas a serem ditas sobre LaTeX, a que mais aparece é a seguinte: LaTeX não é um **processador de texto**! Tudo bem, o que isso quer dizer? Processadores de texto são *softwares* completos, que processam, editam, e mostram a cara do documento final, tudo ao mesmo tempo. Daí vem uma expressão bastante conhecida no mundo dos processadores de textos: WYSIWYG (What You See Is What You Get), ou algo como "O que você vê é o que você tem". O que não foge da realidade, experimente abrir um processador de texto tal como Microsoft Word, OpenOffice Word Processor, ou qualquer outro equivalente, e digitar algumas coisas. Vá então ao mundarel de ícones e opções disponíveis nos menus do seu programa e experimente modificar a forma do texto que você tem em mãos (ou em tela), o que você vê não é o que você tem?

Ok, ok. Sabemos que isso não explica exatamente a idéia de um WYSIWYG. Para entender melhor, vamos direto ao nosso louvável LaTeX e demonstrar os conceitos através das diferenças. Como começar um documento em LaTeX?

Bom, a primeira coisa é escrever o conteúdo seguindo algumas regrinhas básicas. Sim! Você precisa focar-se no conteúdo. Taí uma diferença crucial: o LaTeX foi desenvolvido para que você possa focar-se completamente no conteúdo, despreocupando-se com a aparência (depois de matar a curiosidade de ver como seus documentos ficam, claro).

Considerações dos autores

Os autores deste livro levaram em conta algumas considerações para o estudo da linguagem tipográfica LaTeX. A primeira delas é que você esteja usando um sistema operacional baseado em *GNU/Linux*, portanto se você estiver diante de uma máquina com *Windows*, ou *Mac OS*,

procure informações sobre o procedimento para a compilação de seu arquivo `.tex`. O restante é o mesmo, tanto para *GNU/Linux*, quanto *Windows* ou *Mac OS*.

Uma segunda consideração dos autores é que você tenha pouca, ou nenhuma, familiaridade com o sistema *GNU/Linux* e os jargões da área de informática. Portanto o início do livro, principalmente o início do capítulo 1, traz instruções passo-a-passo, de modo que o leitor possa se familiarizar com os comandos, seus significados, e os conceitos que estão envolvidos.

Compilação de um arquivo `.tex`

Antes de começarmos com comandos e estruturas, é importante explicar o que se entende por compilar um arquivo `.tex`.

Assim como programas escritos em várias linguagens de programação presentes na área de computação, um documento escrito em LaTeX deve ser compilado para que seu conteúdo seja gerado em um arquivo de saída de extensão `.dvi`. Para compilar um documento, usa-se o comando `latex`, que analisa o arquivo `.tex` de entrada, verificando se todos os comandos foram escritos corretamente, e retorna três arquivos: um `.aux`, outro `.dvi` e outro `.log`. Esses arquivos representam, respectivamente, auxílio de referências para o compilador TeX, saída gerada a partir do conteúdo do arquivo `.tex` pelo sistema LaTeX, um registro dos eventos relevantes efetuados pelo LaTeX em tempo de compilação.

Outros arquivos auxiliares tambem são gerados na compilação e ao mesmo tempo lidos para a organização de algumas referências para tratar de alguns comandos, como `\tableofcontents`, que gera o sumário. Assim é necessário compilar duas vezes o mesmo arquivo para que o conteúdo seja atualizado na versão final.

1 Aprendendo na Prática

Ao iniciarmos um trabalho escrito, o primeiro passo é termos em mãos algumas ideias. Em seguida podemos começar a escrevê-lo, rabiscando os pontos principais em um papel ou editando um arquivo de texto em um computador. Cada um pode seguir modos diferentes de organização até a conclusão do trabalho, com um texto final de apresentação. O autor do texto então, tem em mente quais ideias estão em cada parte deste trabalho e deseja que o leitor seja capaz de compreendê-las. LaTeX nos deixa livre de modo que não tenhamos preocupação com a formatação, e nossa tarefa é "marcar" nosso texto de maneira que o LaTeX possa finalizá-lo de maneira elegante. Neste capítulo veremos como funcionam essas "marcas" para que tenhamos nosso trabalho bem apresentado, utilizando esta ferramenta tipográfica.

1.1 Começando um documento LaTeX

Vamos começar a entender a estrutura de um documento LaTeX e seu funcionamento redigindo um pequeno texto. Abra seu editor de texto favorito e copie *exatamente* como a seguir:

```
\documentclass{article}
\usepackage[brazil]{babel}
\usepackage[utf8]{inputenc}
\begin{document}
Esse é o meu \emph{primeiro} documento em
        \LaTeX.
\end{document}
```

Você deve tomar cuidado para não confundir o caractere \ (*backslash*, ou contra-barra) com o / (*slash*, ou barra), pois os comandos LaTeX são todos chamados com a contra-barra (\). Salve esse arquivo com o nome que você preferir, aqui consideraremos que o arquivo foi salvo como `documento.tex`. Outro fato em que você deve prestar atenção é a extensão de seu documento: ele deve ser `.tex`. Você pode não adicionar a extensão `.tex`, ou adicionar outra extensão, no

entanto, é mais fácil para organizar seus arquivos, e portanto, fortemente aconselhável manter as extensões corretas dos arquivos. Para compilar seu documento, primeiramente abra um terminal e digite

```
latex documento.tex
```

e, para visualizá-lo, digite

```
xdvi documento.dvi
```

Para convertê-lo em um arquivo no formato `.pdf`, mais comumente usado, digite

```
dvipdf documento.dvi nome_do_novo_documento.pdf
```

e visualize-o com seu programa visualizador preferido. Por exemplo:

```
xpdf nome_do_novo_documento.pdf
```

Agora note o que pode ser visto no documento: uma folha já formatada com seu texto contido, no caso,

> Esse é o meu *primeiro* documento em LaTeX.

1.1.1 Estrutura básica

Agora voltemos ao texto que você digitou no arquivo fonte `documento.tex`. A primeira linha especifica a *classe* de seu documento, no nosso caso trata-se de um artigo. Se, por acaso, você quisesse escrever um livro, deveria mudá-la para `\documentclass{book}`. Veremos mais classes no decorrer deste livro. Duas linhas iniciam com `\usepackage`, que indicam que estamos escrevendo um texto com a lingua usada no *Brasil* e que a codificação de caracteres é `utf8`, que significa que utilizaremos acentuação de letras no texto. As próximas linhas indicam o início de um documento LaTeX, seu conteúdo e o fim, respectivamente. Note que todo conteúdo deverá estar entre `\begin{document}` e `\end{document}`. O comando `\emph{}` enfatiza (*emphasize*, em inglês) a palavra dentro das chaves. Um documento LaTeX seguirá então esta estrutura básica e chamaremos tudo o que fica antes de um `\begin{document}` de *preâmbulo*.

1.1.2 Conteúdo do arquivo `.tex`

Experimente digitar a seguinte frase e compilá-la:

Eu acho LaTeXlegal.

```
Eu acho \LaTeX legal.
```

Note que o espaço entre LaTeX e legal desapareceu. Por quê? O motivo é que o TeX engole todos os espaços que vêm depois de um comando, no entanto, não se trata de um erro, mas sim uma característica da linguagem tipográfica TeX. Mas como corrigir isso? É bem simples, é só modificar o texto para

Eu acho LaTeX legal.

```
Eu acho \LaTeX\ legal.
```

ou ainda

Eu acho LaTeX legal.

```
Eu acho \LaTeX{} legal.
```

Adicionar o caracter \ ou {} faz com que o TeX entenda a macro \LaTeX como uma chamada a alguma referência, ignorando os espaços após o comando. No LaTeX muitos caracteres de espaçamento são tratados como apenas um espaço. E uma linha em branco entre duas de texto define o fim de um parágrafo. Muitas linhas em branco também são entendidas como uma só. Veja só isso:

Não importa se você escreve um ou muitos espaços depois de uma palavra.

Uma ou mais linhas em branco iniciam um novo parágrafo.

```
Não importa se                  você
escreve um ou       muitos espaços
depois de uma palavra.


Uma ou mais linhas em branco iniciam um
novo parágrafo.
```

Você pode forçar uma quebra de linha com \\

Essa é a primeira linha.
Essa é a segunda.

```
Essa é a primeira linha.\\
Essa é a segunda.
```

Você também pode dar um argumento especial, que aumenta a distância vertical entre as linhas. Por exemplo:

Essa é a primeira linha.

Essa é a segunda.

```
Essa é a primeira linha.\\[10pt]
Essa é a segunda.
```

Você certamente percebeu que o TEX alinha o texto do seu próprio jeito, independentemente da forma como o texto está formatado. Agora, suponha que você queira fazer algo assim:

PET Computação

Certificado

Certificamos que o Sr. Fulano De Tal participou do curso de LATEX básico realizado pelo grupo PET Computação da UFPR - Universidade Federal do Paraná.

O Coordenador
PET Computação UFPR

```
\begin{center}
    PET Computação\\[.50cm] Certificado
\end{center}
\noindent Certificamos que o sr.Alguém
participou do curso de \LaTeX{} básico
realizado pelo grupo PET Computação da
UFPR - Universidade Federal do Paraná.
\begin{flushright}
   O Coordenador \\ PET Computação UFPR
\end{flushright}
```

Os comandos `\begin{center} ... \end{center}` colocam o texto exatamente no centro da página, e os comandos `\begin{flushright} ... \end{flushright}` levam o texto para a margem direita. Falta ainda mencionar o correspondente para a margem esquerda, que é `\begin{flushleft} ... \end{flushleft}`. Esses são exemplos, do que o LATEX chama de ambiente, que aparecem na forma de `\begin{nome} ... \end{nome}` onde *nome* é o nome do ambiente.

Podemos escrever nosso texto em um arquivo `.tex` como em qualquer documento de texto, mas temos que saber que alguns caracteres imprimíveis que fazem parte da pontuação são especiais, e não representam seu significado literal. São estes os caracteres:

```
# $ % & _ { } ~ ^ \
```

e para utilizar os 7 primeiros devemos colocar uma contra-barra na frente do caractere

assim: # $ % & _ { }

```
assim: \# \$ \% \& \_ \{ \}
```

Ainda, o tratamento das aspas simples e dupla se dá um pouco diferente, pois aqui aspas direita e esquerda são símbolos diferentes.[1]. Assim como o hífem e travessão possuem uma notação especial.

“Abre aspas dupla é ‘diferente’ de fecha aspas dupla” e um hífem inter-palavra, um hífem – e ainda um—travessão.

```
‘‘Abre aspas dupla é ‘diferente’
de fecha aspas dupla’’ e um hífem
inter-palavra, um hífem -- e ainda
um---travessão.
```

[1] Abre aspas dupla são duas crases do teclado e fecha aspas dois apóstrofos, para aspas simples são os mesmos símbolos, mas individuais

Podemos incluir comentários em nosso documento, ou seja, anotações que não serão apresentadas no arquivo de saída, mas que podem nos ajudar para nossa organização. Tudo o que vier depois do caracter %, será descartado para a produção do texto.

1.1.3 Classes de documentos

Como foi dito anteriormente, todo arquivo em LaTeX começa com a especificação de uma classe. Para isso usa-se o comando `\documentclass{..}`. Existem várias classes disponíveis no LaTeX, como a `article` (artigo) já vista, `book` (livro), `report` (relatório), `letter` (carta), dentre outras. Todas têm suas características próprias, no entanto, compartilham muito em comum. É possível, além de especificar o tipo do documento, modificar o formato padrão da página e o tamanho da fonte, através de algumas opções. A sintaxe do comando `\documentclass` é na verdade assim

```
\documentclass[opções]{classe}
```

Note que as opções devem estar entre colchetes, e não entre chaves. Vejamos então o que podemos especificar nestas opções.

Tamanho da Fonte Nós podemos selecionar o tamanho da fonte para o texto normal com uma das opções `10pt`, `11pt` ou `12pt`, que são especificadas no comando `documentclass`, como por exemplo,

```
\documentclass[11pt]{classe}
```

para que o texto normal fique com o tamanho de fonte `11pt`. O tamanho padrão é `10pt`, se for o que você quiser, não se preocupe em especificar qualquer opção de tamanho.

Tamanho do Papel Nós sabemos que o LaTeX possui seus próprios métodos de quebrar linhas para fazer parágrafos. Ele também faz quebras verticais para produzir páginas. Mas para que essas quebras funcionem, é preciso saber a largura e altura da papel usado. As várias opções para selecionar o tamanho do papel são dadas na tabela abaixo:

letterpaper	11x8.5 in	a4paper	20.7x21 in
legalpaper	14x8.5 in	a5paper	21x14.8 in
executivepaper	10.5x7.25 in	b5paper	25x17.6 in

O tamanho de papel padrão é o `letterpaper`.

Formato da Página Existem opções para colocar o texto em uma coluna ou em duas. Isso é feito por `onecolumn` ou `twocolumn`. O padrão é `onecolumn`.

Há também a opção para especificar como o documento será impresso, em um só lado, frente e verso. Os nomes das opções são `oneside` e `twoside`.

O padrão é `oneside` para `article`, `report` e `letter` e `twoside` para `book`.

1.2 Estruturando o texto

Existem diferentes partes lógicas de um texto que para melhor representação, devem ser destacadas seguindo algumas regras de formatação, como em uma lista ou citação em um parágrafo, divisão em seções, etc... Existem diversas estruturas e neste capítulo apresentamos as principais.

1.2.1 Título

Todo texto deve possuir um título, certo? Para isso, o LaTeX tem a parte de título, que consiste no nome do documento, o nome do(s) autor(es) e às vezes a data. Para produzir um título, temos que usar os seguintes comandos:

```
\title{Nome do Documento}
\author{Nome do Autor}
\date{data}

\begin{document}

\maketitle

\end{document}
```

Note que, depois de especificados os argumentos de `\title, \author, \date`, que estão no preâmbulo, tem-se que utilizar o comando `\maketitle` para que esta parte do documento seja tipografada. Se existirem vários autores e seus nomes forem separados pelo comando `\and`, então os nomes aparecerão em duas colunas. É possível deixar um dos argumentos vazios. O comando `\date{}` não imprime data alguma. Note que, se você simplesmente omitir o comando `\date`, a data atual será impressa.

Para a classe `book`, o título estará em uma página vazia.

1.2.2 Seções

Seções certamente organizam melhor um documento. Dividir um documento em partes lógicas, aqui chamadas de várias formas a seguir, é uma boa prática e o LaTeX trata isso de várias maneiras. Existe uma hierarquia de particionamentos.

Se você estiver escrevendo nas classes `book` e `report`, existem os seguintes comandos

```
\chapter
\section
\subsection
\subsubsection
\paragraph
\subparagraph
```

Exceto por `\chapter`, todos esses comandos são disponíveis na classe artigo. Por exemplo o cabeçalho do começo desta seção foi produzido por `\section{Estruturando o texto}`. Veja o resultado de outros comandos compilando o conteúdo abaixo em um arquivo `.tex`

```
\section{Exemplo}
Este é um exemplo de subseção. Note como
as subseções são numeradas.

\subsection{Subexemplo}
Este é um exemplo de subsubseção. Continue
observando a numeração.

\paragraph{Nota}
Parágrafos e subparágrafos não tem números.

\subparagraph{Subnota}
Subparágrafos tem uma identação adicional.
```

Nas classes `book` e `report`, o comando `\chapter` leva para o começo de uma nova página e imprime a palavra capítulo e seu respectivo número, e também o nome que foi dado como argumento para o comando. O comando `\section` produz dois números separados por um

ponto, indicando os números do capítulo e da seção. As subseções tem três números, indicando capítulo, seção e subseção. As subsubseções e os comandos em baixo na hierarquia não possuem números.

Na classe `article`, `\section` é o maior comando na hierarquia, e produz um único número, como `\chapter` na classe `book`. Neste caso, as subsubseções também possuem números, mas os comandos abaixo dela não. Mas, como fazer, se eu não quero que apareçam os números nos comandos de particionamento? É simples, é só colocar um asterisco após o comando. Por exemplo, `\section*{nome}`. Assim como outros comandos em LaTeX, a forma * é uma alternativa ao uso do comando, e neste caso, há a omissão da sua numeração.

1.2.3 Listas

Há três maneiras de se fazer uma lista: numerando os ítens, os identificando com um marcador comum, ou ainda, com elementos destacados. Para isto, temos os ambientes em LaTeX abaixo.

itemize fornece a você uma lista como essa:

Devemos ter em mente quando usamos o TeX

- TeX é uma linguagem tipográfica e não um editor de texto.
- TeX é um programa e não uma aplicação.

```
Devemos ter em mente quando usamos o
\TeX
\begin{itemize}
\item \TeX{} é uma linguagem
tipográfica e não um editor de texto.
\item \TeX{} é um programa e
não uma aplicação.
\end{itemize}
```

O `\begin{itemize} ... \end{itemize}` significa que nós queremos uma lista com um ponto em cada item. E cada item da lista é especificado por um comando `\item` .
Também é possível colocar uma lista dentro de outra. Por exemplo:

Devemos ter em mente quando usamos o TEX

- TEX é uma linguagem tipográfica e não um editor de texto.
- TEX é um programa e não uma aplicação.
- TEX é uma escolha natural para uma dessas situações
  - Se você quiser tipografar um documento contendo fórmulas matemáticas.
  - Se você quiser deixar um documento bonito.

```
Devemos ter em mente quando usamos o
\TeX
\begin{itemize}
  \item \TeX{} é uma linguagem
tipográfica e não um editor de texto.
  \item \TeX{} é um programa e não
uma aplicação.
  \item \TeX{} é uma escolha natural
para uma dessas situações
  \begin{itemize}
    \item Se você quiser tipografar
     um documento contendo fórmulas
     matemáticas.
    \item Se você quiser deixar um
     documento bonito.
    \end{itemize}
\end{itemize}
```

O ambiente `itemize` suporta até quatro níveis de aninhamento.

Enumerate é utilizado quando a ordem dos itens numa lista é importante, precisamos de uma lista que especifique a ordem. Por exemplo:

Os três passos para produzir e ver um documento em LATEX são:

1. Escrever um código-fonte com a extensão tex.
2. Compilar isso com o LATEX para produzir um arquivo dvi.
3. Ver o documento usando um driver dvi.
4. Converter para pdf, formato mais utilizado.

```
Os três passos para produzir e ver
um documento em \LaTeX{} são:
\begin{enumerate}
  \item Escrever um código-fonte
   com a extensão tex.
  \item Compilar isso com o
   \LaTeX{} para produzir um
   arquivo dvi.
  \item Ver o documento usando um
   driver dvi.
  \item Converter para pdf, formato
   mais utilizado.
\end{enumerate}
```

Assim como no ambiente itemize, existem quatro níveis de rótulos.

Description é o terceiro tipo de lista no LaTeX, que é usada para escrever algo assim:

Vamos dar uma olhada no que aprendemos

TeX Uma linguagem tipográfica.

Gedit Um editor de texto.

```
\begin{description}
  \item[\TeX] Uma linguagem tipográfica.
  \item[Gedit] Um editor de texto.
\end{description}
```

Você deve ter notado, que esse ambiente não produz rótulos para os itens, mas é possível criar rótulos, só é colocar colchetes logo após cada `\item`. O código acima mostra isso.

1.2.4 Notas

Podemos colocar notas no rodapé em nosso documento ou então notas marginais. Essas duas tarefas pdoem ser realizadas de maneira bem simples. A primeira pode ser feita com o comando `\footnote`, que apresenta um índice acima da palavra que indicará a nota.

Aqui vai o texto[1].

```
Aqui vai o texto.\footnote{e aqui a nota}
```

[1] e aqui a nota

Já para notas marginais, temos o comando `\marginpar`, que adiciona uma nota que inicia ao lado do texto onde o camando foi apresentado

Aqui vai o texto dentro do parágrafo normal

e aqui as notas marginais

```
Aqui vai o texto\marginpar{e aqui
as notas marginais} dentro do
parágrafo normal
```

1.2.5 Citações

Também de maneira bem simples, podemos fazer citações em nosso texto, escrevendo dentro do ambiente `quote`.

Então as frases foram estas:

> *“e o vento levou”*
>
> *“tudo que sobe tem que descer”*

```
Então as frases foram estas:
\begin{quote}
\enph{‘‘e o vento levou’’\\
‘‘tudo que sobe tem que descer’’}
\end{quote}
```

Para citações mais longas pode ser usado o ambiente *quotation* que possibilita a utilização de parágrafos indentados, já que *quote* alinha tudo na mesma posição na esquerda.

1.3 Fontes e Tamanhos

As letras e símbolos (coletivamente chamado tipo) que o LaTeX produz são caracterizadas pelos seus estilos e tamanhos. Nós podemos produzir pequenos e grandes tipos. O conjunto de tipos de um estilo e tamanho particulares, é chamado de *fonte*.

No LaTeX, um estilo é especificado pela família, série e forma. Um exemplo é comando `\textit` que produz a família romano, série média, forma itálica. A seguir mostramos uma tabela com as famílias, séries e formas, que podem ser combinadas para produzir diferentes fontes.

	estilo	comando
	romano	`\textrm{romano}`
Família	sans serif	`\textsf{sans serif}`
	`máquina de escrever`	`\texttt{máquina de escrever}`
	médio	`\textmd{médio}`
Série	**negrito**	`\textbf{negrito}`
	vertical	`\textup{vertical}`
Forma	*itálico*	`\textit{itálico}`
	inclinado	`\textsl{inclinado}`
	CAIXA ALTA	`\textsc{caixa alta}`

Tradicionalmente, o tamanho de uma palavra é medido em pt. O tamanho padrão que o TeX produz é 10 pt. Existem algumas declarações para mudar o tamanho, elas se encontram na tabela a seguir:

tamanho	`{\tiny{tamanho}}`	tamanho	`{\large{tamanho}}`
tamanho	`{\scriptsize{tamanho}}`	tamanho	`{\Large{tamanho}}`
tamanho	`{\footnotesize{tamanho}}`	tamanho	`{\LARGE{tamanho}}`
tamanho	`{\small{tamanho}}`	tamanho	`{\huge{tamanho}}`
tamanho	`{\normalsize{tamanho}}`	tamanho	`{\Huge{tamanho}}`

2 A Verdadeira Beleza do LaTeX

Apesar do LaTeX ser ótimo para a geração de textos sua característica mais marcante é a possibiliadde de utilizar notações matemáticas diversas, com versatilidade e precisão. Neste capítulo iremos explorar as possibilidades oferecidas pelo modo matemático do LaTeX, assim como a geração de tabelas e inserção de imagens.

2.1 Modo Matemático

O LaTeX é um ótimo editor de fórmulas matemáticas, onde podemos escrever frações, raízes, logaritmos e muito mais... Para adicionar uma fórmula matemática em uma frase, ela deve-ser inserida dentro do ambiente matemático, que é iniciado e terminado através do símbolo $[1]. Para exemplificar, veremos uma linha que usa o modo matemático e os comandos que a geram:

Seja f a função definida por $f(x) = 2x + 1$, e x um número positivo real.

```
Seja $f$ a função definida
por $f(x)=2x+1$, e $x$ um
número positivo real.
```

Seja f a função definida por f(x)=2x+1, e x um número positivo real.

```
Seja f a função definida
por f(x)=2x+1, e x um
número positivo real.
```

Perceba que as letras f e x aparecem escritas com um padrão diferente de fonte que lembra o itálico, mas não é. É para manter esse padrão que deve-se utilizar sempre o modo matemático ao adicionar conteúdo matemático, mesmo que seja apenas uma letra. Um detalhe importante é que o símbolo $ é usado para iserir alguma notação matemática no meio do texto, de modo que ela fique comprimida para não poluir e não aumentar o tamanho da linha. Quando se deseja inserir uma equação matemática destacada, podemos usar o ambiente `equation`, que geram o resultado a seguir:

[1] veja comandos equivalentes na página 23.

$$\int 2f(x)\partial x \qquad (2.1)$$

```
\begin{equation}
\int 2f(x)\partial x
\end{equation}
```

Uma outra vantagem oferecida pelo ambiente `equation` é a possibilidade de criar *labels* para as equações, para poder referênciá-las no futuro. Por exemplo: Ao usarmos os comandos

$$f(x) = 2x^2 - 2x + 5 \qquad (2.2)$$

```
\begin{equation}\label{eq1}
f(x) = 2x^2 - 2x + 5
\end{equation}
```

O nome *eq1* fica associado à equação (2.2) e então poderemos fazer referências à equação (2.2) em qualquer ponto do documento através do comando `(\ref{eq1})`.

Além dessas duas possibilidades existe o ambiente `eqnarray`, que é útil quando se deseja escrever uma sequência de várias equações com referências individuais:

$$\begin{aligned} f(x,y) &= 3(x+2y) && (2.3)\\ &= 3x+6y && (2.4) \end{aligned}$$

```
\begin{eqnarray}
f(x,y)&=& 3(x+2y) \label{ex1}\\
&=& 3x+6y \label{ex2}
\end{eqnarray}
```

Um par de símbolos & é usado, geralmente envolvendo o símbolo de igualdade, para indicar qual deve ser a centralização da equação (O que estiver entre '&' será colocado no centro da página) e como o *eqnarray* não adiciona quebras de linha automaticamente, devemos indicar a posição das quebras através de duas barras invertidas `\\`.

É provavel que você já tenha notado que quando escrevemos no modo matemático, todos os nossos espaços são ignorados, inclusive espaços individuais. Essa é uma das características desse modo no LaTeX. Há também uma diferença na forma de apresentação das fórmulas quando utilizamos equações destacadas, com `equation` ou dentro do texto com o `$`. Um outro tratamento diferenciado, é que LaTeX considera todas as letras como incógnitas e assim, um texto em um ambiente matemático deve ser explicitado com o uso do `\mbox`. Há também modos diferentes de declararmos um amiente matemático, que nos apresenta algumas dierenças. Vejamos então na sequência estas questões.

2.1.1 Escrevendo fórmulas matemáticas

Com os exemplos vitos até aqui, podemos destacar algumas diferenças entre eles. Fórmulas dentro de frases, formulas destacadas e também fórmulas destacadas sem uma numeração de

referência, que ainda não vimos.

Fórmulas em frases são escritas internas a uma par de `$`. Há outras maneiras de se obter resultados equivalentes. Uma é através dos patenteses precedidos de contrabarras `\(` *fórmula* `\)` e outa com o ambiente `math`. Quanto utilizar um ambiente para fórmulas, tenha atenção em não deixar linhas em branco antes e nem após a declaração, pois assim LaTeX deixará um espaço maior para esta fórmula e também haverá um parágrafo, muitas vezes indesejado, na linha posterior.

Uma função $f(x) = y$ também pode ser escrita como $f(x) = y$ ou ainda $f(x) = y$.

```
Uma função $f(x)= y$ também pode
ser escrita como \(f(x)= y\) ou
ainda \begin{math}f(x)= y\end{math}.
```

Fórmulas destacadas são escritas dentro do ambiente `equation`. Essas fórmulas serão apresentadas com um número para sua referência dentro do texto. Muitas vezes não é necessário sua refererenciação. Para que não apareça ao lado da fórmula esse número de referência, usamos o ambiente `displaymath` ou as escrevemos entre colchetes precedidos de contrabarras `\[` *fórmula* `\]`.

Ainda podemos escrever uma sequência de fórmulas e referenciá-las com o ambiente `eqnarray`. Equivalente a `displaymath`, podemos escrever uma sequência de fórmulas sem numeração com a forma variada `eqnarray*`.

Dada a função

$$f(x) = f(x-1) + f(x-2) \qquad (2.5)$$

com os valores iniciais

$$f(0) = 0$$
$$f(1) = 1$$

temos que (2.5)...

```
Dada a função
\begin{equation}\label{chave}
 f(x) = f(x-1) + f(x-2)
\end{equation}
com os valores iniciais
\begin{eqnarray*}
f(0) = 0 \\
f(1) = 1
\end{eqnarray*}
temos que (\ref{chave})...
```

Delimitadores quando estamos escrevendo fórmulas matemáticas muitas vezes são necessários, como por exemplo, para indicar precedência de operações. Mas muitas vezes o conteúdo de uma equação abrange mais de uma linha, e este delimitador, como um parente-

ses não envolve toda a equação, dando um especto nada agradável. Para resolver esse tipo de problema, existem os comandos do LaTeX, `\left` e `\right`, que fazem com que um delimitador tenha seu tamanho adaptado ao tamanho da fórmula. Isso é útil também para gerar a notação de módulo, como veremos em um dos exemplos abaixo.

$$\left(\frac{3x+2}{\frac{y+2}{x-1}}\right) \tag{2.6}$$

$$\left|(3x+y)^2\right| \tag{2.7}$$

```
\begin{eqnarray}
\left(
\frac{3x+2}{\frac{y+2}{x-1}}
\right)\\
\left| (3x+y)^2 \right|
\end{eqnarray}
```

Mas algumas vezes não precisamos de que um delimitador esteja em ambos os lados da fórmula, e para indicar que ele não deve aparecer, completamos seu equivalente com um ponto.

$$f(x) = \begin{cases} 1 & \text{, se } x \leq 0. \\ f(x-1)+k & \text{, se } x > 0. \end{cases}$$

```
\[ f(x)=\left\{ \begin{array}{cl}
      1, & \mbox{ se $x \leq 0$.} \\
      f(x-1)+k & \mbox{,{ se $x > 0$.
    \end{array} \right. \]
```

Note que para utilizar a { como delimitador, sendo ela um caracter espcial em LaTeX temos que escrevê-la como `\{`.

Matrizes O ambiente de matriz do LaTeX é bastante simples de se usar, ele lembra bastante o formato do `tabular` mas ao invés de termos de especificar o formato desejado, ele nos fornece um formato padrão que permite até 10 números centralizados.

Os exemplos abaixo mostram as variações `matrix, pmatrix, bmatrix, vmatrix` e `Vmatrix`, e para usar estes ambientes é necesário incluir o pacote `amsmath` no preâmbulo.

$$\begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix} \tag{2.8}$$

```
\begin{equation}
\begin{matrix}
0 & 1\\ 1 & 0
\end{matrix}
\end{equation}
```

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad (2.9)$$

```
\begin{equation}
\begin{pmatrix}
0 & 1\\ 1 & 0
\end{pmatrix}
\end{equation}
```

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (2.10)$$

```
\begin{equation}
\begin{bmatrix}
0 & 1\\ 1 & 0
\end{bmatrix}
\end{equation}
```

$$\begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix} \quad (2.11)$$

```
\begin{equation}
\begin{vmatrix}
0 & 1\\ 1 & 0
\end{vmatrix}
\end{equation}
```

$$\begin{Vmatrix} 0 & 1 \\ 1 & 0 \end{Vmatrix} \quad (2.12)$$

```
\begin{equation}
\begin{Vmatrix}
0 & 1\\ 1 & 0
\end{Vmatrix}
\end{equation}
```

2.1.2 Alguns comandos importantes

Ao escrever conteúdo matemático os seguintes comandos são os que aparecem com maior frequência, e é importante conhecê-los.

Índices Para criar índices dentro de equações matemáticas o símbolo _ deve ser usado. Se o índice for composto de mais de um caractere, ele deve ser colocado entre chaves:

$$f(x_1, x_2) = x_1 x_2 + 2x_2 \quad (2.13)$$
$$g(x_{valor}) = 3x_{valor} + 2 \quad (2.14)$$

```
\begin{eqnarray}
f(x_1,x_2) &=& x_1 x_2 + 2x_2 \\
g(x_{valor}) &=& 3x_{valor} + 2
\end{eqnarray}
```

Expoentes Seu uso é idêntico ao símbolo para gerar índices, mas ao invés de um _, esse comando utiliza o símbolo ^:

$$f(x,y) = x^2+3xy+y^4 \quad (2.15)$$
$$g(x,y) = 3x^{4-y}+2 \quad (2.16)$$

```
\begin{eqnarray}
f(x,y) &=& x^2+3xy+y^4 \\
g(x,y) &=& 3x^{4-y} + 2
\end{eqnarray}
```

Frações O LaTeX tem um comando específico para criação de frações, que é o \frac. Seu uso é o seguinte: \frac{numerador}{denominador}. Por exemplo:

$$\frac{2x+3}{2y-x} \quad (2.17)$$

```
\begin{equation}
  \frac{2x+3}{2y-x}
\end{equation}
```

Raízes O comando para gerar raízes é o \sqrt{valor}, e é possível usar um parâmetro adicional, entre colchetes, para indicar qual será o índice da raiz.

$$\sqrt{2y-x} \quad (2.18)$$
$$\sqrt[n]{2y-x} \quad (2.19)$$

```
\begin{eqnarray}
  \sqrt{2y-x} \\ \sqrt[n]{2y-x}
\end{eqnarray}
```

Somatórios Mesmo sabendo que o somatório nada mais que é que um sigma, utilizar a letra grega com índice e expoente não gera o mesmo resultado que o comando \sum, como veremos no exemplo abaixo.

O comando \sum gera o seguinte resultado:

$$\sum_{i=0}^{n} 2i^2 \quad (2.20)$$

```
\begin{equation}
\sum^n_{i=0} 2i^2
\end{equation}
```

Enquanto uma letra sigma com índice e expoente resulta no seguinte:

$$\Sigma_{i=0}^{n} 2i^2 \quad (2.21)$$

```
\begin{equation}
\Sigma^n_{i=0} 2i^2
\end{equation}
```

Produtórios Assim como somatórios, existe um comando para gerar produtórios, que gera um resultado final diferente do obtido através de um pi com índice e expoente. O comando `\prod` gera o seguinte resultado:

$$\prod_{i=0}^{n} i+1 \qquad (2.22)$$

```
\begin{equation}
\prod^n_{i=0} i+1
\end{equation}
```

Enquanto uma letra pi com índice e expoente resulta no seguinte:

$$\Pi_{i=0}^{n} i+1 \qquad (2.23)$$

```
\begin{equation}
\Pi^n_{i=0} i+1
\end{equation}
```

Limites Para escrever a notação de limites, é necessário combinar dois comandos. Primeiro usamos o \lim, para gerar o texto *lim* com o padrão correto de fonte, para destacá-lo da fonte padrão do modo matemático. O segundo comando é o \to, que gera o símbolo $\to$.

Exemplo:

$$\lim_{h \to 0} 2^h \qquad (2.24)$$

```
\begin{equation}
\lim_{h \to 0} 2^h
\end{equation}
```

Derivadas Assim como limites, para escrever derivadas é preciso combinar dois comandos diferentes. Primeiro precisamos do \frac, e então do comando \partial, assim:

$$\frac{\partial f(x)}{\partial x} \qquad (2.25)$$

```
\begin{equation}
\frac{\partial f(x)}{\partial x}
\end{equation}
```

Integrais Finalmente, vamos ao comando \int, que é responsável por gerar o símbolo de integral. Podemos também gerar uma integral definida aliando o uso de índice e expoente ao uso do \int.

$$\int \quad f(x)dx \qquad (2.26)$$

$$\int_{-\infty}^{\infty} f(x)\,dx \qquad (2.27)$$

```
\begin{eqnarray}
& \int& f(x) dx \\
& {\displaystyle
\int^{\infty}_{ - \infty}}\!\! & f(x)\, dx
\end{eqnarray}
```

Observe os comandos de espaçamento `\,`, que acrescenta um espaço sutil e o `\!` que retira um espaço equivalente ao `\,`. Note também o efeito de `displaystyle`.

Obviamente estes não são os únicos comandos matemáticos existentes no LaTeX, mas é inviável citar e exemplificar cada um deles. Para uma lista completa dos símbolos, veja o anexo ao final do livro.

2.2 Tabelas

Apesar desse ser considerado por muitos a parte mais complicada do LaTeX veremos agora que não existe nenhum mistério em criar tabelas usando essa ferramenta. Para começar, vamos ver como funciona o comando `tabular`.

Ao iniciarmos o ambiente `tabular` devemos especificar quantas colunas a tabela vai ter e como será o alinhamento do texto em cada uma dessas colunas. Essas informações são parte do comando que inicia a tabela. Assim, `\begin{tabular}{|c|c|c|c|}` estaremos iniciando uma tabela de 4 counas centralizadas (`c`), com uma linha separando cada coluna, inclusive nas bordas.

Cada | indica uma linha vertical presente na tabela, e cada letra indica como será o alinhamento do texto naquela coluna `c` para centralizado, `r` para direita e `l` para esquerda.

Agora que a estrutura da tabela foi definida, basta acrescentar o conteúdo. Para isso é preciso saber que o final de cada coluna deve ser indicado por um símbolo especial, o &. Além disso, ao fim de cada linha deve ser adicionado um par de barras invertidas para que ocorra uma quebra de linha, e as linhas horizontais devem ser adicionadas manualmente. Vamos adicionar algum conteúdo à tabela do exemplo anterior para ver como ela fica.

1	2	3	4
4	3	2	1
2	1	3	4

```
\begin{tabular}{|c|c|c|c|}
\hline
 1 & 2 & 3 & 4 \\
\hline
 4 & 3 & 2 & 1 \\
\hline
 2 & 1 & 3 & 4 \\
\hline
\end{tabular}
```

Muito simples, não? Seguindo essa estrutura podemos adicionar quantas linhas quisermos, basta lembrar de colocar os & e as quebras de linha. Para estruturas mais complexas, podemos usar o camando `\multcolumn`, que junta duas colunas de uma linha, e o comando `\cline`, que desenha uma linha parcial na tabela.

Tabela PET Computação		
Ano	Valor	
2007	baixo	alto
2010	alto	alto

```
\begin{tabular}{|c||cc|}
\hline
\multicolumn{3}{|c|}%
{Tabela PET Computação}\\
\cline{2-3}
Ano & \multicolumn{2}{|c|}{Valor}\\
\cline{2-3}
2007 & baixo & alto \\
\cline{2-3}
2010 & alto & alto
\hline
\end{tabular}
```

2.3 Inserindo Figuras no LaTeX

Pode ser inserido alguns formatos de figura no LaTeX, e para isso é preciso utilizar o comando `\includegraphics`. O formato de imagem mais comum de se inserir é a que tem o formato Encapsulated PostScript (*.eps*).

```
\usepackage[drive]{graphicx}
```

Os drives podem ser **dvips**—inclusão de imagens *.eps* e geração de documento *dvi* e *pdflatex*—inclusão de imagens *.jpg*, *.png* e geração de documento *pdf*. Lembrando que os drives são opcionais. Caso não seja especificado, ele é corretamente identificado, tanto usando o comando *latex* ou *pdflatex* para compilar seu documento.

Para inserir a figura no local desejado do documento, utiliza-se o comando `\includegraphics[opções` onde suas principais opções são:

heigth –altura da figura em unidades aceitas pelo LaTeX(cm, mm, pt, in).

totalheight -altura total da figura, em qualquer unidade do LaTeX.

width –largura da figura, em qualquer unidade do LaTeX.

scale –ampliação ou redução da figura.

angle –rotação da figura em graus e sentido anti-horário.

origin -especifica o ponto em que a figura será rotacionada.

Segue um exemplo do uso destas opções, cada opção pode ser colocada dentro do colchetes, sempre as separando com vírgulas.

```
\includegraphics[width=.95\textwidth,height=5cm]{nome_da_imagem.eps}
```

Observe o comando `\textwidth`, neste caso está sendo dito que a largura (width) da figura será de 95% da largura do texto, ou seja, vai abranger quase toda a extensão da página, um pouco a menos do que as duas margens. Se a imagem não estiver no formato *.eps*, utilizando o linux, você pode converter da seguinte maneira:

```
convert image.bmp image.eps
```

Há também a possibilidade de utilizar conversores disponíveis na internet e o GIMP (software livre).

O comando `\includegraphics` apenas inclui a figura no local com o tamanho desejado, no geral queremos colocar uma legenda e colocar um número para referênciá-la no sumário de figuras ou pelo texto, fazemos isso através de um ambiente de figura chamado figure. O ambiente figure é um ponto flutuante, isso quer dizer que ele não é colocado exatamente no mesmo local em relação ao *.tex*, quer dizer que ele flutua, mas por que ele flutuaria? Qual seria seu critério de flutuação? Por um acaso o LaTeX está de perseguição comigo e não quer colocar minha figura no local desejado? Contenha as emoções, o raciocínio é bem simples, se não couber uma figura em um lugar o LaTeX a coloca o mais próximo possível sem deixar o espaços em branco. Ele flutua de acordo com a ordem de prioridade que lhe for informada, usa-se h para aqui (here), t para topo (top) da página e b para parte de baixo da página (bottom), estas três letras podem ser colocadas na ordem desejada como opção do ambiente figure. Por fim, nem computador e nem linguagem perseguem ninguém, nós é que sempre nos sentimos perseguidos quando se torna um pouco complicado de fazer uma figura ficar parada e não conseguimos, mas calma, muita calma, se não couber em um espaço é por que não cabe e pronto, se couber é por que cabe (quase me sinto um gênio escrevendo isso). O comando a seguir mostra a colocação das sintaxes de posição, para a figura.

```
\begin{figure}[posição da figura]
```

```
\includegraphics{minhafig.eps}
\end{figure}
```

Onde “posição da figura” deve ser substituido por h, t e/ou b na ordem que se queira priorizar.

```
\begin{figure}[htb]
\centering \includegraphics[opções]{nome_da_figura}
\end{figure}
```

As opções de posicionamento de um objeto flutuante, são as seguintes:

- *h* posicionar aqui.
- *t* posicionar em cima.
- *b* posicionar em baixo.
- *p* posicionar em uma página só para objetos flutuantes.
- *!* desconsiderar os padrões para a inserção da figura.

2.3.1 Referenciar figuras

O seguinte comando serve para referenciar uma figura (dar nome a uma figura), então utilizamos `\label{nomechave}` dentro do ambiente figure.

```
\begin{figure}[ht]
\includegraphics[scale=0.5]{nome_da_figura}
\label{nome_da_figura}
\end{figure}
```

Depois disto você pode referenciar a figura no meio do texto, basta fazer como demonstrado.

Observe a figura (10) na página 12.

```
Observe a figura~\ref{nome_da_figura} na
página~\pageref{nome_da_figura}
```

2.3.2 Legenda das figuras

Além de tudo mais, gostamos que nossas figuras fiquem bem identificadas, assim o leitor pode saber com facilidade do que elas se tratam, seja olhando na lista de figuras ou mesmo lendo parte do texto, claro que figuras legíveis e bem construídas, junto de um texto bem elaborado

colaboram muito, mas aqui não estamos tão preocupados se você sabe ou não escrever, nossa preocupação maior é a respeito da formatação do seu texto.

Podemos colocar a legenda para cima ou para baixo da figura, para isso basta usar o comando `\caption{Descrição da figura}` para cima ou para baixo do comando `\includegraphics` dentro do ambiente figure, como nos exemplos que seguem.

```
\begin{figure}[htb]
\centering
\caption{Descrição da figura}
\includegraphics[opções]{nome_da_figura}
\end{figure}

\begin{figure}[htb]
\centering
\includegraphics[opções]{nome_da_figura}
\caption{descrição da figura}
\end{figure}
```

3 *Seguindo Normas*

Até aqui nos concentramos especialmente em como redigir as estruturas lógicas do trabalho, isto é, seu conteúdo. Sabemos que muitos documentos devem ser apresentados com alguns elementos que atendam a certas especificações, como por exemplo a apresentação de trabalhos acadêmicos. Dentre estes elementos, podemos citar capa, sumário, referências bibliográficas, dentre outros, que devem seguir muitas vezes uma forma bem definida. Neste capítulo será apresentado como produzir alguns dos principais componentes da estrutura de documentos ciêntíficos e na seção XX, apresentaremos a classe `abnt`, projeto ABNTEX, que atende às normas da ABNT.

3.1 Sumário

Um componente que requer um certo trabalho para sua contrução é o sumário. A menos que exista uma ferramenta específica do seu editor de documentos favorito, que o gere automaticamente por você, a elaboração de um sumário para um trabalho extenso não é brincadeira.

Em LATEX, existe um comando específico para a geração automática do sumário. Tudo o quê você precisa fazer é inserir a uma linha contendo o comando `\tableofcontents` posicionada no parte do seu documeto que você quer apresentar o sumário. É muito provável que será logo após o título, ou não muito longe dele.

Aqui vai uma dica importante sobre capítulos, seções, etc:sem numeração, aqueles escritos `\section*{...}`. Você vai perceber que eles não entram no sumário. Ao criar um capítulo chamado Introdução como apresentado abaixo, para incluí-lo no sumário será necessário o comando usar o comanod `\addcontentsline`, que insere no arquivo de extensão `.toc` como uma divisão no nível de um capítulo a linha contendo o título deste capítulo como na linha abaixo. `\addcontentsline{toc}{chapter}{Introdução}` Analogamente ao sumário, você também pode inserir uma lista de tabelas e uma lista de figuras com os comandos `\listoftables` e `\listoffigures`. Abaixo há um exemplo de um documento que cria uma lista de figuras, uma

lista de tabelas e um sumário.

```
\documentclass{book}
\usepackage[utf8]{inputenc}
\usepackage[brazil]{babel}

\begin{document}

\listoffigures
\listoftables
\tableofcontents

\capter*{Introdução}
\addcontentsline{toc}{chapter}{Introdução}

Aqui vai o conteúdo da introdução...

\chapter{Capitulo 1}

Este capítulo será o primeiro capítulo
numerado com o número 1.

\end{document}
```

Os arquivos com extensão `.toc`, `.lof`, e `.lot` são gerados ao compilar um documento, apenas quando os comandos vistos acima são utilizados neste documento.

3.2 Índice remissivo

Tão trabalhoso quanto o sumário, sem a utilização do LaTeX, é claro!, é a construção de um índice remissivo. Mas aqui a coisa não é tão complicada. A tarefa mais difícil é saber quais as entradas importantes para que tenhamos um bom índice, e a maneira correta de se fazer isto não é durante a elaboração do texto. Hã? como assim? sim, é isso mesmo. Tenha em mente que durante a elaboração do texto, não se pode listar todas as ocorrências do termo que vai no índice, mas sim seus conceitos principais. Mas você pode ir marcando todas essas entradas na elaboração do texto, para sua própria referência ao definir a versão final.

Em seu documento então, basta marcar as ocorrências dos termos a serem incluídos no

índice com o comando `\index`. Este comando apenas inclui o termo que vai em seu parâmetro no índice e não imprime a palavra em seu texto.

para incluir o termo gnu, e tambem kiwi.

Índice

gnu 16
kiwi 16

```
\begin{document}
\usepackage{makeidx}
\makeidx
\begin{document}

para incluir o termo gnu\index{gnu},
e tambem kiwi\intex{kiwi}.

\printindex
\end{document}
```

Note que embora a entrada esteja no comando, se deve escrevê-la para que ela também apareça no texto.

3.2.1 Construindo o índice remissivo

Existe uma ferramenta auxiliar para a construção do índice que é o `makeindex`. O `makeindex` vai ordenar as entradas do arquivo de extensão `.idx` e vai produzir um novo arquivo de extensão `.ind`. Este arquivo é que será incluído no seu índice. Afinal, para incluir o índice em seu documento, é através do comando `\printindex`, que deve estar na posição em que você quer apresentá-lo. Normalmte a última seção do documento. Se você tentou compilar seu documeto antes de passar por aqui, verá algumas mensagens de erro. Para que tudo isso funcione, você deve incluir o pacote `makeidx` e também o comando `\makeindex` no preâmbulo do seu documento, para que todas as entradas dos comandos `\index` sejam escritas no arquivo de extensão `.idx`, e aí sim rodar o `makeindex`. Vai lá a sequência:

```
latex documento.tex

makeindex documento

latex documento.tex
```

Algumas entradas podem ser subentradas, e então você deve especificá-las no comando `\index` com uma exclamação após a entrada principal, como visto no exemplo. LaTeX permite o máximo de três níveis para as entradas. `gnu\index{gnu!gnats} \index{alamo@álamo}`

Uma alternativa ao uso do `makeindex` é o ambiente `theindex`, porém todas as entradas do índice devem ser inseridas manualmente. Para cada ítem da entrada usamos o `\item` e subentrada `\subitem`.

Índice

gnu 16
    bad, 20
kiwi 16
    fruta, 42

```
\begin{theindex}
\item gnu 16
  \subitem bad, 20
\item kiwi
  \subitem fruta, 42
\end{theindex}
```

Para trabalhos com poucas entradas no índice pode ser útil, mas não há motivos para não utilizar o `makeindex`.

3.3 Bibliografias

As referências bibliográficas ou citações bibiográficas de um trabalho são um conjunto padronizado de elementos de uma obra escrita, como título, autor, editora, local de publicações e outros, retirados de um documento que permite sua identificação individual, e posteriormente as informações contidas no texto possam ser comprovadas, se necessário. Podemos incluir referências como notas de rodapé ou apresentá-las ao fim do trabalho em uma lista de referências. Para o segundo caso, já que o primeiro pode ser facilmente aplicado com o comado `\footnote`, o LaTeX, nos disponibiliza o ambiente `thebibliography` que possui um parâmetro obrigatório, que é o número máximo de referências da lista. Similarmente ao ambiente enumerate, cada referência é dada por um comando `\bibitem` que tem como parâmetro a chave para referênciação no texto

Como visto em [Gos99] concluímos que...

```
Como visto em~\cite{gos} concluímos
que...
```

Referências

...
[Gos99] M. Goossens. *The Latex Companion.*
...

```
\begin{thebibliography}[9]
...
\bibitem[Gos99]{gos} M. Goossens.
\emph{The Latex Companion.}
...
\end{thebibliograph}
```

O problema dessas duas formas de referência, é que não há um padrão em sua apresentação, de

mode que para cada item de rodapé ou na lista de referência, teremos que formatá-lo, um a um, de acordo com a norma requerida para o trabalho.

Contudo, apresentaremos a ferramenta BIBTEX que agiliza a inserção das referências. O BIBTEX busca os dados para gerá-las em um arquivo externo de extensão `.bib`, que deve conter as informações em um formato estruturado, onde cada unidade lógica de uma referência é preenchida separadamente. Esta é uma ferramenta muito importante para a padronização das referências de um trabalho e possibilitar a alteração de estilos, de acordo com especificações ou modelos requeridos, sem necessitar a alteração da formatação do texto, que como vimos acima, demanda muito trabalho braçal!

3.3.1 Usando o BIBTEX

BIBTEX trata-se de um compilador de texto de formato `.tex`, interagindo com o texto no formato `.bib`, este arquivo no formato `.bib` deve possuir informações específicas sobre os trabalhos a serem citados, de forma que possamos chamar cada citação por uma chave, da mesma maneira como visto para outros elementos, como tabelas euquações, etc, com o comando `\cite{chave}`. Abaixo apresentamos como deve ser escrito este arquivo com extensão `.bib`.

Assumindo então que estamos com as referências a mão, no arquivo chamado `minharef.bib`, (assumimos também que este arquivo está no mesmo diretório em que o arquivo tex compilado) incluímos o comando `\bibliography{minharef}` posicionada onde você pretende obter a lista de referências(nas normas da ABNT, a referência bibliográfica é sempre o primeiro elemento pós-textual). Note que não é preciso colocar a extensão do arquivo, ele automaticamente reconhece que este é um arquivo `.bib`.

Em conjunto com este comando, você deve especificar o estilo em que as referências serão apresentadas, com o comando `\bibliographystyle{acm}` que especifíca o estilo acm. Este comando é obrigatório quando se usa um arquivo bib para inserir referências, e por padrão, seu estilo no latex é `plain`. `\bibliographystyle{plain}`

```
apalike  phiaea  authordate1  amsplain  alpha
```

Além destes acima, outros estilos podem ser utilizados.[1]

[1] http://www.cs.stir.ac.uk/ kjt/software/latex/showbst.html
http://amath.colorado.edu/documentation/LaTeX/reference/faq/bibstyles.html

3.3.2 Formato BIBTEX

No arquivo `.bib` devemos ter apenas o registro de cada referência que possivelmente citaremos pelos nossos textos, cada tipo de referência tem um formato diferente e para atender os diferentes estilos de referências bibliográficas, a seguir damos alguns exemplos:

```
@Book{   chave,
 author = {},
 editor = {},
 title = {},
 publisher = {},
 year = {}       }

@Article{chave,
 author = {},
 title = {},
 journal = {},
 year = {}       }
```

Diferentemente do LATEX, BIBTEX não difere maiúsculas de minúsculas. O primeiro parêmetro da referência é a sua chave, ou apelido, que será usado em um comando `\cite{chave}` no texto. Cada campo é seguido de um =, com ou sem espaços em volta, e seu parâmetro pode estar entre chaves ou entre aspas dupla. Cada parâmetro é separado por uma vírgula, com ou sem espaços antes ou após a vírgula. Tudo isso é cercado por chaves ou parênteses.

Em cada tipo de referência, existem parâmetros que podem diferir, como visto no exemplo acima, e vale frisar que alguns deles são obrigatórios para seu tipo. No caso do exemplo todos são obrigatórios. Os campos `volume, series, address, edition, month, note` são os opcionais para o tipo `@book` e os campos `volume, number, pages, month, note` são opcionais para o tipo `@article`.

Compilando A sequencia para que as referências sejam aplicadas corretamente pelo latex é a dos seguintes comandos no terminal:

```
latex nomedoarquivo.tex
bibtex nomedoarquivo
latex nomedoarquivo.tex
latex nomedoarquivo.tex
```

Isso poderia ser feito com a mesma lógica em editores próprios para o LaTeX como no editor Kile ou TeXmaker.

3.4 Documentos nos padrões da ABNT

Muitos trabalhos em suas especificações requerem a apresentação de documentos nos padrões da ABNT. Aqui apresentamos uma nova classe de documentos LaTeX que nos permite gerar este tão desejado formato, a classe `abnt`[2]. Porém esta classe não é padrão nas distribuições do LaTeX, o que requer uma tarefa adicional para aqueles que não puderem compilar o primeiro exemplo abaixo.

3.4.1 Estrutura essencial

A maoir parte dos trabalhos acadêmicos possuem alguns elementos comuns, que estão presentes nas suas normas de especifícação. A seguir, veremos os principais elementos, e como os construímos com o LaTeX. Atenção a ordem de apresentação destes elementos em seu documento final.

Capa O primeiro elemento obrigatório é a capa, que deve conter: nome da instituicao, autor, titulo, local e ano. E não é difícil concluir do exemplo acima que o comando `\capa` contrói pra você uma capa que coincide com os abjetos requeridos em uma capa!

Folha de rosto Na sequência, temos a folha de rosto que além dos elementos contidos na capa, também exige uma nota indicando a natureza acadêmica do trabalho e nome do orientador (e co-orientador se houver). Da mesma maneira nos é intuitivo a finalidade do comando folha de rosto.

Sumário Note que quando utilizamos a classe de documentos abnt, podemos gerar o sumário automaticamente com o comado `\sumario`, ao invés de `\tableofcontents`, apesar de que em princípio o resultado é o mesmo. Uma diferença é que capítulos, seções, etc não numerados como `section*...` também são incluídos no sumário; em outras classes, este tipo de seção para ser apresentada no sumário deveria ser indicada explicitamente com o comanto `\addcontentsline`. Podemos descatar aqui uma opção dessa classe que é a opçao `normaltoc`,

[2] esta classe faz parte do projeto ABNTeX

para que as páginas não contenham a indicação p. precedendo sua numeração. Similarmente, a lista de figuras e lista de tabelas, são geradas a partir dos comandos `\listadefiguras` e `\listadetabelas` respectivamente!

Bom, se você compilou um documento abntex e pretende que este documento esteja de acordo com as normas da UFPR, estará se perguntando o porquê que estes elementos pré-textuais não estão de acordo com o manual da UFPR. Há uma grande discussão sobre as possíveis interpretações das normas da ABNT, e a apresentada neste manual, é um pouco mais específica. Vamos então modificar o documento para que este atenda ao manual. Calma, não é necessário alterar o texto, apenas inclua o pacote `abnt-UFPR` no preâmbulo do seu arquivo `.tex` e veja o resultado.

```
\usepackage{abnt-UFPR}
```

Após estes elementos pré-textuais, não nos carece muitos outros cuidados e podemos aplicar quaisquer estruturas vistas até aqui para apresentação do texto, tendo em vista, é claro, que algumas estruturas não convém com as normas, como dividir o texto em colulas ou notas marginais. Um outro ambiente que deve ser substituído é o quotation para citações longas por citacao.

```
\begin{citacao} ... \end{citacao}
```

Referências bibliográficas Por fim temos as referências bibliográficas como último elemento obrigatório em documentos. As citações podem ser feitas de duas maneiras: uma utiliza o sistema autor-data e a outra o sistema numérico. Apesar de não ser apresentado no exemplo acima, o seu uso é feito da mesma maneira como visto na seção (bibliografia). Temos que adicionar o pacote `abntcite` com a opção `abntcite[num]` ou `abntcite[alf]` e os estilos bibliográficos ou abnt-cite abnt-alf, correspondente as opções do pacote. Para fazer referência no texto, além de `\cite` pode-se usar `\citeonline` para citações diretas.

3.4.2 Outros elementos

Acima foram apresentados os elementos obrigatórios em todos os trabalhos acadêmicos, como teses monografias e dissertações, mas a classe abnt nos oferece um pouco mais de artifícios para escrevermos nosso documento.

Resumo O resumo pode ser redigido dentro do ambiente resumo, assim como o resumo em outra lingua no ambiente abstract. Para alterar a língua estrangeira, que por padrão está no

inglês, devemos refazer o comando `\ABNTabstractname` que contém a palavra resumo em língua estrangeira, `\renewcommand{\ABNTabstractname}{Resumem}` seria o sufiente para nosso resumo em espanhol, por exemplo.

Anexos e apêndice Similarmente ao comando `\apendix`, os comandos `\apendice` e `\anexo` devem ser declarados para que os próximos capítulos sejam apêndices ou anexos respectivamente. Aqui também podemos optar pora que não apareça a palavra Apêndice ou Anexo na declaração de seus títulos, acrescentando a opção anapnoname na declaração documentclass. Em algum trabalho, pode requerer uma destas opções.

Folha de aprovação Para facilitar a redigir a folha de aprovação, que é obrigatória em trabalhos de conclusão de curso, temos a mão o ambiente flolhadeaprovacao. Como as normas não dizem muito sobre o formato desta folha, você está livre para estruturá-la. Mas saiba que algumas informações são escenciais, como as informações contidas na folha de rosto. Para os nomes dos professores da banca, temos o comando `\assinatura` e abaixo mostramos um modelo de folha de aprovação.

Índice remissivo e glossário Para a construção do índice remissívo não há nenhuma modificação. Sua construção é da mensta forma como foi visto nos capítulos anteriores, mas com um resultado um pouco diferente.

3.4.3 Considerações finais sobre ABNTEX

Existem mais informações úteis a respeito de abntex, como por exemplo, se você achar necessário alterar a fonte do título dos capítulos, que se dá como indicado no exemplo abaixo, ou sobre outras opções da clase.

```
\renewcommand{\ABNTchapterfont}%
{\bfseries\sffamily\fontseries{sbc}\selectfont}
```

Para saber sobre essas e outras coisas mais, consulte a documentação online em sua instalação.

4 LaTeX Beamer

4.1 O que é Beamer?

Uma classe de documentos do LaTeX que serve especificamente para elaborar apresentações. Esta classe é cheia de riquezas quanto a efeitos de apresentação, para que estes efeitos apareçam devemos informar os códigos para cada efeito desejado e não são poucos. O padrão do Beamer é bastante limpo, como o LaTeX é uma ferramenta de edição de textos profissional, o formato padrão dele tem foco em ser informativo, com pouca variação de cor, justamente por que em uma apresentação o foco é no conteúdo. Quem assiste não conseguiria se concentrar no assunto se existir muito enfeite na tela, estes "frufrus" tiram completamente a atenção do assunto. Imaginem que terrível fazermos uma apresentação sobre dinâmicas de grupo e em cada transição de slides aparece um bonequinho fazendo graça, metade da apresentação seria um bonequinho e outra seriam as dinâmicas de grupo, pois bem, se ainda quiséssemos fazer estas coisas, mesmo com esta argumentação, no Beamer seria possível. Por uma ordem natural, tanto da confecção da apresentação quanto de importância, começaremos aqui, com o foco em fazer uma apresentação limpa. Só não se pode confundir, limpa não significa feia, muito pelo contrário, o tradicional é limpo e belo, mas sem enfeites desnecessários.

4.2 Como começar?

4.2.1 Preâmbulo

Comecemos pelo preâmbulo, um simples, quase vazio, só o essencial.

```
\documentclass{beamer}
\usepackage[utf8]{inputenc}
\usepackage[brazil]{babel}
```

Naturalmente muitos outros comandos podem ser colocados no preâmbulo, como os que já foram apresentados até aqui. Lembrando que a classe `beamer` já inclui pacotes como `graphics` e `amsmath`. Podemos também definir no preâmbulo o tema a ser usado no Beamer, como no comando seguinte:

```
\usetheme{Warsaw}
```

`Warsaw` é um dos muitos temas possíveis para a apresentação.

Na apresentação usualmente queremos iniciar com uma tela que identifique o autor, título e instituição, para isso os seguintes comandos são utilizados:

```
\title{Simple Beamer Class}
\author{myName}
\institute{German University in Cairo}
\date{\today}

\begin{document}

\frame{\titlepage}

\end{document}
```

4.2.2 Frames

Deve-se ter muito claro em mente que cada slide é um slide, esta é uma das dificuldades para quem está fazendo as primeiras apresentações com esta classe de dados, parece tão óbvio falado dessa forma, mas em se tratando de passar código para o computador os principiantes acabam cometendo alguns erros. Pois bem, se cada tela a ser apresentada é um slide, tem que ser dito isso para LaTeX, então temos os delimitadores de frames (cada frame é um slide). Dentro das definições de `\begin{document}` e `\end{document}`, pode-se utilizar duas maneiras de fazer *slides*:

```
% Forma completa para usar opções
\begin{frame}[opções]{Nome do Frame}

% Esta região é onde se coloca o conteúdo do slide
```

```
\end{frame}

% Forma simples e rápida, sem opções
\frame{ ...  }
```

Há ainda um tipo especial de *frame* que contém a *Table of Contents* (Tabela de Conteúdos), que pode ser usado da seguinte forma:

```
% Usando a forma simples de fazer frames:
\frame{ \tableofcontents }
```

No modo simples de se fazer *frames*, utiliza-se `\frametitle{}` para dar título ao *frame* atual. Já no modo completo, o nome é passado como parâmetro e é opcional. E as opções são `[fragile]` e `[plain]`. A diferença entre as duas opções é que, se você quiser usar o ambiente *verbatim* em um slide, deve-se usar a opção fragile. Este ambiente é utilizado para simular a digitação, e dentro dele, todos os caracteres são impressos, inclusivecaracteres especias e multiplos espaços.

4.2.3 Seções

Do mesmo modo que em um `article` ou `book`, há seções no `beamer`. O nome das seções são apresentados separadamente, dependendo do tema.

```
\section{Nome da Seção}
% Os frames estão contidos nas seções
\begin{frame}
  ...
\end{frame}
```

Assim como as seções, as subseções têm seu destaque de acordo com o tema.

```
\subsection{Nome da Subseção}
% Os frames podem estar contidos nas subseções e seções
 \begin{frame}
  ...
 \end{frame}
```

4.3 Fazendo uma apresentação

4.3.1 Overlays

Para dar a sensação de inserção de um novo item, figura ou texto no mesmo slide, utiliza-se o \pause.

Há também o \item<i-j>, onde o item (isso dentro do itemize!) vai aparecer nos *slides* **i** a **j**, começando a contagem a partir do *slide* atual.

Uma alternativa, se você quiser que apareça um item por vez, é usar o $[< +- >]$ no itemize:

```
\begin{itemize}[<+->]
   \item L
   \item A
   \item T
   \item E
   \item X
\end{itemize}
```

4.3.2 Uncover, only e invisible

Se quisermos um efeito semelhante ao do \pause, mas com um maior controle sobre o conteúdo que será omitido, temos os comandos uncover, only e invisible. Esses comandos oferecem a possibilidade de especificar um trecho de texto usando chaves, da seguinte forma:

```
\item Language used by Beamer: L\uncover<2->{A}TEX
\item Language used by Beamer: L\only<2->{A}TEX \invisible<1>{Texto
  invisível no {\it slide} 1}.
```

Nos dois primeiros comandos o número 2 colcado antes do texto entre chaves indica a partir de qual slide o conteúdo entre chaves se tornará visível, enquanto no \invisible o número entre os símbolos de maior e menor (¡ ¿) indica o oposto, isto é, em qual slide o texto entre chaves estará invisível.

4.4 Efeitos

4.4.1 Blocos de Texto

Existem três tipos de blocos:

```
\begin{block}{Título do bloco}
   Texto.
\end{block}

\begin{alertblock}{Título do bloco de alerta}
   Texto.
\end{alertblock}

\begin{exampleblock}{Título do bloco de exemplo}
   Texto.
\end{exampleblock}
```

4.4.2 Tabelas

Suponhamos a seguinte tabela:

```
\begin{tabular}{lcccc}
\hline
 Class & A & B & C & D  \pause \\
 X & 1 & 2 & 3 & 4 \pause \\
 Y & 3 & 4 & 5 & 6 \pause \\
 Z & 5 & 6 & 7 & 8 \\
\hline
\end{tabular}
```

Para dar o efeito dinâmico nas linhas, utilizamos o `\pause` em cada linha da tabela, antes de quebrá-la.

ANEXO A – Lista de símbolos matemáticos

Aqui estão alguns símbolos úteis para serem utilizados em fórmulas matemáticas. E para que seja possível incluí-los no documento, é necessário que o pacote `amssymb` esteja no preâmbulo do documento.

$\leq$	`\leq`	$\geq$	`\geq`	$\equiv$	`\equiv`	$\models$	`\models`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`	$\perp$	`\perp`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`	$\mid$	`\mid`
$\ll$	`\ll`	$\gg$	`\gg`	$\asymp$	`\asymp`	$\parallel$	`\parallel`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`	$\bowtie$	`\bowtie`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`	$\Join$	`\Join`[b]
$\sqsubset$	`\sqsubset`[b]	$\sqsupset$	`\sqsupset`[b]	$\neq$	`\neq`	$\smile$	`\smile`
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\doteq$	`\doteq`	$\frown$	`\frown`
$\in$	`\in`	$\ni$	`\ni`	$\propto$	`\propto`	$=$	`=`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$<$	`<`	$>$	`>`
$:$	`:`						