



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INTRODUÇÃO AO \LaTeX

Carlos Frederico Bastarz

Apostila de introdução à linguagem de marcação \LaTeX .

URL do documento original:

[<http://urlib.net/xx/yy>](http://urlib.net/xx/yy)

INPE
São José dos Campos
2019

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE

Gabinete do Diretor (GB)

Serviço de Informação e Documentação (SID)

Caixa Postal 515 - CEP 12.245-970

São José dos Campos - SP - Brasil

Tel.:(012) 3945-6923/6921

Fax: (012) 3945-6919

E-mail: pubtc@sid.inpe.br

**COMISSÃO DO CONSELHO DE EDITORAÇÃO E PRESERVAÇÃO
DA PRODUÇÃO INTELECTUAL DO INPE (DE/DIR-544):****Presidente:**

Marciana Leite Ribeiro - Serviço de Informação e Documentação (SID)

Membros:

Dr. Gerald Jean Francis Banon - Coordenação Observação da Terra (OBT)

Dr. Amauri Silva Montes - Coordenação Engenharia e Tecnologia Espaciais (ETE)

Dr. André de Castro Milone - Coordenação Ciências Espaciais e Atmosféricas (CEA)

Dr. Joaquim José Barroso de Castro - Centro de Tecnologias Espaciais (CTE)

Dr. Manoel Alonso Gan - Centro de Previsão de Tempo e Estudos Climáticos (CPT)

Drª Maria do Carmo de Andrade Nono - Conselho de Pós-Graduação

Dr. Plínio Carlos Alvalá - Centro de Ciência do Sistema Terrestre (CST)

BIBLIOTECA DIGITAL:

Dr. Gerald Jean Francis Banon - Coordenação de Observação da Terra (OBT)

Clayton Martins Pereira - Serviço de Informação e Documentação (SID)

REVISÃO E NORMALIZAÇÃO DOCUMENTÁRIA:

Simone Angélica Del Ducca Barbedo - Serviço de Informação e Documentação (SID)

Yolanda Ribeiro da Silva Souza - Serviço de Informação e Documentação (SID)

EDITORAÇÃO ELETRÔNICA:

Marcelo de Castro Pazos - Serviço de Informação e Documentação (SID)

André Luis Dias Fernandes - Serviço de Informação e Documentação (SID)



MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

INTRODUÇÃO AO \LaTeX

Carlos Frederico Bastarz

Apostila de introdução à linguagem de marcação \LaTeX .

URL do documento original:

[<http://urlib.net/xx/yy>](http://urlib.net/xx/yy)

INPE
São José dos Campos
2019



Esta obra foi licenciada sob uma [Licença Creative Commons Atribuição-NãoComercial 3.0 Não Adaptada](#).

This work is licensed under a [Creative Commons Attribution-NonCommercial 3.0 Unported License](#).

Informar aqui sobre marca registrada (a modificação desta linha deve ser feita no arquivo publicacao.tex).

*“The language in which we express our ideas has a strong influence
on our thought processes.”*

DONALD ERVIN KNUTH
em “*Literate Programming*”, 1992

Aos alunos, colaboradores e servidores do INPE.

AGRADECIMENTOS

Agradeço ao André Fernandes pelo convite, a Simone Del Ducca pelas sugestões e revisão do documento.

À Danusa Caramello pelo auxílio na organização dos documentos para a realização do curso.

Aos colegas do INPE que desenvolveram a ideia original e mantêm as versões do estilo do INPE, objeto deste material.

Ao pessoal do Sistema de Informação e Documentação (SID) do INPE pela organização do curso.

Às pessoas que participam dos fóruns da *internet*, especialmente o TeX *StackExchange*, que tem a paciência e a motivação necessários para discutir, colaborar e responder às perguntas sobre como fazer no LaTeX.

RESUMO

Este material apresenta a linguagem de marcação \LaTeX para a confecção de textos científicos, tendo como foco o estilo de publicações do INPE. São apresentados os aspectos históricos de formulação da linguagem e as motivações para o seu uso dentro do ambiente acadêmico. O objetivo principal do uso da linguagem é permitir que o usuário concentre-se na escrita do texto, no desenvolvimento das suas ideias sem ter que se preocupar com a determinação e o posicionamento dos diversos elementos estruturais de um documento. Não se trata, porém, de um curso de escrita científica, mas sim de um manual objetivo para a aplicação da linguagem de marcação \LaTeX utilizando, especialmente, o estilo de publicações do INPE. Ao consultar o conteúdo deste material, o usuário estará habilitado a aplicar a linguagem na elaboração e desenvolvimento dos seus trabalhos acadêmicos e científicos.

Palavras-chave: \LaTeX . Escrita Científica. Linguagem de Marcação.

INTRODUCTION TO L^AT_EX

ABSTRACT

This material presents the L^AT_EX markup language as a type writing tools focused at the INPE's style sheet. It is presented the historical aspects of the language formulation as well as the motivations for its use in the academia. The iams for the language usage is to allow the user to keep focused in the writing and the ideas development without the need to focus at textual elements. It is not a scientific writing course though, but a objective manual for the language application. At the end, it is expected that the user is able to apply the language markup system at the type writing of their own documents.

Keywords: L^AT_EX. Scientific Writing. Markup Language.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE EXERCÍCIOS

Pág.

SUMÁRIO

Pág.

1 Parte I - Preparação

1.1 Introdução

Metodologia científica compreende os conhecimentos necessários para a produção científica. Artigos, relatórios, dissertações e teses são documentos que trazem relatos de experimentações, muitas vezes práticas, que surgem da necessidade de se testar hipóteses. Estas hipóteses frequentemente se referem ao mundo físico em vivemos, mas podem também, serem formulados a cerca de ideias abstratas. O método científico, assim como preconizou o patrono de todas as ciências, Renè Descartes, em seu “O Discurso do Método” (??), representa uma sequência de etapas que visam testar as hipóteses que formulamos e transformá-las, então, em teses, em teorias.

A escrita é parte fundamental da metodologia científica. É através dela que documentamos todo o processo de desenvolvimento da ciência, é através dela que se faz a comunicação formal da ciência que se produz e que, finalmente, se materializa o conhecimento adquirido. A escrita científica deve ser pautada por normas que ajudam a verificar a natureza do que se escreve e a validade dos argumentos com que se trata o objeto de estudo.

O T_EX (pronuncia-se “Tec”, sendo o “X” ao final a letra χ do alfabeto Grego), foi criado pelo Matemático e Cientista da Computação americano Donald Ervin Knuth, em 1978 para facilitar a escrita e melhorar a apresentação de textos científicos, principalmente aqueles com notações matemáticas. Naquela época, não haviam editores de texto formatado, como por exemplo o *Microsoft Word*, *Libre Office* e outros. Estes software viriam a ser lançados apenas na década de 1990. Além disso, o computador pessoal estava em seus primórdios, e viria a se tornar realmente pessoal com o lançamento do Apple II, no final dos anos 1970. Este tipo de computador, incluindo os seus clones (i.e., demais computadores que possuíam o mesmo *layout* de processador e memória de acesso randômico) não possuíam interfaces gráficas, mouses e discos rígidos; eles possuíam apenas um compilador BASIC e tela monocromática. Tendo-se em vista este cenário, a produção científica já havia avançado, pois o computador sempre foi uma ferramenta essencial nas mais diversas áreas do conhecimento (imagine como se comunicava a ciência antes do advento dos computadores, ou mesmo antes da invenção da prensa!).

Por outro lado, com os avanços tecnológicos e a sofisticação dos computadores pessoais, houve também a necessidade de se melhorar a representação tipográfica dos textos científicos, além da qualidade gráfica de imagens e gráficos. Em 1986, Leslie

Lamport lança a primeira versão do \LaTeX , uma versão aprimorada e de mais fácil utilização do que o \TeX puro. De forma geral, as linguagens de marcação de texto como o \TeX e o \LaTeX

É fato que a confecção de documentos utilizando o \LaTeX pode ser um pouco trabalhosa, visto que a linguagem é focada na escrita, e não na formatação como é o caso das suítes de escritório como o *Microsoft Office*. Neste caso, o usuário deve ponderar sobre o tipo de documento que tem por intenção produzir e o tamanho que este virá a ter. Portanto, a escrita de um documento utilizando o \LaTeX é vantajosa quando: 1) um estilo já está preparado (e.g., um artigo científico, dissertação, tese, relatório etc); 2) quando muitos elementos textuais estiverem presentes (e.g., figuras, tabelas, referências cruzadas, quebras de seções etc); 3) quando se tem tempo suficiente. Entre estes três pontos, há que se ressaltar o tempo necessário para a escrita de documentos utilizando \LaTeX . Embora a linguagem permita que a escrita seja focada no conteúdo do texto (ao invés do seu aspecto), há uma curva de aprendizado e é bastante frequente o usuário da linguagem se encontrar em situações em que necessita criar uma tabela um pouco mais complicada, ou inserir um conjunto de figuras de uma forma diferente. Estas situações não são óbvias de serem resolvidas e o usuário precisa ou ler a documentação dos pacotes (o que acaba não sendo muito prático), ou recorre a Fóruns na internet para resolver seus problemas. Há vários fóruns na internet que são especializados na linguagem \LaTeX e são uma boa fonte para a solução de diversas dúvidas e problemas. Apesar disso, o efeito colateral disso é que o usuário acaba não aprendendo como usar efetivamente a linguagem, porque nunca está a par da documentação, tendo à mão apenas receitas prontas. Logo, esta apostila foi escrita como uma forma de orientar os usuários a adquirirem o mínimo de independência na utilização do \LaTeX .

O texto do documento está organizado em três partes: a “Parte I - Entendendo o \LaTeX ” foi preparada como um introdução à linguagem de marcação \LaTeX , a fim de facilitar o entendimento das funções e estruturas básicas da linguagem, que serão extensamente utilizadas nos estilos; a “Parte II - Estilos do INPE”, contém um manual de utilização dos estilos do INPE para a escrita de dissertações e teses. Se o leitor já possui alguma experiência com a linguagem de marcação \LaTeX , ela pode pular diretamente para esta parte. Na “Parte III - Pacote Beamer”, há a apresentação básica do pacote Beamer, frequentemente utilizado para a confecção de apresentações (no estilo do *Microsoft PowerPoint*) e pôsteres.

1.2 Objetivos


Nesta apostila são apresentados os conceitos fundamentais da linguagem de marcação \LaTeX , com especial atenção à utilização dos estilos do INPE para a escrita de dissertações e teses. Os objetivos específicos são:

- Apresentar a linguagem de marcação \LaTeX , acompanhado de um breve histórico do seu desenvolvimento;
- Mostrar ao usuário como instalar o compilador/interpretador da linguagem nas plataformas mais conhecidas;
- Apresentar ao usuário os conceitos fundamentais da linguagem, levando-o a ter independência na utilização dos estilos do INPE;
- Treinar o usuário na utilização dos estilos de INPE para a escrita de dissertações e teses.

1.3 Estrutura e Organização do Documento

Este documento foi preparado utilizando o estilo de teses e dissertações do INPE, com a finalidade de servir não apenas como uma manual de utilização do estilo, mas também como um documento simples que possa ser utilizado como uma referência no aprendizado da linguagem de marcação \LaTeX . Para cumprir com esta finalidade, ao longo das seções que se seguem, alguns elementos textuais foram incorporados para sinalizar instruções específicas, como comandos do ambiente *Shell* do Linux e dicas ou instruções sobre pontos específicos do que está sendo apresentado.

Dessa forma, dicas e observações são destacados da seguinte forma:



! Isto é uma dica ou uma observação!

De outra forma, comando que devem ser digitados em algum ambiente computacional específico, são destacados como:



```
$ echo ''Isto é um comando Shell do Linux/UNIX!''
```

Exemplos da linguagem são apresentados em uma caixa, contendo a grafia dos comandos e o seu resultado ame anexo. Exercícios são apresentados de forma seme-

lhante, mas com a diferença de que é apresentado um exemplo (eg., uma tabela) o qual o usuário deverá reproduzir em ambiente local ou online configurado para tal. As respostas dos exercícios são então apresentadas no Anexo ???. Ao longo do texto, o leitor irá notar que na maioria dos exemplos que contém algum tipo de texto, aparece um texto prolixo. Este texto é gerado automaticamente com o auxílio de um pacote específico (`lipsum`), e não faz referência a nenhum tipo de conhecimento ou conceitos. Em outros exemplo, a frase “*The quick brown fox jumps over the lazy dog*” é utilizada. Esta frase é um pangrama, e contém todas as letras do alfabeto da língua inglesa. Ela é frequentemente utilizada como prova tipográfica, o que permite verificar a renderização de todas as letras e do seu espaçamento em uma única frase, de acordo com o tipo de fonte utilizada.

Neste documento boa parte dos exemplos e trechos de código foi obtido da *internet* de outros tutoriais. Uma lista de sites em que os exemplos foram obtidos, pode ser encontrado no Anexo ??.

O documento está organizado em 4 partes. A Parte 1 trata da introdução e objetivos deste documento e da linguagem LaTeX. A Parte 2 apresentam uma introdução aos elementos e marcadores principais da linguagem. Ao final desta parte, o usuário deverá ser capaz de produzir documentos LaTeX simples, utilizando as classes mais comuns e os elementos textuais mais frequentes. Na Parte ??, é apresentado o estilo de INPE para a escrita de teses e dissertações. Ao final desta parte, o usuário deverá ser capaz de utilizar o estilo do INPE para a escrita de sua tese ou dissertação. É importante salientar, entretanto, que a Parte ?? requer o aprendizado do conteúdo da Parte 2. A Parte ?? apresenta o pacote Beamer, uma classe que pode ser utilizada para confeccionar apresentações digitais.

1.4 Preparação do Ambiente

O L^AT_EX é, essencialmente, um compilador/interpretador. Para a sua utilização, é necessário instalar ele no computador. Nas seções a seguir, é mostrado como instalar o L^AT_EX nos sistemas operacionais nos sistemas Windows, Linux e Mac OS. A utilização da linguagem pode ser feita de diversas formas, em linha de comando ou utilizando editores de texto puro ou ainda editores mais avançados do tipo *What You See Is What You Get* (WYSIWYG). A utilização da linguagem será vista nas seções mais adiante.

1.4.1 Escolhendo e instalando o compilador

Nas próximas seções, será mostrado como instalar e configurar o compilador/interpretador da linguagem.

Linux

Nos sistemas GNU Linux, a instalação dos pacotes do L^AT_EX é bastante simples, mas pode variar de acordo com a distribuição utilizada. Neste manual, serão abordadas as distribuições mais populares e que utilizam os sistemas de pacotes “apt” (Debian e derivados) e “dnf” (RedHat e derivados). A vantagem destes gerenciadores de pacotes está no fato de que eles resolvem automaticamente as dependências, i.e., eles são capazes de instalar outros pacotes que são necessários para o correto funcionamento do programa principal. Em outras distribuições o processo de instalação pode ser diferente ou mesmo envolvendo a instalação a partir dos códigos fontes dos pacotes.

O site oficial do L^AT_EX é o <https://www.latex-project.org/>. No Linux, a principal distribuição da linguagem é o pacote “texlive” (<https://www.tug.org/texlive/>). Para instalar o pacote no Debian, basta fazer:

```
$ sudo apt install texlive-full
```

No RedHat, basta fazer:

```
$ sudo dnf install texlive-scheme-full
```

! Mesmo instalando o pacote completo do “texlive”, é possível que outros pacotes precisem ser instalados depois.

Windows

No sistema operacional *Microsoft Windows*, a instalação do pacote texlive pode ser feita de forma convencional, através do instalador oficial da distribuição disponível em <http://mirror.ctan.org/systems/texlive/tlnet/install-tl-windows.exe> (a url indicada sempre aponta para o pacote mais recente). Após baixar o pacote, siga as instruções a seguir para completar a instalação.

! Outras informações sobre a instalação do LaTeX no sistema operacional *Windows*, podem ser encontradas no documento [oficial](#) do Serviço de Informação e Documentação (SID) do INPE.

MacOS

No MacOS, a forma mais simples de instalar o pacote do texlive, é a partir do instalador disponível em <http://tug.org/cgi-bin/mactex-download/MacTeX.pkg> (da mesma forma, este url sempre aponta para o pacote mais recente). Se você está habituado a utilizar algum tipo de gerenciador de pacote no MacOS, e.g., o “brew”, pode tentar também a instalação com os seguintes comandos:

```
$ brew install caskroom/cask/brew-cask
$ brew cask install mactex
$ brew cask install texmaker
```

2 Parte II - Entendendo o \LaTeX

2.1 Introdução ao \LaTeX

Com o compilador/interpretador do \LaTeX instalado no computador, vamos dar os primeiros passos no aprendizado da linguagem. Vale ressaltar que o objetivo não é aprender ou treinar de forma exaustiva a linguagem, mas levar o leitor a compreender como e quando utilizar a linguagem. Desse forma, nas seções a seguir, são os comandos e estruturas principais da linguagem que são mais frequentemente utilizados em geral.

Antes de iniciarmos com as estruturas textuais, é necessário compreender como a linguagem \LaTeX interpreta comandos. A escrita de um documento em linguagem \LaTeX , independente do tipo de editor utilizado (e.g., em linha de comando ou um editor do tipo WYSIWYG), o usuário estará sempre escrevendo o código fonte do que virá a ser o seu documento, no formato escolhido com suas tabelas, imagens, equações etc.

Ao longo das próximas seções, o usuário irá aprender sobre os diversos aspectos da linguagem com vários exemplos, que foram obtidos a partir de várias fontes disponíveis na internet, em fóruns de discussões sobre a linguagem e manuais, disponíveis em diferentes sites. Os exemplos dados são apresentados em caixas destacadas, em que os comando apresentados vem acompanhados dos seus respectivos resultados.

Além disso, ao final do Capítulo 2, há uma série de exercício que o usuário deve realizar a fim de fixar o aprendizado dos comandos aprendidos. Em cada exercício, o usuário deverá reproduzir um exemplo, cujas respostas (comandos \LaTeX utilizados) são apropriadamente apresentados em uma seção em anexo ao documento principal.

No trecho de código a seguir é mostrado um documento \LaTeX , escrito da forma mais simples:

Exemplo 2.1.1: Um documento LaTeX mínimo

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    \ return M
import numpy as np

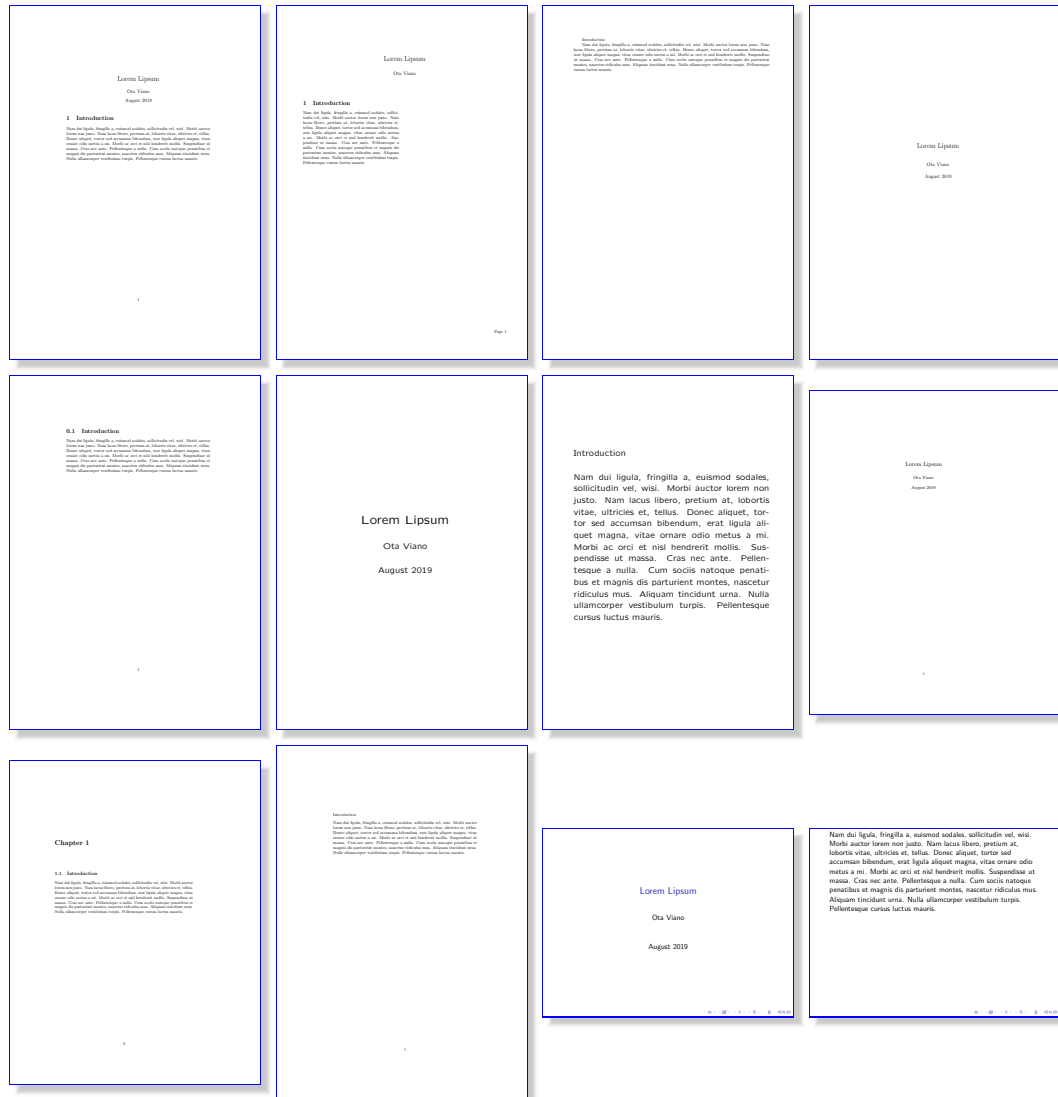
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
```

Exemplo 2.1.2: Um documento LaTeX mínimo

```
\backslash\mintinline{latex}{\documentclass{article}}\\  
\backslash\mintinline{latex}{\begin{document}}\\  
  
The quick brown fox jumps over the lazy dog.\\  
  
\backslash\mintinline{latex}{\end{document}}
```



No Exemplo 2.1.2 acima, observe que um documento LaTeX possui uma estrutura específica que se inicia com um descrição do tipo de documento dado pelo comando

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

(indicando que o documento tem o formato de *article*, i.e., um artigo). Tudo o que é escrito entre esta instrução e a próxima (**document**), é chamada de “preâmbulo”. Nesta seção podem ser carregados pacotes específicos da linguagem que permitem o uso de diferentes ambientes além de outros tipos de marcação. Em seguida, inicia-se o ambiente principal do


```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

LaTeX, que é o `return M` . Entre

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

as palavras reservadas    return M

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

e `return M` , o documento em si é escrito. Documentos LaTeX, independente da sua classe (que pode ser *book*, *report*, *article* e *letter*), podem ser muito simples ou complexos. O estilo para Teses e Dissertações do INPE (apresentado no Capítulo ??), é um exemplo de documento complexo que inclui estilo e formatação próprios.

Nas próximas seções, iremos tratar dos diversos marcadores que podem ser utilizados para alterar a aparência e o posicionamento dos textos e parágrafos.

2.1.1 Caracteres e símbolos especiais

No LaTeX há 10 tipos de caracteres especiais. São eles:

a) \	c) \$	e) &	g) _	i) }
b) #	d) %	f) ^	h) {	j) ~

Às vezes é necessário utilizá-los ao longo do texto, e então, faz necessário “escapá-los”. Há duas formas de fazer isso. 1) Escapando-os ou; 2) Utilizando comandos especiais.

Na primeira forma, basta utilizar a barra invertida “\”. Na segunda, pode-se utilizar comandos específicos do LaTeX.

Exemplo 2.1.3: Marcação para caracteres especiais

```
$\backslash$ \\  
\~{} \\  
\texttt{\~{}}
```

```
\  
^  
~
```

!

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M

```

No Exemplo 2.1.3, o marcador
pula uma linha.

2.1.2 Acentos

No LaTeX, os acentos podem ser escritos de forma literal, i.e., diretamente nas palavras sem a necessidade de marcadores especiais, desde que os pacotes necessários estejam carregados. O *babel* é um pacote do LaTeX que fornece os formatos de marcação e linguagem adequados para a acentuação de, por exemplo, caracteres

latinos acentuados. Para digitar acentos de forma natural, é necessário carregar os pacotes a seguir, no preâmbulo do documento:

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    •   return M
```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    • return M

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)


    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    • return M

```

 No estilo do INPE, os pacotes relacionados acima já estão pré-carregados.

Entretanto, em algumas situações é necessário marcar-se os acentos de forma explícita (e.g., na edição de um arquivo de referências do BibTeX).

No Exemplo 2.1.4 a seguir, são mostrados os acentos mais comuns.

Exemplo 2.1.4: Uso de acentos latinos no LaTeX

```
\'a \A \'e \E \'i \I \'o  
  \'O \'u \U  
  
\^a \^A \~a \~A \`a \`A \~o \~O  
  
\^e \^E \^o \^O  
  
\"u \"U  
  
\c{c} \c{C}
```

á Á é É í Í ó Ó ú Ú
â Â ã Ã à À ã Õ
ê Ê ô Ô
ü Ü
ç Ç



Outras marcações especiais para acentuação de caracteres podem ser obtidas em https://en.wikibooks.org/wiki/LaTeX/Special_Characters.

2.1.3 Tipos, tamanhos e estilos de letras

O texto básico pode ser marcado em estilos comuns, como o *itálico*, o sublinhado, o **negrito**, o texto ^{sobrescrito} e o texto _{subscrito}.

Exemplo 2.1.5: Estilos mais comuns em fontes

```
\textit{itálico} \\  
\textsl{itálico} \\  
\underline{sublinhado} \\  
\textbf{negrito} \\  
\textsuperscript{o}C \\  
H\textsubscript{2}O
```

itálico
itálico
sublinhado
negrito
°C
H₂O

No Exemplo 2.1.5, observe as diferenças entre o texto *itálico* produzido

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    com o marcador return M
    e o texto inclinado produzido pelo marcador

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

. No primeiro caso, as fontes produzidas são naturais, ou seja, há uma variação em *itálico* do tipo de fonte original. No segundo caso, a fonte original é renderizada a partir da inclinação da fonte natural.

! Dependendo do tipo de fonte utilizado, as diferenças entre os tipos *itálico* e *inclinado* podem ser mais evidentes. Veja mais no Exemplo 2.1.9.

Outros estilos também podem ser utilizados, mas dependem de outros

pacotes. Dois pacotes que fornecem estilos de marcações sobre as pa-
import numpy as np

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)
```

lavras, são o return M

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

e o `return M`. Para utilizá-los, deve-se antes carregar os pacotes necessários com os co-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

mandos          return M          e

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

```

Com o pacote `return M` , pode-se
riscar as palavras (forma mais comum).

Exemplo 2.1.6: Marcação com o pacote
import numpy as np

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M
```

\sout{palavra riscada}

~~palavra riscada~~

Exemplo 2.1.7: Marcação com o pacote `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

`\cancel{palavra cancelada} \\\`
`\bcancel{palavra cancelada} \\\`
`\xcancel{palavra cancelada} \\\`
`\cancelto{valor}{expressao}`

~~palavra cancelada~~
~~palavra cancelada~~
~~palavra cancelada~~
~~expressao~~ ^{valor}

No LaTeX, ao longo de um parágrafo, é possível alterar o tamanho da fonte. Por padrão há 10 tamanhos de letra que podem ser utilizados.

Exemplo 2.1.8: Tamanhos de fontes

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

Huge
huge
LARGE
Large
large
normalsize
small
footnotesize
scriptsize
tiny

Para alterar o tamanho de uma fonte localmente, basta fazer

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

No LaTeX é possível também alterar o tipo da fonte. Alguns estilos incluem fontes

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

```

no estilo

return M

(máquina de

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    escrever), return M (com serifa)

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

e    return M

```

(sem serifa).

Exemplo 2.1.9: Alguns tipos de fontes

```

\texttt{Typewriter Font} |
\texttt{\textit{Typewriter Font}} |
\texttt{\textsl{Typewriter Font}}

\textsf{Serif Font} |
\textsf{\textit{Serif Font}} |

```

```
\textsf{\textsl{Serif Font}}
```

```
\textrm{Roman Font} |
```

```
\textrm{\textit{Roman Font}} |
```

```
\textrm{\textsl{Roman Font}}
```

Typewriter Font | *Typewriter Font* | *Typewriter Font*

Serif Font | *Serif Font* | *Serif Font*

Roman Font | *Roman Font* | *Roman Font*

2.1.4 Títulos e seções

No LaTeX, é possível organizar o texto utilizando seções em até 7 níveis.

Na Seção 2.1 foram mostradas as diferentes classes padrão disponíveis para documentos LaTeX. Observe que as partes de conteúdo mar-


```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2== M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

cadas
como
return M
e

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

estão disponíveis

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    apenas para as classes    return M

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    e    return M

```

2.1.5 Cores e Paletas de Cores

As cores padrão que geralmente são utilizadas em um documento, e que não dependem de pacotes extras, são apresentadas a seguir.



Assim como em qualquer editor de textos WYSIWYG, as cores do texto podem ser alteradas para palavras isoladas, frases ou parágrafos.

Exemplo 2.1.10: Texto com fonte colorida

```
\textit{The \color{pink}{quick} \color{magenta}{brown} fox jumps
\color{green}{over} the lazy \color{blue}{dog}.}
```

The quick brown fox jumps over the lazy dog.

Além de modificar a cor das fontes, é possível também marcá-las de forma que o fundo fique colorido.

Exemplo 2.1.11: Texto com fundo colorido

```
\textit{The \colorbox{pink}{quick} \colorbox{magenta}{brown} fox
jumps \colorbox{green}{over} the lazy \colorbox{blue}{dog}.}
```

The quick brown fox jumps over the lazy dog.

Você pode escolher, por exemplo, utilizar a patela de cores do projeto “Solarized”. Para utilizá-la, basta carregar o pacote

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

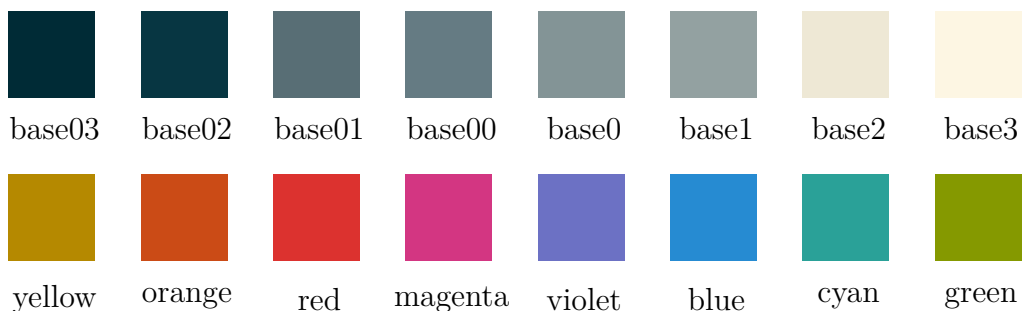
        VT = np.zeros((n*m,1), int)

    return M

```

. A paleta de cores

do “Solarized” é a seguinte:



Para utilizar as novas cores, basta utilizar um dos nomes definidos pela paleta, prece-
import numpy as np

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)
```

dido por return M . Por exemplo:

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

Outra paleta de cores harmoniozas, é provido pelo pacote `xcolor-material`. Esta é a paleta de cores do *Material Design* do Google. Para utilizá-la, basta carregar


```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

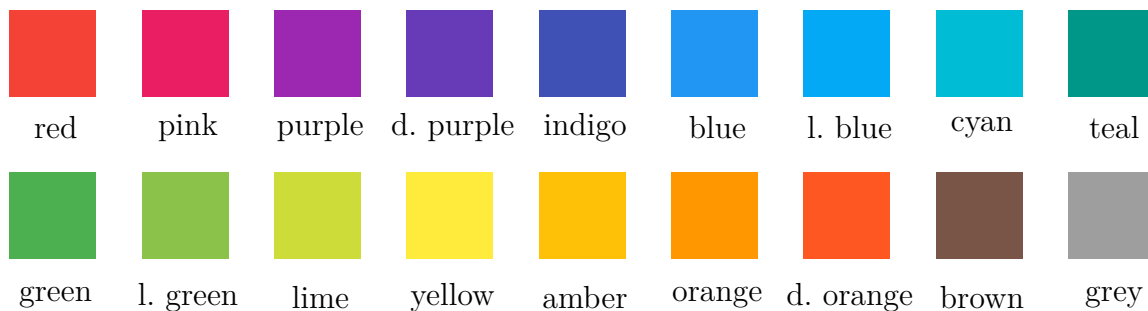
            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

o pacote      return M                               no preâmbulo
do documento.

```

As cores básicas do pacote xcolor-material são as seguintes (além do branco e preto):



Na paleta de cores mostrada acima, “d.” foi utilizado para abreviar a palavra *deep* e “l.” foi utilizada para abreviar a palavra *light*. Para utilizar as cores do pacote `xcolor-material`, é necessário utilizá-las da seguinte forma: a cor *Deep Purple* deve ser referenciada como “MaterialDeepPurple”, ou seja, a palavra reservada “Material” deve preceder o nome da cor, que por sua vez, deve ser indicada com a primeira letra em caixa alta.

! Pode ser necessário incluir o arquivo `xcolor-material.sty` à sua distribuição LaTeX. Veja a página do pacote para mais informações (<https://www.ctan.org/pkg/xcolor-material>).

Se for necessário, é possível definir qualquer cor utilizando códigos HTML, *Red Green Blue* (RGB) ou *Cyan Magenta Yellow Black* (CMYK) utilizando o comando

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

Exemplo 2.1.12: Definindo cores

```

\definecolor{meularanja1}{HTML}{FF7F00}
\definecolor{meularanja2}{rgb}{1,0.5,0}
\definecolor{meularanja3}{RGB}{255,127,0}
\definecolor{meularanja4}{cmyk}{0,0.5,1,0}

\begin{tikzpicture}

```

```

\fill [meularanja1] (0,0) rectangle ++(1.25,1.25);
\draw (0.6,-0.5) node {meularanja1};
\fill [meularanja2] (3,0) rectangle ++(1.25,1.25);
\draw (3.6,-0.5) node {meularanja2};
\fill [meularanja3] (6,0) rectangle ++(1.25,1.25);
\draw (6.6,-0.5) node {meularanja3};
\fill [meularanja4] (9,0) rectangle ++(1.25,1.25);
\draw (9.6,-0.5) node {meularanja4};
\end{tikzpicture}

```



meularanja1



meularanja2



meularanja3



meularanja4

! Veja outras opções de paletas e cores em [LaTeXColor](#).

2.1.6 Medidas

As medidas na linguagem LaTeX podem ser apresentadas em unidades diversas. Você pode misturá-las e isso pode ocorrer quando você reutiliza algum código que produz uma formatação específica que você gostaria de criar e acabou encontrando na internet. A Tabela 2.2 a seguir, mostra as unidades mais comuns. Vale ressaltar, entretanto, que a unidade padrão é o ponto e que o comprimento padrão é

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

Tabela 2.2 - Unidades de Medidas mais Comuns no LaTeX.

Abreviação	Definição	Valor em Po
Ponto	<pre> import numpy as np def incmatrix(genl1,genl2): m = len(genl1) n = len(genl2) M = None to become the incidence matrix VT = np.zeros((n*m,1), int) dummy variable compute the bitwise xor matrix M1 = bitxormatrix(genl1) M2 = np.triu(bitxormatrix(genl2),1) for i in range(m1): for j in range(i+1, m): [r,c] = np.where(M2 == M1[i,j]) for k in range(len(r)): VT[(i)*n + r[k]] = 1; VT[(i)*n + c[k]] = 1; VT[(j)*n + r[k]] = 1; VT[(j)*n + c[k]] = 1; if M is None: M = np.copy(VT) else: M = np.concatenate((M, VT), 1) VT = np.zeros((n*m,1), int) return M </pre>	1
Milímetro	<pre> import numpy as np def incmatrix(genl1,genl2): m = len(genl1) n = len(genl2) M = None to become the incidence matrix VT = np.zeros((n*m,1), int) dummy variable compute the bitwise xor matrix M1 = bitxormatrix(genl1) M2 = np.triu(bitxormatrix(genl2),1) for i in range(m1): for j in range(i+1, m): [r,c] = np.where(M2 == M1[i,j]) for k in range(len(r)): VT[(i)*n + r[k]] = 1; VT[(i)*n + c[k]] = 1; VT[(j)*n + r[k]] = 1; VT[(j)*n + c[k]] = 1; if M is None: M = np.copy(VT) else: M = np.concatenate((M, VT), 1) VT = np.zeros((n*m,1), int) return M </pre>	$2,84 = \frac{72}{25}$
	<pre> import numpy as np def incmatrix(genl1,genl2): m = len(genl1) n = len(genl2) M = None to become the incidence matrix VT = np.zeros((n*m,1), int) dummy variable compute the bitwise xor matrix M1 = bitxormatrix(genl1) M2 = np.triu(bitxormatrix(genl2),1) for i in range(m1): for j in range(i+1, m): [r,c] = np.where(M2 == M1[i,j]) for k in range(len(r)): VT[(i)*n + r[k]] = 1; VT[(i)*n + c[k]] = 1; VT[(j)*n + r[k]] = 1; VT[(j)*n + c[k]] = 1; if M is None: M = np.copy(VT) else: </pre>	

Em documentos escritos na linguagem LaTeX, é possível especificar as medidas utilizando os valores nas unidades indicadas na tabela acima e também utilizando algumas macros. Estas macros são, especificamente, representam algumas medidas padrão na linguagem e são apresentadas na Tabela [2.3](#) abaixo.

Tabela 2.3 - Algumas Macros de Medidas do LaTeX.

Macro	Descrição
<pre> import numpy as np def incmatrix(genl1,genl2): m = len(genl1) n = len(genl2) M = None to become the incidence matrix VT = np.zeros((n*m,1), int) dummy variable compute the bitwise xor matrix M1 = bitxormatrix(genl1) M2 = np.triu(bitxormatrix(genl2),1) for i in range(m1): for j in range(i+1, m): [r,c] = np.where(M2 == M1[i,j]) for k in range(len(r)): VT[(i)*n + r[k]] = 1; VT[(i)*n + c[k]] = 1; VT[(j)*n + r[k]] = 1; VT[(j)*n + c[k]] = 1; if M is None: M = np.copy(VT) else: M = np.concatenate((M, VT), 1) VT = np.zeros((n*m,1), int) return M </pre>	Distância vertical entre as linhas em um parágrafo
<pre> import numpy as np def incmatrix(genl1,genl2): m = len(genl1) n = len(genl2) M = None to become the incidence matrix VT = np.zeros((n*m,1), int) dummy variable compute the bitwise xor matrix M1 = bitxormatrix(genl1) M2 = np.triu(bitxormatrix(genl2),1) for i in range(m1): for j in range(i+1, m): [r,c] = np.where(M2 == M1[i,j]) for k in range(len(r)): VT[(i)*n + r[k]] = 1; VT[(i)*n + c[k]] = 1; </pre>	

No LaTeX, um comprimento é um número real seguido por uma unidade de medida, o qual pode ser modificado por um valor ou uma macro.



Veja mais detalhes, informações e exemplos em <https://en.wikibooks.org/wiki/LaTeX/Lengths>.

2.1.7 Parágrafos

Os parágrafos no LaTeX são blocos de texto separados por uma ou mais linhas. Para iniciar um parágrafo, basta pular uma linha. Uma outra forma de separar parágrafos, é através da utilização de duas barras invertidas

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

```

(return M). Observe as diferenças entre os exemplos a seguir.

Exemplo 2.1.13: Parágrafos Contíguos

```

\lipsumsentence[1-4]
\lipsumsentence[5-8]

```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu

libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem.

Exemplo 2.1.14: Parágrafos Separados por um Espaço

`\lipsumsentence[1-4]`

`\lipsumsentence[5-8]`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna.

Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem.

Exemplo 2.1.15: Parágrafos Separados por Duas Barras invertidas
import numpy as np

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    ( return M )
```

```
\lipsumsentence[1-4] \\  
\lipsumsentence[5-8]
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna.

Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem.

Posição e espaçamento

Boa parte dos elementos de um texto podem ser, basicamente, posicionados à esquerda, ao centro ou à direita. O LaTeX possui marcadores especiais para estes posicionamentos, que podem ser utilizados não apenas nos parágrafos, mas também com figuras e tabelas.

Exemplo 2.1.16: Parágrafos centralizados

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    Utilizando o ambiente return M
```

```
\begin{center}
\lipsumsentence[9-10] \\\
```

```
\lipsumsentence[11-12]
\end{center}
```

Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet
tortor gravida placerat.
Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel
leo ultrices bibendum.

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
```

```
    m = len(genl1)
```

```
    n = len(genl2)
```

```
    M = None to become the incidence matrix
```

```
    VT = np.zeros((n*m,1), int)    dummy variable
```

```
    compute the bitwise xor matrix
```

```
    M1 = bitxormatrix(genl1)
```

```
    M2 = np.triu(bitxormatrix(genl2),1)
```

```
    for i in range(m1):
```

```
        for j in range(i+1, m):
```

```
            [r,c] = np.where(M2 == M1[i,j])
```

```
            for k in range(len(r)):
```

```
                VT[(i)*n + r[k]] = 1;
```

```
                VT[(i)*n + c[k]] = 1;
```

```
                VT[(j)*n + r[k]] = 1;
```

```
                VT[(j)*n + c[k]] = 1;
```

```
            if M is None:
```

```
                M = np.copy(VT)
```

```
            else:
```

```
                M = np.concatenate((M, VT), 1)
```

```
    VT = np.zeros((n*m,1), int)
```

Além de utilizar o ambiente

```
    return M
```

,

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    é possível utilizar o marcador return M .

```

Exemplo 2.1.17: Parágrafos centralizados

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    Utilizando o marcador    return M
```

```
\centering
\lipsumsentence[13-14] \\\
\lipsumsentence[15-16]
```

Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla.
Curabitur auctor semper nulla. Donec varius orci eget risus.

Exemplo 2.1.18: Parágrafos alinhados à esquerda

```
\begin{flushleft}  
\lipsumsentence[17-18] \\  
\lipsumsentence[19-20]  
\end{flushleft}
```

Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi.

Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus.

Exemplo 2.1.19: Parágrafos alinhados à direita

```
\begin{flushright}  
\lipsumsentence[21-22] \\  
\lipsumsentence[23-24]  
\end{flushright}
```

Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis.
Suspendisse ut massa. Cras nec ante.

Espaçamentos horizontais e verticais são dados pelos marca-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

dores          return M          e

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

, respectivamente.

Exemplo 2.1.20: Espaçamento vertical

```

\lipsumsentence[21-22]
\vspace{1cm}
\lipsumsentence[23-27]

```

Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut

massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna.

Exemplo 2.1.21: Espaçamento horizontal

```
\hspace{2cm}\lipsumsentence[28-29] \\  
\lipsumsentence[30-31]
```

Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.
Nulla malesuada porttitor diam.
Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis.

Notas de rodapé

Notas de rodapé podem ser inseridas com o marcador `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

após a palavra a qual se quer referir. Nos exemplos a seguir, vamos usar o pangrama¹ “À noite, vovô Kowalsky vê o ímã cair no pé do pinguim queixoso e vovó põe açúcar no chá de tâmaras do jabuti feliz²”. O Exemplo 2.1.22 mostra como utilizar o marcador

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

Exemplo 2.1.22: Nota de rodapé utilizando o marcador

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

À noite, vovô Kowalsky\footnote{Esta é uma nota de rodapé.} vê o
 ímã cair no pé do pinguim queixoso e vovó põe açúcar no chá de
tâmaras do jabuti feliz\footnote{Este é uma outra nota de
rodapé}.

À noite, vovô Kowalsky^a vê o ímã cair no pé do pinguim queixoso e vovó põe açúcar
no chá de tâmaras do jabuti feliz^b.

^aEsta é uma nota de rodapé.

^bEste é uma outra nota de rodapé

2.1.22, foram incluídas duas notas de rodapé. Elas são ordenadas sequencialmente ao final da página em que foram inseridas.

Outra forma de incluir notas de rodapé, é a partir da utilização dos

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    marcadores          return M          e
```

¹Um pangrama é uma sentença que possui todas as letras do alfabeto.

²Este pangrama contém 90 caracteres e todas as letras acentuadas: à, á, â, é, ê, í, ó, ô, õ, ú e ç.


```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

O primeiro, insere o marcador na posição desejada, e o segundo, insere o texto referente àquele marcador. Esta forma é mais clara, pois destacam-se os comandos e marcadores fora do parágrafo que se está escrevendo, deixando-o mais limpo. Veja o Exemplo 2.1.23 a seguir:

Exemplo 2.1.23: Nota de rodapé utilizando os marcadores

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

e latex

À noite, vovô Kowalsky vê o ímã³ cair no pé do
pinguim queixoso⁴ e vovó põe açúcar no chá de
tâmaras do jabuti feliz.

`\footnotetext{Esta é uma nota de rodapé.}`

`\footnotetext{Esta é uma outra nota de rodapé.}`

À noite, vovô Kowalsky vê o ímã³ cair no pé do pinguim queixoso⁴ e vovó põe açúcar
no chá de tâmaras do jabuti feliz.

Esta é uma nota de rodapé.
Esta é uma outra nota de rodapé.

Um problema bastante comum com notas de rodapé no LaTeX ao se utilizar `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)
```

zarr os marcadores `return M` e

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)    dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

, é que os índices de
marcação podem se repetir nas notas de rodapé. Veja no Exemplo 2.1.24:

Exemplo 2.1.24: Nota de rodapé utilizando o marcado

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

À noite, vovô Kowalsky\footnote{Esta é uma nota de rodapé.} vê o
ímã cair no pé do pinguim queixoso e vovó põe açúcar no chá de
tâmaras do jabuti feliz\footnote{Este é uma outra nota de
rodapé}.

À noite, vovô Kowalsky^a vê o ímã cair no pé do pinguim queixoso e vovó põe açúcar
no chá de tâmaras do jabuti feliz^b.

^aEsta é uma nota de rodapé.

^bEste é uma outra nota de rodapé

2.1.22, observe que o estilo aplicado à nota de rodapé é alfabético. É possível alterar o estilo de numeração renovando o marcador `footnote`, e.g, `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

. Neste caso, a opção `roman` indica que o estilo de numeração dos índices dos marcadores será em algarismos romanos:

Exemplo 2.1.25: Nota de rodapé com referência numérica

À noite, vovô Kowalsky vê o ímã^{\footnotemark{}} cair no pé do pinguim queixoso^{\footnotemark{}} e vovó põe açúcar^{\footnote{Esta é mais uma nota de rodapé}} no chá de tâmaras do jabuti feliz.

^{\footnotetext{Esta é uma nota de rodapé.}}

^{\footnotetext{Esta é uma outra nota de rodapé.}}

À noite, vovô Kowalsky vê o ímã^v cair no pé do pinguim queixoso^{vi} e vovó põe açúcar^a no chá de tâmaras do jabuti feliz.

^aEsta é mais uma nota de rodapé

^aEsta é uma nota de rodapé.

^aEsta é uma outra nota de rodapé.

Listas

Listas ordenadas e não ordenadas podem ser facilmente criadas no LaTeX dentro de ambientes específicos. Listas não ordenadas são criadas den-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

tro do ambiente return M
e listas ordenadas são criadas dentro do ambiente

```



```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

No Exemplo 2.1.26, tem-se uma lista simples não ordenada.

Exemplo 2.1.26: Lista não ordenada utilizando o ambiente

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\begin{itemize}
  \item Item 1
  \item Item 2
  \item Item 3
\end{itemize}
```

-
- Item 1
 - Item 2
 - Item 3

Listas podem ser aninhadas, de forma que subitens possam ser obtidos. Observe no Exemplo 2.1.27 que o estilo dos subitens é alterado automaticamente:

Exemplo 2.1.27: Lista não ordenada aninhada utilizando o ambiente `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\begin{itemize}
  \item Item 1
  \begin{itemize}
    \item Item 1.1
    \item Item 1.2
  \end{itemize}
\end{itemize}
```

```

\item Item 2
\item Item 3
\begin{itemize}
  \item Item 3.1
  \item Item 3.2
  \item Item 3.3
\end{itemize}
\end{itemize}

```

- Item 1
 - Item 1.1
 - Item 1.2
- Item 2
- Item 3
 - Item 3.1
 - Item 3.2
 - Item 3.3

No Exemplo 2.1.28 a seguir, tem-se uma lista simples ordenada. Compare com o Exemplo 2.1.26 e veja a única diferença entre eles está apenas no tipo de ambiente utilizado (`itemize` e `enumerate`, respectivamente).

Exemplo 2.1.28: Lista ordenada utilizando o ambiente

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\begin{enumerate}
  \item Item 1
  \item Item 2
  \item Item 3
\end{enumerate}
```

a) Item 1

b) Item 2

c) Item 3

Assim como nas listas não ordenadas, listas ordenadas também podem ser aninhadas. Neste caso, observe que a numeração dos subitens é incrementada automaticamente:

Exemplo 2.1.29: Lista ordenada aninhada utilizando o ambiente

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\begin{enumerate}
  \item Item 1
\begin{enumerate}
```

```

\item Item 1.1
\begin{enumerate}
  \item Item 1.1.1
  \item Item 1.1.2
\end{enumerate}
\item Item 1.2
\end{enumerate}
\item Item 2
\item Item 3
\begin{enumerate}
  \item Item 3.1
  \begin{enumerate}
    \item Item 3.1.1
    \begin{enumerate}
      \item Item 3.1.1.1
      \item Item 3.1.1.2
    \end{enumerate}
    \item Item 3.1.2
  \end{enumerate}
  \item Item 3.2
\end{enumerate}
\end{enumerate}

```

-
- a) Item 1
 - Item 1.1
 - i. Item 1.1.1
 - ii. Item 1.1.2
 - Item 1.2
 - b) Item 2
 - c) Item 3
 - Item 3.1
 - i. Item 3.1.1
 - A. Item 3.1.1.1
 - B. Item 3.1.1.2
 - ii. Item 3.1.2
 - Item 3.2

Listas ordenadas podem ser organizadas de formas diferentes. Pode-se organizá-las de forma numérica, alfabética ou de forma alfanumérica. Para alterar a forma como as listas são ordenadas, é necessário definir o estilo de ordenamento com o código a seguir:

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)
```

mandando o código a seguir para o arquivo `incmatrix.py`, onde `<nível>` pode ser i, ii, iii ou vi. O estilo, dado pelo modificador `<estilo>`, pode assumir as seguintes opções:

- a) `alph` Letras minúsculas (a, b, c, ...);
- b) `Alph` Letras maiúsculas (A, B, C, ...);

- c) arabic Numerais arábicos (1, 2, 3, ...);
- d) roman Numerais minúsculos romanos (i, ii, iii, ...);
- e) Roman Numerais maiúsculos romanos (I, II, III, ...).

Combinando os estilos listados acima com os níveis, o comando `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

<estilo> pode as-

sumir algumas das seguintes construções:

- Numerais arábicos (1, 2, 3, ...) no Nível 1:

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M

```

- Letras minúsculas (a, b, c, ...) no Nível 2:

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M

```

- Numerais minúsculos romanos (i, ii, iii, ...) no Nível 3:

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M

```

- Letras maiúsculas (A, B, C, ...) no Nível 4:

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

No Exemplo 2.1.30 a seguir, alteramos o estilo de ordenamento do Nível 2, utilizando algarismos romanos:

Exemplo 2.1.30: Lista ordenada aninhada com níveis customizados

```

\renewcommand{\labelenumi}{\arabic{enumi}}
\renewcommand{\labelenumii}{\alph{enumii}}
\renewcommand{\labelenumiii}{\roman{enumiii}}

```

```

\renewcommand{\labelenumiv}{\Alph{enumiv}}
\begin{enumerate}
  \item Item 1
  \begin{enumerate}
    \item Item 1.1
    \begin{enumerate}
      \item Item 1.1.1
      \item Item 1.1.2
    \end{enumerate}
    \item Item 1.2
  \end{enumerate}
\item Item 2
\item Item 3
\begin{enumerate}
  \item Item 3.1
  \begin{enumerate}
    \item Item 3.1.1
    \begin{enumerate}
      \item Item 3.1.1.1
      \item Item 3.1.1.2
    \end{enumerate}
    \item Item 3.1.2
  \end{enumerate}
  \item Item 3.2
\end{enumerate}
\end{enumerate}

```

-
- 1 Item 1
 - a Item 1.1
 - i Item 1.1.1
 - ii Item 1.1.2
 - b Item 1.2
 - 2 Item 2
 - 3 Item 3
 - a Item 3.1
 - i Item 3.1.1

A Item 3.1.1.1

B Item 3.1.1.2

ii Item 3.1.2

b Item 3.2

2.1.8 Figuras

Figuras podem ser incluídas em um documento LaTeX de formas variadas. Dependendo da complexidade da informação apresentada, ambientes específicos devem ser utilizados para organizar não apenas a apresentação, mas também a redação do documento.

Uma figura pode ser incluída de forma simples utilizando o marcador

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```


Exemplo 2.1.31: Figura com o marcador

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

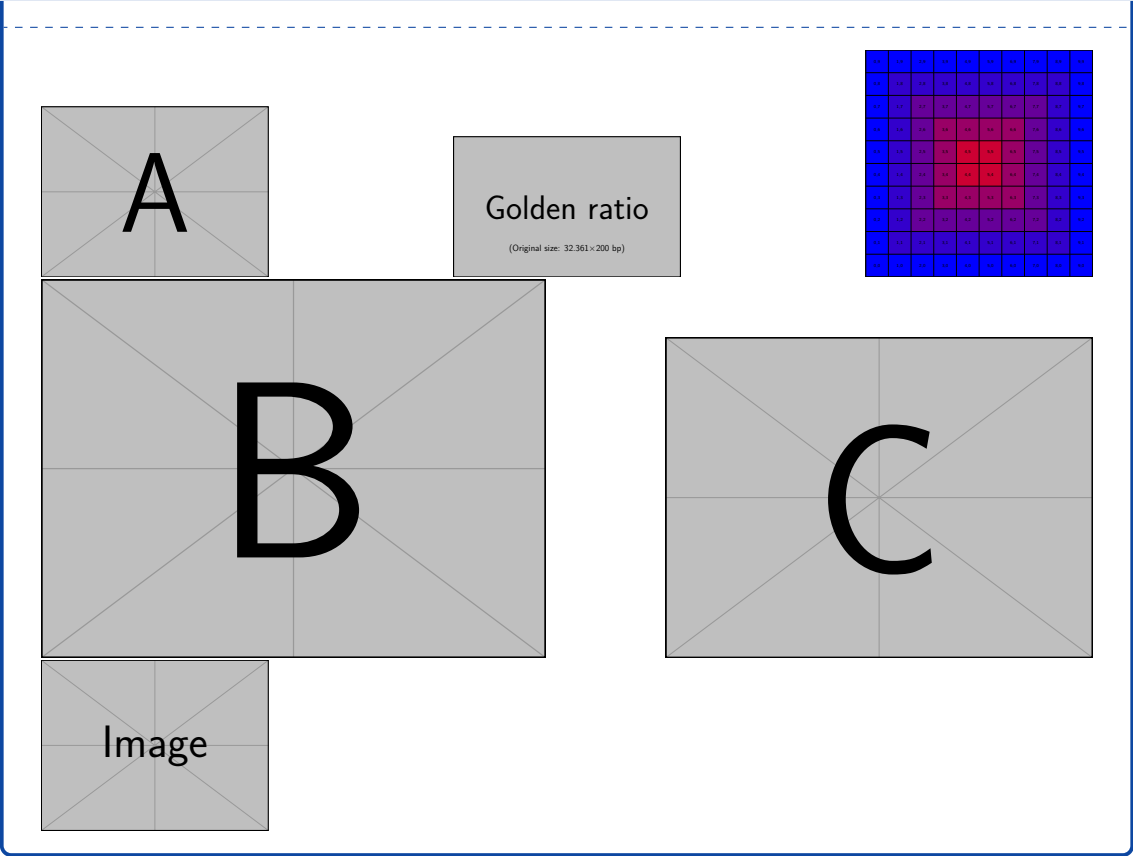
    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\includegraphics[width=3cm]{example-image-a}
\includegraphics[width=3cm]{example-image-golden}
\includegraphics[width=3cm]{example-grid-100x100pt}
\includegraphics[height=5cm]{example-image-b}
\includegraphics[scale=0.5]{example-image-c}
\includegraphics[width=3cm]{example-image}
```



As imagens do Exemplo 2.1.31 acima foram incluídas com o pacote `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M
```

sas imagens de exemplos.

que possui diver-

No Exemplo 2.1.31, observe que o marcador

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

aceita algu-
 mas opções que são delimitadas por um par de []. Pode-se especi-
 ficar, por exemplo, o tamanho da figura pode ser especificado com

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

as opções return M ,

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

ou

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

2.1.9 Formatos de Figuras

Figuras podem ser incorporadas a partir de diferentes formatos em um documento LaTeX. Os formatos preferenciais, entretanto, são o PDF e o EPS. Estes formatos são vetoriais e permitem, por exemplo, a impressão em alta resolução das figuras.

É possível converter formatos como o PNG, GIF e JPEG para os formatos PDF e EPS. Para tanto, recomenda-se a utilização do programa *Imagemagick* para esta conversão. Conversores online também podem ser utilizados para esta finalidade.

O *Imagemagick* possui um *script* chamado *convert* que pode ser utilizado para realizar a conversão entre estes formatos:

```
$ for i in $(ls *.png); do j=$(echo $i | sed 's,.png,.pdf,g');  
    convert -i $i -o $j; done
```

Outro detalhe que pode ser importante, é remover os espaços em branco nas margens das figuras. Isso é útil especialmente quando deseja-se incluir figuras lado a lado. Para isso, pode-se utilizar o *script* *autotrim* (que utiliza o comando *convert*):

```
$ autotrim -i figura.png -o figura_crop.png
```



Veja a página <http://www.fmwconcepts.com/imagemagick/index.php> com diversos exemplos e *scripts* úteis do *Imagemagick*.



Para mais exemplos sobre a incorporação e conversão entre formatos de arquivos de imagens, tenha como referência a página https://en.wikibooks.org/wiki/LaTeX/Importing_Graphics.

Ambientes de figuras

A forma mais simples de incluir figuras em um documento LaTeX, é a partir do `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)
```

comando `return M` . Observe que este comando (assim como a maioria dos comandos e marcadores da linguagem) possui uma seção de opções (ou argumentos) que são indicados entre os colchetes e o caminho para a imagem em si, que é informada entre as chaves. O Exemplo [2.1.32](#) mostra um exemplo simples:

Exemplo 2.1.32: Incorporando uma figura com o comando

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

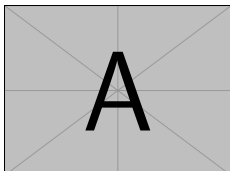
    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

```
\includegraphics[width=3cm]{example-image-a}
```



Observe, entretando, que apenas inserimos uma figura, mas não definimos uma posição relativa ao parágrafo ou à página. Para isso, é necessário incorporar o comando

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

dentro de um ambiente específico que permita o seu posicionamento relativo. Este ambiente, é o ambiente **figure**. O Exemplo 2.1.33 mostra um exemplo com o ambiente **figure**:

Exemplo 2.1.33: Incorporando uma figura com o comando

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)
```

```
    return M
```

dentro do ambi-

ente figure

```
\begin{figure}[h]
\includegraphics[width=3cm]{example-image-a}
\end{figure}
```

O ambiente `figure` deve ser configurado para possuir uma das seguintes posições relativas:

Tabela 2.4 - Opções de posicionamento relativo do ambiente `figure`.

Opção	Descrição
<code>h</code>	Posiciona o ambiente “aqui” (<code>h</code> vem do inglês <i>here</i>). A posição exata pode variar dependendo dos outros elementos textuais
<code>t</code>	Posiciona o ambiente no “topo” da página (<code>t</code> vem do inglês <i>top</i>)
<code>b</code>	Posiciona o ambiente na “base” da página (<code>b</code> vem do inglês <i>bottom</i>)
<code>p</code>	Posiciona o ambiente em uma “página” separada (<code>p</code> vem do inglês <i>page</i>)
<code>!</code>	Força o LaTeX a usar a posição textual onde o ambiente se encontra (e.g., <code>h!</code>)
<code>H</code>	Posiciona o ambiente precisamente no local em que se encontra (depende do pacote <code>float</code> e é equivalente a <code>h!</code>)

No Exemplo 2.1.34 é mostrado o posicionamento de uma figura utilizando a posição relativa `H`:

Exemplo 2.1.34: Incorporando uma figura com o comando

```
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

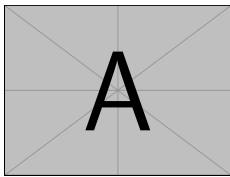
dentro do ambi-

ente figure com a posição relativa H

```
\lipsum[1]
\begin{figure}[H]
\includegraphics[width=3cm]{example-image-a}
\end{figure}
\lipsum[2]
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu

libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.



Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.



No estilo do INPE, o ambiente padrão para o posicionamento do ambiente **figure** é H.

Uma vez definido o posicionamento relativo (i.e., relativo ao parágrafo ou página), pode-se centralizar a figura utilizando-se o marcador

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

ou o ambiente center.

O Exemplo 2.1.35 mostra estas duas opções:

Exemplo 2.1.35: Centralizando figuras dentro do ambiente figure

```

\lipsum[1]
\begin{figure}[H]
\centering
\includegraphics[width=3cm]{example-image-a}
\end{figure}

```

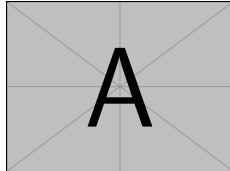


```

\lipsum[2]
\begin{figure}[H]
  \begin{center}
    \includegraphics[width=3cm]{example-image-a}
  \end{center}
\end{figure}

```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.



Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.



Observe no Exemplo 2.1.35, ambos o marcador
`import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

e o ambiente `center`
 devem ser colocados dentro do ambiente `figure`.

Figuras podem também ser colocadas dentro de um parágrafo de forma que o texto circunde o ambiente. Para isso, é necessário utilizar o ambiente `wrapfigure`:

Exemplo 2.1.36: Centralizando figuras dentro do ambiente `figure`

```
\begin{wrapfigure}{r}{0.25\textwidth}
  \centering
  \includegraphics[width=0.25\textwidth]{example-image-a}
\end{wrapfigure}
\lipsum[2]

\begin{wrapfigure}{l}{0.25\textwidth}
  \centering
  \includegraphics[width=0.25\textwidth]{example-image-b}
\end{wrapfigure}
\lipsum[3]
```

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

No Exemplo 2.1.36, observe que o ambiente `wrapfigure` aceita as opções `l` e `r`, que permite a figura ser alinhada à esquerda (`l`, do inglês *left*) ou à direita (`r`, do inglês *right*). Além disso, o ambiente pode ser posicionado da mesma forma como o ambiente `figure` padrão.

! Consulte a página https://www.overleaf.com/learn/latex/Inserting_Images para mais opções de configuração.

Construindo figuras

Figuras no LaTeX podem ser desenhadas utilizando pacotes específicos. As figuras podem ser incorporadas a partir de arquivos .tex separados ou desenhadas em ambientes apropriados. O pacote TikZ é um pacote do LaTeX orientado para a construção de diagramas. Com ele você pode criar diferentes tipos de gráficos, grafos, diagramas etc, que são alinhados com o formato SVG. Com o pacote pstricks, é possível criar imagens vetoriais complexas utilizando o interpretador do *GhostScript*. A diferença entre estes dois pacotes está mais relacionada com a forma como os seus resultados são interpretados e as suas imagens renderizadas dentro do documento LaTeX.

TikZ

A Figura 2.1.9 mostra um exemplo de uma imagem vetorial programada com o TikZ.

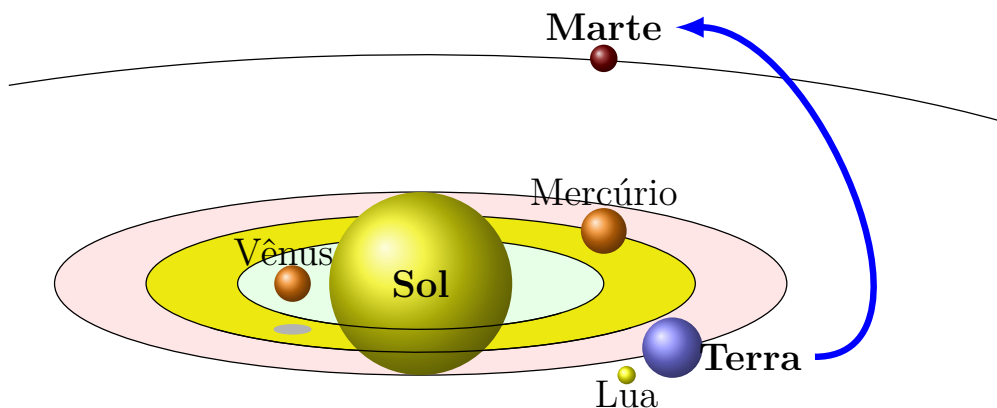


Figura 2.1 - Exemplo TickZ - Planetas.

! A página TeXTemplate possui vários exemplos utilizando o pacote TicZ.

PSTricks

A Figura 2.1.9 mostra um exemplo de uma figura vetorial programada com o PSTricks.

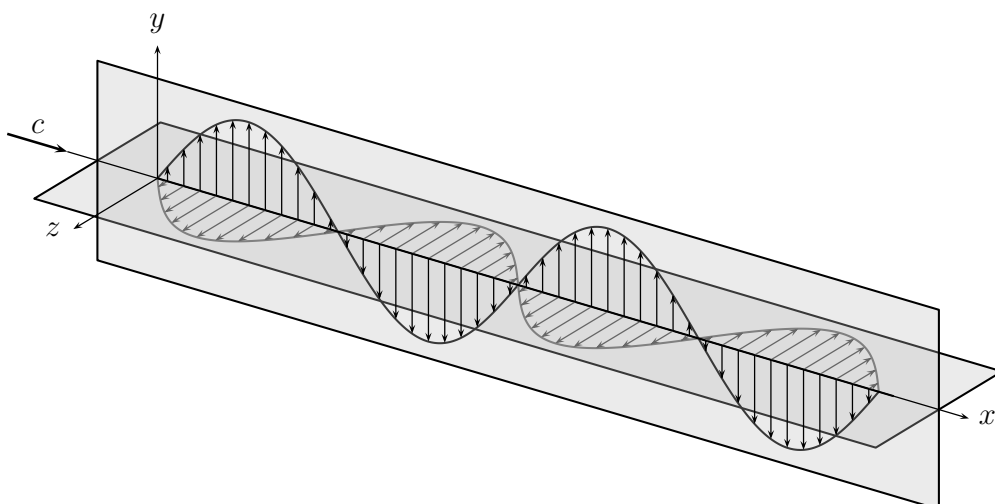


Figura 2.2 - Exemplo PSTricks - A Onda Eletromagnética.

Para facilitar o desenho de diagramas, você pode utilizar o programa [LaTeXDraw](https://tug.org/LaTeXDraw/) (em Java), disponível para os sistemas operacionais mais populares. Vários outros programas estão preparados para exportar gráficos para o formato TeX.



A página <https://tug.org/PSTricks/main.cgi?file=index> apresenta muitos exemplos de desenhos variados em PSTricks.

2.1.10 Matemática e equações

Equações podem ser digitadas diretamente em parágrafos (de forma *inline*) utilizando um par de \$'s como delimitadores. Por exemplo, equação $ax^2+bx+c=0$ pode

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

ser digitada como `return M` no meio de uma frase ou parágrafo.

No LaTeX é possível inserir praticamente todos os símbolos relacionados às ciências exatas. No Anexo ?? há uma lista destes símbolos, os quais poderão ser utilizados para a realizações dos exercícios da Seção ??.

Para digitar equações em blocos, há ambientes próprios para casos variados, os quais são mostrados a seguir.

Ambientes de equações

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)
```

O ambiente `return M`
ente de equação mais comum:

é o ambi-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

Exemplo 2.1.37: Ambiente `return M`

```

\begin{equation*}
\label{apI_eq:1}
J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} -
\mathbf{x}^{\{b\}})^T \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}^{\{b\}})
+ \frac{1}{2}[\mathbf{y}^{\{o\}} -
\textit{H}(\mathbf{x})]^T \mathbf{R}^{-1}[\mathbf{y}^{\{o\}} -
\textit{H}(\mathbf{x})]
\end{equation*}

```


$$J(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}^b)^T \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}^b) + \frac{1}{2}[\mathbf{y}^o - H(\mathbf{x})]^T \mathbf{R}^{-1}[\mathbf{y}^o - H(\mathbf{x})]$$

Quando for necessário alinhar equações, pode-se utilizar o ambiente `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

Exemplo 2.1.38: Ambiente `return M`

```

\begin{align*}
&\text{\label{apI_eq:3}} \\
\mathbf{y}^{\{o\}} - \text{\textit{H}}(\mathbf{x}) &= \mathbf{y}^{\{o\}} - \\
&\quad \text{\textit{H}}[\mathbf{x}^{\{b\}} + (\mathbf{x} - \mathbf{x}^{\{b\}})] \quad \text{\textit{H}} \\
&\text{\label{apI_eq:4}} \\
\mathbf{y}^{\{o\}} - \text{\textit{H}}(\mathbf{x}) &= \mathbf{y}^{\{o\}} - \\
&\quad \text{\textit{H}}(\mathbf{x}^{\{b\}}) - \text{\textit{H}}(\mathbf{x} - \\
&\quad \mathbf{x}^{\{b\}}) \\
&\text{\end{align*}}

```

$$\mathbf{y}^o - H(\mathbf{x}) = \mathbf{y}^o - H[\mathbf{x}^b + (\mathbf{x} - \mathbf{x}^b)]$$

$$\mathbf{y}^o - H(\mathbf{x}) = \mathbf{y}^o - H(\mathbf{x}^b) - H(\mathbf{x} - \mathbf{x}^b)$$

Observe no exemplo acima, que as equações estão alinhadas pelo sinal de `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

2.1.11 Tabelas

Tabelas são os elementos do texto que resumem e organizam informações que podem ser referenciadas no texto. No LaTeX, tabelas são escritas em ambientes específicos, que podem, dependendo da necessidade, ajustar automaticamente o seu conteúdo aos limites das dimensões do texto. Antes de apresentar os ambientes mais comuns de tabelas, salienta-se que a construção de tabelas pode se tornar uma tarefa um pouco mais complicada do que parece, principalmente se a tabela em questão possuir muitas células mescladas. Portanto, prefira construir tabelas de forma simples e clara, como a tabela do Exemplo 2.1.39.

Exemplo 2.1.39: Exemplo de uma tabela simples

```
\begin{tabular}{c c}
\hline
\textbf{L0C1} & \textbf{L0C2} \\
\hline
L1C1 & L1C2 \\
L2C1 & L2C2 \\
L3C1 & L3C2 \\
L4C1 & L4C2 \\
L5C1 & L5C2 \\
\hline
\end{tabular}
```

L0C1	L0C2
L1C1	L1C2
L2C1	L2C2
L3C1	L3C2
L4C1	L4C2
L5C1	L5C2

Na tabela do Exemplo 2.1.39, tem-se apenas duas colunas e algumas linhas. Para separar o conteúdo, utilizou-se apenas linhas horizontais para separar o cabeçalho, i.e., os nomes das colunas, do conteúdo. Pode-se, melhorar o aspecto do espaçamento das linhas utilizando o comando

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

. Lembre-se que

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

a instrução return M
pula uma linha; o argumento desta instrução, i.e.,

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

indica que o espaço de uma linha deve ser recuado em $-0.5em$. Na Tabela 2.2 que a unidade em refere-se à altura do caractere “M” da fonte em uso, isso garante que o espaçamento será sempre consistente independente do estilo da fonte em uso. Veja o Exemplo ?? a seguir:

Exemplo 2.1.40: Exemplo de uma tabela simples (Altura das Linhas)

```

\begin{tabular}{l r}
\hline

```

```

\\[-0.5em]
\textbf{L0C1} & \textbf{L0C2} \\
\\[-0.5em]
\hline
\\[-0.5em]
L1C1 & L1C2 \\
\\[-0.5em]
L2C1 & L2C2 \\
\\[-0.5em]
L3C1 & L3C2 \\
\\[-0.5em]
L4C1 & L4C2 \\
\\[-0.5em]
L5C1 & L5C2 \\
\\[-0.5em]
\hline
\end{tabular}

```

L0C1	L0C2
<hr/>	
L1C1	L1C2
L2C1	L2C2
L3C1	L3C2
L4C1	L4C2
L5C1	L5C2

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

No Exemplo 2.1.40, observe a instrução `return M`

Como a tabela do exemplo possui apenas duas colunas, indica-se com um par de colchetes o seu alinhamento, logo após o início do ambiente `tabular`. Neste caso, o conteúdo da coluna da esquerda encontra-se alinhado à esquerda, enquanto que o conteúdo da coluna da direita, encontra-se alinha à direita (por isso `l r`). Portanto, para alinhar o conteúdo à esquerda, utilize `l` (do inglês *left*), para alinhar à direita utilize `r` (do inglês *right*) e para centralizar o conteúdo (tal como no Exemplo 2.1.39), utilize `c` (do inglês *center*).

Além de alterar o espaçamento vertical dentro de uma tabela, pode-se tam-

bém alterar a largura das colunas. Para isso, pode-se utilizar o comando `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

, onde u. corresponde a alguma medida. Veja o Exemplo 2.1.41 a seguir:

Exemplo 2.1.41: Exemplo de uma tabela simples (Largura das Colunas)


```
\begin{tabular}{p{3cm} p{5cm}}
\hline
\\[-0.5em]
\textbf{LOC1} & \textbf{LOC2} \\
```

```

\\[-0.5em]
\hline
\\[-0.5em]
L1C1 & L1C2 \\
\\[-0.5em]
L2C1 & L2C2 \\
\\[-0.5em]
L3C1 & L3C2 \\
\\[-0.5em]
L4C1 & L4C2 \\
\\[-0.5em]
L5C1 & L5C2 \\
\\[-0.5em]
\hline
\end{tabular}

```

L0C1	L0C2
L1C1	L1C2
L2C1	L2C2
L3C1	L3C2
L4C1	L4C2
L5C1	L5C2


 Observe no Exemplo 2.1.41, que o conteúdo das colunas foi marcado como **p** (do inglês *paragraph*). Nessa forma mais simples de se especificar a largura das colunas, não é possível posicionar o texto de outra forma.

Assim como as tabelas produzidas em editores WYSIWYG, no LaTeX também é possível mesclar células (na direção das colunas ou das linhas). Para isso, utiliza-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

sem os comandos

return M

para

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

mesclar colunas e `return M` para
mesclar linhas. Veja o Exemplo 2.1.42 a seguir:

Exemplo 2.1.42: Exemplo de uma tabela simples

```

\begin{tabular}{|p{3cm}|p{3cm}|p{3cm}|p{3cm}|}
\hline
\multicolumn{4}{|c|}{4 Células Mescladas (colunas)} \\
\hline
\multicolumn{2}{|c|}{2 Células Mescladas (colunas)} &

```

```

\multicolumn{2}{c|}{2 Células Mescladas (colunas)} \\
\hline
\multicolumn{1}{|c|}{Coluna 1} & \multicolumn{1}{c|}{Coluna 2} &
\multicolumn{1}{c|}{Coluna 3} & \multicolumn{1}{c|}{Coluna 4} \\
\hline
\lipsumsentence[1-2] & \lipsumsentence[3-4] & \lipsumsentence[5-6]
& \lipsumsentence[7-8] \\
\hline
\end{tabular}

```

4 Células Mescladas (colunas)			
2 Células Mescladas (colunas)		2 Células Mescladas (colunas)	
Coluna 1	Coluna 2	Coluna 3	Coluna 4
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis.	Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate magna.	Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.	Mauris ut leo. Cras viverra metus rhoncus sem.

Na tabela do Exemplo 2.1.42, tem-se uma tabela mais complexa, em que colunas estão mescladas de formas diferentes. Além disso, diferentemente dos exemplos anteriores, a tabela apresentada possui limitadores verticais que são desenhados utilizando-se o símbolo *pipe* (`\`), como argumento do comando que

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

inicia o ambiente `return M` :

|p3cm|p3cm|p3cm|p3cm|. Neste caso, observe que a tabela desenhada possui um total de quatro colunas, cujas larguras podem ser especificadas (no exemplo, cada uma com 3cm de comprimento). Outro detalhe a ser observado neste exemplo, é a forma como o conteúdo das células individuais é alinhado dentro das células. Neste caso, o alinhamento é dado por um argu-

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    mento do comando      return M      :

```



```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

4 indica a quantidade de células a serem mescladas e

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M

```

indica que o conteúdo das células a serem mescladas será centralizado e delimitado por *pipes* nos limites da célula.

Ambientes de tabelas

Dependendo da necessidade, ambientes especiais de tabelas podem ser necessários. Alguns ambientes de tabelas mais comuns são **tabular**, **tabularx** e **booktabs**, os quais possuem características e propriedades específicas.

Com relação ao ambiente **tabular**, tem-se duas variações do mesmo, os ambientes

`tabular*` e o `tabularx`. A diferença entre eles está na forma como a largura da tabela é ajustada com relação ao texto. Observe o Exemplo 2.1.43: a tabela gerada com o ambiente `tabular` não possui largura fixa e ela se ajusta à quantidade de conteúdo, tendo como limite a largura do parágrafo onde estiver inserida.

Exemplo 2.1.43: Exemplo de uma tabela simples utilizando o ambiente `tabular`

```
\begin{tabular}{|l|c|r|}
\hline
foo   & bar   & fubar \\
fubar & toobar & foo   \\
\hline
\end{tabular}
```

foo	bar	fubar
fubar	toobar	foo

Já no Exemplo 2.1.44, a tabela gerada com o ambiente `tabular*` possui largura fixa, e o seu limite é a largura do parágrafo no qual está inserida (por isso o marcado `*`). Observe que o conteúdo da tabela se ajusta à largura fixa.

Exemplo 2.1.44: Exemplo de uma tabela simples utilizando o ambiente `tabular*`

```
\begin{tabular*}{\textwidth}{@{\extracolsep{\fill}}|l|c|r|}
\hline
foo   & bar   & fubar \\
fubar & toobar & foo   \\
\hline
\end{tabular*}
```

foo	bar	fubar
fubar	toobar	foo

O Exemplo 2.1.45 a seguir, utiliza o ambiente `tabularx` que permite deixar as colunas da tabela com tamanhos iguais, além de possuir largura total fixa com limite relativo à largura do parágrafo (assim como o ambiente `tabular*`):

Exemplo 2.1.45: Exemplo de uma tabela simples utilizando o ambiente `tabularx`

```
\begin{tabularx}{\textwidth}{|X|X|X|}  
  \hline  
  foo   & bar   & fubar \\  
  fubar & toobar & foo  \\  
  \hline  
\end{tabularx}
```

foo	bar	fubar
fubar	toobar	foo

O ambiente `booktabs` permite utilizar linhas mais grossas através dos

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    marcadores return M ,

```

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

e

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M

```

. Veja o Exemplo 2.1.46 a seguir e compare o resultado com as tabelas dos exemplos anteriores que

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

utilizaram o marcador `return M`
para separar as linhas das tabelas:

Exemplo 2.1.46: Exemplo de uma tabela simples utilizando o ambiente tabular
import numpy as np

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

e os marcadores      return M
import numpy as np
```

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)
```

```

\begin{tabular}[t]{lcc}
\toprule
& Coluna 1 & Coluna 2 & \\
\midrule
John Smith & 1 & 2 & \\
Jane Doe & -- & 3 & \\
Mary Johnson & 4 & 5 & \\
\bottomrule
\end{tabular}

```

	Coluna 1	Coluna 2
John Smith	1	2
Jane Doe	–	3
Mary Johnson	4	5

Para utilizar o ambiente booktabs, é necessário carregar o pacote booktabs. Para isso, digite a linha `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

documento.

no preâmbulo do

! Você pode utilizar editores online para construir tabelas simples no LaTeX. Tenha em mente que tabelas muito complexas são difíceis de manipular e atualizar. Veja o site <https://www.tablesgenerator.com/> e <https://www.latex-tables.com> para mais informações.

2.1.12 Outros ambientes

Listing

Muitas vezes, dependendo do tipo de documento que se está produzindo, faz-se necessário a inserção de códigos que representam um determinado processo. Um exemplo, é quando se quer mostrar um código escrito em alguma linguagem de programação. O LaTeX possui alguns pacotes que permitem destacar a seção de código inserida, em ambientes específicos. O ambiente `verbatim` é o mais simples de ser utilizado, e poder ser aplicado para destacar algum tipo de texto. Algumas peculiaridades do ambiente, é que ele “escapa” os comandos da linguagem. Veja no Exemplo 2.1.47 um exemplo.

Exemplo 2.1.47: Exemplo de uso do ambiente `verbatim` para destacar texto

```
\begin{verbatim}
Text enclosed inside \texttt{verbatim} environment is
printed directly and all \LaTeX{} commands are ignored.
\end{verbatim}
```

```
Text enclosed inside \texttt{verbatim} environment is
printed directly and all \LaTeX{} commands are ignored.
```

No Exemplo 2.1.48, utilizou-se o mesmo ambiente do Exemplo 2.1.47, mas adicionou-se um `*` no início do ambiente `verbatim`, de forma que os espaços entre as palavras sejam realçados.

Exemplo 2.1.48: Exemplo de uso do ambiente `verbatim*` para destacar texto

```
\begin{verbatim*}
Text enclosed inside \texttt{verbatim} environment is
printed directly and all \LaTeX{} commands are ignored.
\end{verbatim*}
```

Text enclosed inside `\texttt{verbatim}` environment is printed directly and all `\LaTeX{}` commands are ignored.

É possível também utilizar o ambiente `verbatim inline`, ou seja, diretamente dentro de um parágrafo, o que pode ser útil quando se necessita destacar algum comando (e.g., quando o contexto requer isso). Para utilizar o ambiente `verbatim inline`, utilize o comando `import numpy as np`

```
def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int) dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

    return M
```

precedendo o comando

```

import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None to become the incidence matrix
    VT = np.zeros((n*m,1), int)  dummy variable

    compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

    VT = np.zeros((n*m,1), int)

```

desejado: “o comando `return M`
produz `LATEX`”.

O pacote `listings` é o mais simples de ser utilizado, mas aceita diferentes opções, que permitem realçar as palavras reservadas da linguagem, além de mostrar a numeração das linhas e criar uma caixa ao redor do código fonte mostrado. No Exemplo 2.1.49, é mostrado um exemplo de código-fonte Python com algumas opções do pacote `listings`

Exemplo 2.1.49: Exemplo da apresentação de um código escrito em linguagem Python utilizando o pacote listings

```
\begin{lstlisting}
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int) #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

            if M is None:
                M = np.copy(VT)
            else:
                M = np.concatenate((M, VT), 1)

            VT = np.zeros((n*m,1), int)

    return M
\end{lstlisting}
```

```
import numpy as np

def incmatrix(genl1 , genl2):
    m = len(genl1)
```