LaTeX

Teresa Becerril Torres
terebece1508@gmail.com

1. Introducción

1.1. ¿Qué es LaTeX?

Es un procesador de texto que funciona mediante comandos de control y ejecuta un programa para poder visualizar el texto ya formateado, es decir, primero recibe todo el texto y posteriormente lo transforma con el formato correspondiente. Este procesador sirve para elaborar notas, cartas, reportes, tareas, presentaciones, artículos, libros, tesis, etc.

1.2. Breve historia de LaTeX

En los años ochenta Leslie Lamport escribió una serie de macros de TeX para facilitar su uso, TeX es un sistema de tipografía desarrollado por Donald E. Knuth en 1978. Esto dio lugar a LaTeX, una abreviatura de Lamport TeX.

1.3. ¿Por qué lo usamos?

Algunas de las razones por las que lo utilizamos son:

a) Es un paquete de distribución gratuita, disponible en casi todos los sistemas Unix.

b) Permite un manejo más preciso del documento, lo cual brinda una edición más rápida.

c) Cuenta con una gran cantidad de paquetes adicionales para casi cualquier cosa, como graficación, música, animación, notación científica, entre otros.

d) Fácil control de versiones, ya que permite utilizar Git para implementar el control de versiones y hacer un seguimiento de los cambios.

e) Es el más usado en el medio de Ciencias de la Computación y Matemáticas para la elaboración de trabajos.

1.4. Diferencias entre LaTeXy Word

Algunas de las diferencias entre LaTeXy Word son:

a) Word es un procesador de textos que muestra en la misma pantalla la forma y formato del documento que estamos creando. En LaTeX, en cambio, añadimos el contenido junto con comandos que indican el formato de las distintas partes del documento y para poder visualizar el documento final es necesario un proceso de compilación.

b) En Word, cuando un documento alcanza una longitud considerable, la velocidad de carga aumenta, referencias internas se rompen, etc. Todos estos problemas son una consecuencia de trabajar simultáneamente el contenido y el formato. A diferencia de LaTeX que trabaja con un documento sencillo sin formato, además, gestiona automáticamente los índices de contenido, las listas de bibliografía o las referencias cruzadas a imágenes.

1.5. Overleaf

Overleaf[1] es un editor de LaTeX en línea que permite a múltiples usuarios trabajar en cualquier documento al mismo tiempo y la salida de código se compila automáticamente para proporcionar una vista previa en tiempo real de cualquier cambio producido en el documento. También detecta todos los errores de código y permite realizar un control de versiones.

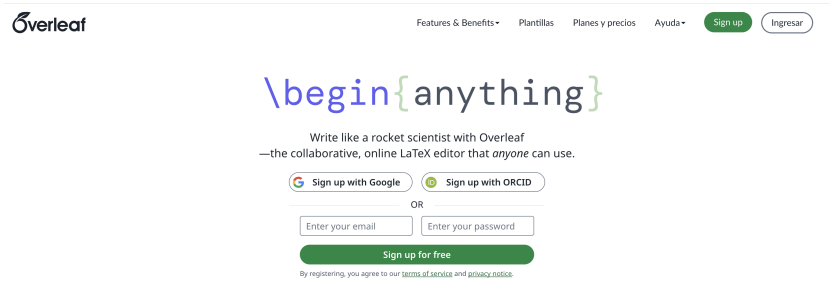


Figura 1: Página de inicio de Overleaf.

1.6. Detexify

Detexify[2] es una página web que sirve para encontrar cualquier símbolo. Su uso es muy sencillo: dibujamos el símbolo que buscamos en el cuadro blanco y nos aparece tanto el comando como el paquete al que pertenece cada una de las opciones más parecidas a dicho símbolo.

[1] Página de Overleaf: https://es.overleaf.com/
[2] Página de Detexify: https://detexify.kirelabs.org/classify.html

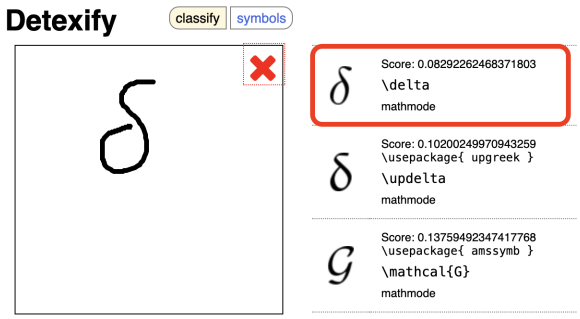


Figura 2: Búsqueda del símbolo delta en Detexify.

2. Instalación

2.1. LaTeX

En esta subsección vamos a ver cómo instalar LaTeX en Fedora y Debian.

2.1.1. Fedora

Lo primero que debemos hacer es actualizar la información de los paquetes que pueden ser instalados:

```
$ sudo dnf update
```

Una vez que se realice la actualización de los paquetes, podemos instalar LaTeX con los siguientes comandos:

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

2.1.2. Debian

Primero actualizamos la información de los paquetes:

```
$ sudo apt-get update
```

Posteriormente, procedemos con la instalación de LaTeX mediante:

```
$ sudo apt-get install texlive
```

La instalación tanto en Fedora como en Debian tomará algunos minutos.

2.2. Paquetes

En esta subsección vamos a ver cómo instalar tanto clases como paquetes en Fedora y Debian.

2.2.1. Fedora

- Para instalar una clase utilizamos:

```
$ sudo dnf install 'tex(clase.cls)'
```

  Donde `clase` es el nombre de la clase que deseamos instalar.

- Para instalar un paquete usamos:

```
$ sudo dnf install 'tex(paquete.sty)'
```

  Donde `paquete` es el nombre del paquete que queremos instalar.

2.2.2. Debian

Lo primero que debemos hacer es entrar al sitio web de CTAN y buscar el paquete o clase que deseamos instalar, descargar el archivo `nombre.tar.gz.`, donde `nombre` es el nombre del paquete o clase. Una vez que el archivo se haya descargado, debemos descomprimirlo mediante:

```
$ tar -xvf nombre.tar.gz
```

Antes de continuar debemos verificar que el archivo `nombre.ins` aparezca en el directorio de Descargas, para ello podemos utilizar el comando `ls`. Si el archivo se encuentra, lo ejecutamos a través de LaTeX:

```
$ latex nombre.ins
```

Esto va a generar el archivo `nombre.sty`, el cual vamos a utilizar para instalar el paquete o clase.

- Instalación en un usuario específico.

  Primero debemos crear el siguiente directorio:

```
$ mkdir -p ~/texmf/tex/latex/nombre
```

  A continuación, copiamos el archivo `nombre.sty` en el directorio que acabamos de crear mediante:

```
$ cp nombre.sty ~/texmf/tex/latex/nombre
```

  Finalmente, actualizamos el caché de LaTeX con:

```
$ texhash ~/texmf
```

  El paquete sólo estará disponible para ese usuario.

- Instalación en el sistema.

  Lo primero que debemos hacer es convertirnos en superusuarios, para ello vamos a utilizar el comando `su root`. Posteriormente, creamos el siguiente directorio:

```
# mkdir /usr/share/texmf/tex/latex/nombre
```

A continuación, copiamos el archivo `nombre.sty` en el directorio mediante:

```
# cp nombre.sty /usr/share/texmf/tex/latex/nombre
```

Finalmente, actualizamos el caché de LaTeX con:

```
$ texhash
```

El paquete estará disponible para todos los usuarios del sistema.

2.3. Documentación

CTAN[3] (Comprehensive TeX Archive Network) es un sitio web de referencia, en el cual podemos consultar la documentación de las clases y paquetes de LaTeX.

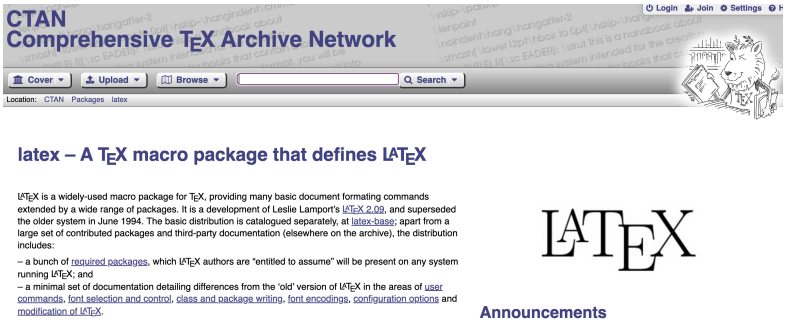


Figura 3: Página de inicio de CTAN para la documentación de LaTeX.

3. LaTeX

Lo primero que debemos hacer para crear un documento en LaTeX es indicar el tipo de documento en el que queremos trabajar. Esto se indica mediante el siguiente comando:

```
\documentclass{article}
```

Las clases controlan la apariencia general del documento, en este caso hemos creado un documento con la clase `article`. También existen otras clases como: `book`, `report`, `beamer` o `letter`. A continuación, debemos agregar el preámbulo, el cual actúa como la sección de "configuración" del documento. En esta sección puede incluirse detalles específicos del documento, los paquetes que se deben cargar para agregar funcionalidades, entre otras configuraciones. Por ejemplo:

```
\documentclass[12pt, letterpaper]{article}
\usepackage[spanish,es-tabla]{babel}
\usepackage{graphicx}
```

Los parámetros adicionales se incluyen entre corchetes [...] y deben estar separados por comas. En este ejemplo, los parámetros hacen lo siguiente:

[3] Página de CTAN: https://www.ctan.org/pkg/latex

- `12pt` - Establece el tamaño de la fuente.
- `letterpaper` - Establece el tamaño del papel.
- `spanish` - Establece el idioma del documento.
- `es-tabla` - Modifica las etiquetas de las tablas.

La línea del preámbulo `\usepackage{graphicx}` nos permite cargar un paquete externo (en este caso, `graphicx`). Existe una gran cantidad de paquetes que pueden ser cargados en LaTeX: para crear fórmulas matemáticas, tablas, figuras, gráficas, etc. En el preámbulo también podemos agregar el título, autor y fecha del documento, utilizando los siguientes tres comandos:

- `\title{Mi documento}` - El título del documento.
- `\author{Teresa Becerril}` - El nombre del autor, en caso de que sean varios autores, se agrega `\\` después de cada nombre.
- `\date{Julio 2024}` - Fecha del documento, se puede ingresar manualmente o usar el comando `\today` para escribir la fecha actual.

Con estas líneas agregadas, el preámbulo debería verse así:

```
\documentclass[12pt, letterpaper]{article}
\usepackage[spanish,es-tabla]{babel}
\usepackage{graphicx}
\title{Mi documento}
\author{Teresa Becerril}
\date{Julio 2024}
```

Una vez declarado el preámbulo, debemos indicar el comienzo del contenido mediante las etiquetas `\begin{document}` y `\end{document}`, las cuales delimitan el contenido de nuestro documento y con ellas podemos crear un documento que contenga el texto que nosotros deseamos.

```
\begin{document}
Mi primer documento en \LaTeX!
\end{document}
```

Para que el título, autor y fecha sean visibles en el documento debemos agregar el comando `\maketitle` después de la línea `\begin{document}`, como se muestra a continuación:

```
\documentclass[12pt, letterpaper]{article}
\usepackage[spanish,es-tabla]{babel}
\usepackage{graphicx}
\title{Mi documento}
\author{Teresa Becerril}
\date{Julio 2024}
\begin{document}
\maketitle
Mi primer documento en \LaTeX!
\end{document}
```

Este ejemplo produce el siguiente salida:

Mi documento

Teresa Becerril

Julio 2024

Mi primer documento en LaTeX!

3.1. Estructura básica del documento

En esta subsección veremos la estructura básica de un documento en LaTeX, en diferentes párrafos, secciones y capítulos.

3.1.1. Párrafos y nuevas líneas

Para escribir párrafos y líneas en nuestro documento debemos incluirlos dentro del entorno `document`, de la siguiente manera:

```
\begin{document}
Este sera el primer parrafo del documento, debemos
presionar ‘‘enter’’ dos veces para comenzar el segundo.
Aenean quis est leo. Class aptent taciti sociosqu ad
litora torquent per conubia nostra, per inceptos himenaeos.
Maecenas sit amet enim a lectus feugiat elementum.

En esta linea comenzara el segundo parrafo.

Esta linea es el tercer parrafo y podemos agregar \\
para saltar de linea manualmente, el texto comienza en
una nueva linea pero sigue siendo parte del mismo parrafo.

\end{document}
```

Este ejemplo genera el siguiente resultado:

Este sera el primer parrafo del documento, debemos presionar “enter” dos veces para comenzar el segundo. Aenean quis est leo. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Maecenas sit amet enim a lectus feugiat elementum.

En esta linea comenzara el segundo parrafo.

Esta linea es el tercer parrafo y podemos agregar \\ para saltar de linea manualmente, el texto comienza en una nueva linea pero sigue siendo parte del mismo parrafo.

Como podemos ver en el ejemplo anterior, LaTeX aplica automáticamente la sangría a los párrafos. Se recomienda no utilizar múltiples \\ para agregar espacios en blanco, ya que pueden causar un mensaje de aviso o de error.

3.1.2. Capítulos y secciones

Los documentos suelen dividirse en partes, capítulos, secciones, subsecciones, etc. LaTeX proporciona comandos de estructura, pero estos dependen de la clase de documento que se utilice. Por ejemplo, los documentos creados con la clase `book` se pueden dividir en capítulos, secciones, subsecciones, etc. El siguiente ejemplo se muestran los comandos utilizados para estructurar un documento según la clase `book`:

```
\documentclass{book}
\begin{document}

\chapter{Primer capitulo}

\section{Introduccion}

Esta es la primera seccion.

Lorem ipsum dolor sit amet, consectetuer adipiscing
elit. Etiam lobortisfacilisis sem. Nullam nec mi et
neque pharetra sollicitudin. Praesent imperdietmi nec ante.
Donec ullamcorper, felis non sodales...

\section{Segunda seccion}

Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Etiam lobortis facilisissem. Nullam nec mi et neque pharetra
sollicitudin. Praesent imperdiet mi necante...

\subsection{Primero subseccion}
Praesent imperdietmi nec ante. Donec ullamcorper,
felis non sodales...
\end{document}
```

Este ejemplo produce el siguiente salida:

Chapter 1

Primer capitulo

1.1 Introduccion

Esta es la primera seccion.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortisfacilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdietmi nec ante. Donec ullamcorper, felis non sodales...

1.2 Segunda seccion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisissem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi necante...

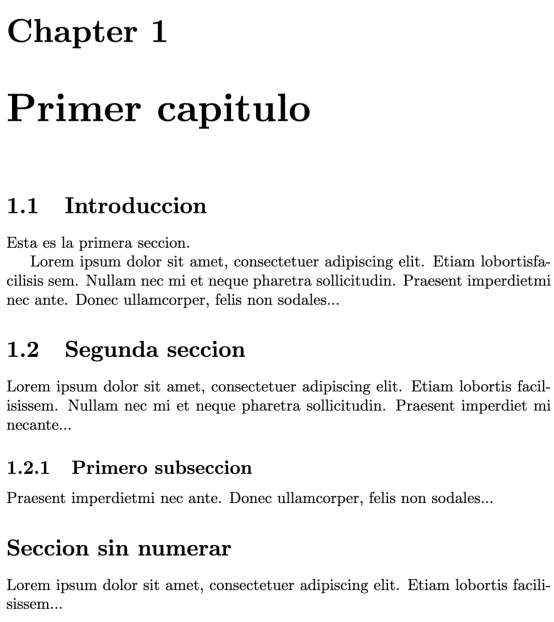
1.2.1 Primero subseccion

Praesent imperdietmi nec ante. Donec ullamcorper, felis non sodales...

La numeración es automática, pero la podemos desactivar utilizando un asterisco * al final del nombre del comando de estructura, por ejemplo:

```
\section*{Seccion sin numerar}
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Etiam lobortis facilisissem...
```

Este ejemplo genera el siguiente resultado:



Chapter 1

Primer capitulo

1.1 Introduccion

Esta es la primera seccion.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortisfacilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdietmi nec ante. Donec ullamcorper, felis non sodales...

1.2 Segunda seccion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisissem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi necante...

1.2.1 Primero subseccion

Praesent imperdietmi nec ante. Donec ullamcorper, felis non sodales...

Seccion sin numerar

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisissem...

Las clases de documentos en LaTeX proporcionan los siguientes comandos de estructura:

- `part{...}` - Parte
- `chapter{...}` - Capítulo
- `section{...}` - Sección
- `subsection{...}` - Subsección
- `subsubsection{...}` - Subsubsección
- `paragraph{...}` - Párrafo
- `subparagraph{...}` - Subpárrafo

Los comandos `part{...}` y `chapter{...}` sólo están disponibles en las clases `report` y `book`.

3.2. Comentarios

Los comentarios son una sección de texto que se usa para agregar notas de "cosas por hacer", notas explicativas, organizar el texto o comentar líneas/secciones de código al depurar el documento fuente. Para añadir un comentario debemos colocar el carácter reservado `%` al principio del texto que queremos comentar. Como se muestra en el siguiente código:

```
\documentclass[12pt, letterpaper]{article}
\usepackage[spanish,es-tabla]{babel}
\usepackage{graphicx}
\title{Mi documento}
\author{Teresa Becerril}
\date{Julio 2024}
\begin{document}
\maketitle
Mi primer documento en \LaTeX!

% Mi primer comentario!!!!
\end{document}
```

Este ejemplo produce la siguiente salida:

Mi documento

Teresa Becerril

Julio 2024

Mi primer documento en LaTeX con comentarios.

3.3. Tipos y tamaños de letras

Los tipos de letra son comandos de cambio de estado que vienen con un argumento, el cual es el texto que se va a cambiar. Algunos de ellos cuentan con abreviaturas, las cuales funcionan como comandos de cambio de estado, por lo que si solamente queremos que se aplique a una parte del texto, debemos encerrar el comando junto con el texto entre llaves, como se muestra a continuación:

```
{\sc TeXtO EsCaLoNaDaS}

Texto normal

\textit{Texto en cursiva}
```

TeXtO EsCaLoNaDaS

Texto normal

Texto en cursiva

Los tipos de letras que proporciona LaTeX son:

Comando	Abr.	Descripción
`\textmd{...}`		Texto en tipo normal
`\textbf{...}`	`\bf`	**Texto en negritas**
`\textrm{...}`	`\rm`	Texto en Roman Family
`\textsf{...}`	`\sf`	Texto en Sans Serif Family
`\texttt{...}`	`\tt`	`Texto como de máquina de escribir`
`\textup{...}`		Texto sin inclinación
`\textit{...}`	`\it`	*Texto en itálicas*
`\textsl{...}`	`\sl`	*Texto ligeramente inclinado*
`\textsc{...}`	`\sc`	Texto en mayúsculas EsCaLaDaS

Tabla 1: Tipos de letras.

Para cambiar el tamaño de las letras vamos a usar un comando de estado, por lo que también debemos escribir entre llaves el tamaño de la letra que vamos a utilizar, como se muestra a continuación:

```
{\Large Texto grande}

Texto normal

{\tiny Texto diminuto}
```

Texto grande

Texto normal

Texto diminuto

Los tamaños de letras que proporciona LaTeX son:

Comando	Descripción del tamaño
`\tiny`	Texto en tamaño tiny
`\scriptsize`	Texto en tamaño scriptsize
`\footnotesize`	Texto en tamaño footnotesize
`\small`	Texto en tamaño small
`\normalsize`	Texto en tamaño normalsize
`\large`	Texto en tamaño large
`\Large`	Texto en tamaño Large
`\LARGE`	Texto en tamaño LARGE
`\huge`	Texto en tamaño huge
`\Huge`	Texto en tamaño Huge

Tabla 2: Tamaños de letras.

Sin embargo, cuando se está usando ambiente matemático estos comandos no tendrán efecto alguno dentro del ambiente y pueden causar un mensaje de aviso.

3.4. Ejercicio

En Emacs crea un archivo `ejemplo1.tex` de tipo **article** y realiza lo siguiente:

- Escribe la estructura básica del documento: preámbulo y el entorno document.
- Agrega el título, autor y fecha del documento: el título va a ser Ejemplo 1, el autor va a ser tu nombre y la fecha va a ser la de hoy.
- Añade dos secciones: la primera sección sin numerar y debe llamarse Introducción y la segunda debe llamarse Ejercicios.
- En la primera sección, deben agregar un comentario y escribir tres párrafos pequeños:
    - En el primer párrafo las letras deben ser de tamaño Large y en itálicas.

- En el segundo párrafo las letras tienen ser de tamaño small y mayúsculas escalonadas.
- En el tercer párrafo las letras deben ser de tamaño Huge y en negritas.

Pueden utilizar la página Lipsum para generar los párrafos: https://es.lipsum.com/

3.5. Listas

En LaTeX se pueden crear diferentes tipos de listas mediante entornos, los cuales se utilizan para encapsular el código necesario para implementar una función de composición tipográfica específica. Un entorno comienza con `begin{entorno}` y termina con `end{entorno}` donde `entorno` podría ser uno de los tipos de listas: `itemize` para listas desordenadas o `enumerate` para listas ordenadas.

3.5.1. Listas desordenadas

Las listas desordenadas son generadas por el entorno `itemize`, cada entrada de la lista debe ir precedida del comando `\item`, como se muestra a continuación:

```
\begin{itemize}
    \item Orientada a objetos
    \item Funcional
    \item Logica
    \item Estructurada
\end{itemize}
```

- Orientada a objetos
- Funcional
- Logica
- Estructurada

Podemos modificar el símbolo de cada entrada de la lista, para ello debemos colocar el símbolo deseado entre corchetes [...] después del comando `\item`, como se ilustra en el siguiente ejemplo:

```
\begin{itemize}
    \item[$\bullet$] Orientada a objetos
    \item[$\circ$] Funcional
    \item[$\diamond$] Logica
    \item[$\star$] Estructurada
\end{itemize}
```

- Orientada a objetos
- Funcional
- Logica
- Estructurada

Estas listas se pueden anidar entre sí hasta cuatro niveles de profundidad, el símbolo de cada nivel cambia, por ejemplo:

```
\begin{itemize}
    \item Declarativa
    \begin{itemize}
        \item Funcional
        \begin{itemize}
            \item Haskell
            \item Scale
            \item Scheme
        \end{itemize}
        \item Logica
    \end{itemize}
    \item Imperativa
\end{itemize}
```

- Declarativa
  - Funcional
    - Haskell
    - Scale
    - Scheme
  - Logica
- Imperativa

3.5.2. Listas ordenadas

Las listas ordenadas utilizan la misma sintaxis que las listas desordenadas, pero se crean utilizando el entorno `enumerate`:

```
\begin{enumerate}
    \item Orientada a objetos
    \item Funcional
    \item Logica
    \item Estructurada
\end{enumerate}
```

1. Orientada a objetos
2. Funcional
3. Logica
4. Estructurada

Estas listas también las podemos anidar entre sí, cada nivel tiene su propio contador como se muestra a continuación:

```
\begin{enumerate}
    \item Declarativa
    \begin{enumerate}
        \item Funcional
        \begin{enumerate}
            \item Haskell
            \item Scale
            \item Scheme
        \end{enumerate}
        \item Logica
    \end{enumerate}
    \item Imperativa
\end{enumerate}
```

1. Declarativa
    a) Funcional
        1) Haskell
        2) Scale
        3) Scheme
    b) Logica
2. Imperativa

Además, podemos definir qué tipo de numeración queremos utilizar en la lista. La forma más sencilla es incluyendo el paquete `enumerate` en el preámbulo del documento y colocar el tipo de numeración que deseamos usar de la siguiente manera:

Preámbulo

```
\usepackage{enumerate}
```

Cuerpo

```
\begin{enumerate}[a)]
    \item Orientada a objetos
    \item Funcional
    \item Logica
    \item Estructurada
\end{enumerate}
```

a) Orientada a objetos
b) Funcional
c) Logica
d) Estructurada

Nota: Se puede crear un entorno `enumerate` dentro de un entorno `itemize` y viceversa.

```
\begin{itemize}
    \item Declarativa
    \begin{enumerate}
        \item Funcional
        \begin{itemize}
            \item Haskell
            \item Scale
            \item Scheme
        \end{itemize}
        \item Logica
    \end{enumerate}
    \item Imperativa
\end{itemize}
```

- Declarativa
  1. Funcional
     - Haskell
     - Scale
     - Scheme
  2. Logica
- Imperativa

3.5.3. Ejercicio

En la sección Ejercicios del archivo `ejemplo1.tex` realiza lo siguiente:

- Crea una lista ordenada de tres elementos con el entorno enumerate.
- Crea una lista desordenada de tres elementos con el entorno itemize.
- Crea una lista de dos elementos con el entorno enumerate y cambia el tipo de contador, para ello debes agregar el paquete enumerate en el preámbulo.
- Agrega una lista de dos elementos con el entorno itemize dentro de la última lista y modifica el símbolo de cada entrada.

3.6. Modo matemático

Una de las principales ventajas de LaTeXes la facilidad con la que se pueden escribir expresiones matemáticas. Ofrece dos tipos de ambientes matemáticos:

1. Modo matemático en línea, se utiliza para poder escribir una expresión o un símbolo dentro de un parrafo. Esto se logra, de cualquiera de las siguientes maneras:

```
$n_i=2\cdot\alpha^3$
```
$n_i = 2 \cdot \alpha^3$

```
\(n_i=2\cdot\alpha^3\)
```
$n_i = 2 \cdot \alpha^3$

```
\begin{math}n_i=2\cdot\alpha^3\end{math}
```
$n_i = 2 \cdot \alpha^3$

2. Modo matemático de despliegue, se utiliza para escribir expresiones matemáticas que no son parte de un parrafo dentro de un recuadro. Esto se logra utilizando cualquiera de las siguientes formas:

```
\[\n_i=2\cdot\alpha^3\]
```
$$n_i=2\cdot\alpha^3$$

```
\begin{displaymath}
    n_i=2\cdot\alpha^3
\end{displaymath}
```
$$n_i=2\cdot\alpha^3$$

```
$$n_i=2\cdot\alpha^3$$
```
$$n_i=2\cdot\alpha^3$$

3.6.1. Exponentes

En las expresiones matemáticas podemos agregar exponentes colocando el símbolo `^` seguido del carácter que queremos que aparezca como exponente o seguido de varios símbolos entre llaves. Las llaves sirven para indicar dónde empieza y termina el exponente, ya que si no lo indicamos LaTeX considerará únicamente al primer carácter que sigue a `^` . Como se muestra a continuación:

```
$a^{n + m} b^m c^n$
```
$a^{n+m}b^m c^n$

```
$a^{b^m}$
```
a^{b^m}

```
$a^nb^{2n}c^{3n}$
```
$a^nb^{2n}c^{3n}$

```
$a^4b$
```
a^4b

3.6.2. Subíndices

Funcionan de la misma manera que los exponentes, a excepción de que se utiliza el símbolo `_` en lugar de `^`. Las mismas reglas aplican para subíndices que para exponentes. Como se ilustra a continuación:

```
$a_{n + m} b_m c_n$
```
$a_{n+m}b_m c_n$

```
$a_{b_m}$
```
a_{b_m}

```
$a_nb_{2n}c_{3n}$
```
$a_nb_{2n}c_{3n}$

```
$a_4b$
```
a_4b

3.6.3. Ejercicio

En la sección Ejercicios del archivo `ejemplo1.tex` realiza lo siguiente:

- Crea dos expresiones matemática con exponentes y subíndice utilizando modo matemático en línea.
- Crea dos expresiones matemática con exponentes y subíndice usando modo matemático de despliegue.

3.7. Figuras

Lo primero que debemos hacer es incluir el paquete `graphicx` en el preámbulo del documento y la ruta de la carpeta que contiene todas las imágenes que

vamos a utilizar, de esta forma sólo debemos agregar el nombre de la imagen cuando incluyamos una imagen como se muestra a continuación:

```
\usepackage{graphicx}
\graphicspath{{./img/}}
```

Con este paquete podemos utilizar el comando `\includegraphics`, el cual nos permite incluir una imagen. De la siguiente forma:

```
\includegraphics[opciones]{imagen}
```

Entre corchetes van las opciones, las cuales indican cómo debe mostrarse la imagen. A continuación, indicamos entre llaves la ruta donde está guardada la imagen. Las principales opciones que podemos incluir son:

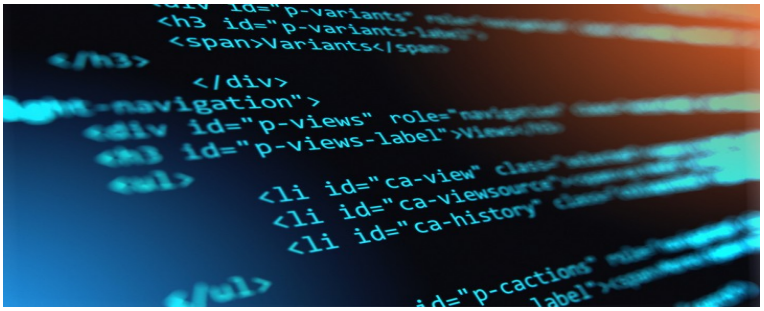
- `width` - Anchura de la imagen.
- `height` - Altura de la imagen.
- `scale` - Aumentar o disminuir el tamaño de la imagen.
- `angle` - Ángulo de rotación con el que debe aparecer la imagen.

La anchura y altura pueden definirse en distintas unidades, algunas de las unidades más comúnes son:

- `in` - Pulgadas.
- `cm` - Centímetros.
- `mm` - Milímetros.
- `pt` - Puntos.

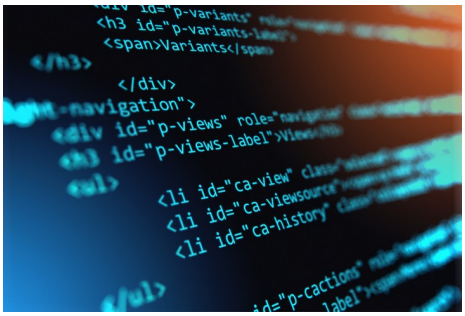
Por ejemplo, podemos insetar una imagen con un ancho de 10 cm y una altura de 4 cm mediante:

```
\includegraphics[width=10cm, height=4cm]{Codigo.png}
```



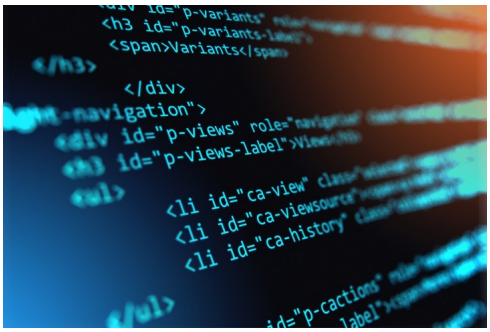
También podemos definir la anchura de la figura con relación a la longitud del texto. Esto se hace mediante el comando `\linewidth`, como se muestra a continuación:

```
\includegraphics[width=0.5\linewidth]{Codigo.png}
```



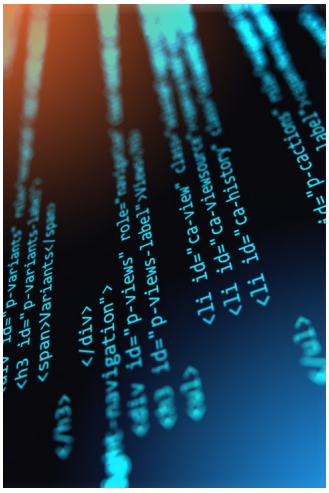
La opción `scale` cambia el tamaño de la imagen respecto a su tamaño original. Por ejemplo:

```
\includegraphics[scale=0.25]{Codigo.png}
```



Por último, podemos utilizar la opción `angle` para rotar la imagen un cierto ángulo:

```
\includegraphics[scale=0.25, angle=90]{Codigo.png}
```



3.7.1. Posición de la imagen

Si queremos controlar la posición de la figura, así como agregar una descripción o hacer referencia a ella en el texto, debemos incluir la imagen dentro de un entorno `figure`. De la siguiente manera:

```
\begin{figure}[h]
    \includegraphics[width=0.5\linewidth]{Codigo.png}
\end{figure}
```

Podemos indicar donde queremos que se muestre la figura, las principales opciones son:

- `h` (here) - Mostrar la imagen aproximadamente en el mismo lugar donde se inserta.
- `H` (Here) - Mostrar la imagen en el mismo lugar donde se inserta.
- `b` (bottom) - Mostrar la imagen en la parte inferior de la página.
- `t` (top) - Mostrar la imagen en la parte superior de la página.
- `p` (page) - Mostrar la imagen en una página aparte.

También es posible combinar distintas opciones de posición, por ejemplo:

```
\begin{figure}[bt]
    \includegraphics[width=0.5\linewidth]{Codigo.png}
\end{figure}
```

Dentro del entorno `figure` podemos definir una descripción o leyenda para que aparezca debajo de la imagen utilizando el comando `\caption`:

```
\begin{figure}
    \centering
    \includegraphics[width=0.5\linewidth]{Codigo.png}
    \caption{Descripcion de la imagen.}
\end{figure}
```

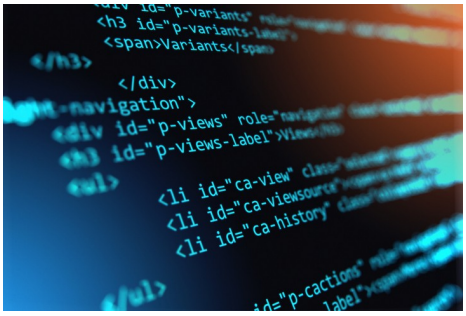


Figura 4: Descripcion de la imagen.

También podemos definir una etiqueta mediante el comando `\label` para poder referenciar la figura en alguna parte del documento, por ejemplo:

```
\begin{figure}[h]
    \centering
    \includegraphics[width=0.5\linewidth]{Codigo.png}
    \caption{Codigo en html.}
    \label{fig:codigo}
\end{figure}
```

En la figura 5 podemos ver líneas de código en HTML.

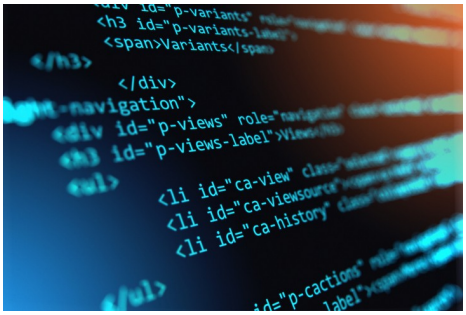


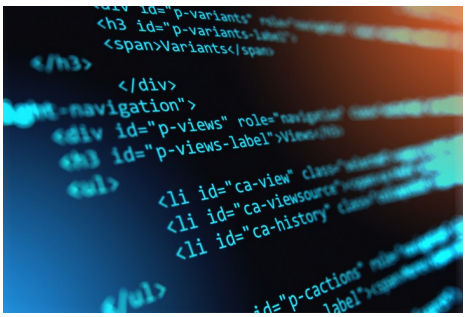
Figura 5: Codigo en html.

Además, dentro del entorno `figure` podemos indicar si la figura debe mostrarse centrada, alineada a la izquierda o alineada a la derecha. Para ello incluimos el comando correspondiente antes de `\includegraphics`:

- `\centering` - Figura centrada.
- `\raggedleft` - Figura alineada a la derecha.
- `\raggedright` - Figura alineada a la izquierda.

Por ejemplo, podemos alinear la imagen a la izquierda de la siguiente forma:

```
\begin{figure}[h]
    \raggedleft
    \includegraphics[width=0.5\linewidth]{Codigo.png}
\end{figure}
```



3.7.2. Ejercicio

En la sección Ejercicios del archivo `ejemplo1.tex` realiza lo siguiente:

- Descarga una imagen del logo de Linux.
- Guarda la imagen en una carpeta llamada img y agrega la ruta de la carpeta en el preámbulo del documento.
- Inserta la imagen utilizando includegraphics: con un ángulo de 180 grados y una escala de 0.80.
- Inserta la imagen usando el entorno figure: con scale 0.5, posición H, descripción breve y etiqueta fig:figuraLinux.

3.8. Tablas

Uno de los entornos más utilizado para crear tablas es `tabular`. Los comandos necesarios para generar una tabla son:

```
\begin{tabular}{l c}
    \bf{Lenguaje} & \bf{Programas} \\
    Java & 4 \\
    Python & 10 \\
    Haskell & 3
\end{tabular}
```

Lenguaje	**Programas**
Java	4
Python	10
Haskell	3

Después de abrir el entorno `tabular` debemos definir el número de columnas. Esto se hace escribiendo entre llaves una letra para cada columna, esta letra indica si el contenido debe ser alineado a la izquierda (`l`), a la derecha(`r`) o centrado (`c`). En el ejemplo anterior definimos dos columnas, la primera alineada a la izquierda y la segunda centrada.

Utilizamos el símbolo `&` para separar el contenido de cada columna y escribimos `\\` para indicar el final de una fila. Por ejemplo, en la siguiente línea de código:

```
Java & 4 \\
```

en la primera columna se escribirá Java, 4 en la segunda columna y a continuación cerrará la fila y pasará a la siguiente. Se pueden agregar tantas filas como sea necesario.

Podemos incluir líneas horizontales entre dos filas, para ello debemos utilizar el comando `\hline` después de la doble barra invertida `\\`. Como se muestra a continuación:

```
\begin{tabular}{l c}
    \bf{Lenguaje} & \bf{Programas} \\
    \hline
    Java & 4 \\
    \hline
    Python & 10 \\
    \hline
    Haskell & 3 \\
    \hline
\end{tabular}
```

Lenguaje	**Programas**
Java	4
Python	10
Haskell	3

También es posible agregar líneas verticales entre las distintas columnas. Estas deben crearse al definir las columnas. Por ejemplo, si queremos dos columnas centradas con líneas verticales entre ellas escribimos:

```
\begin{tabular}{c | c}
    \bf{Lenguaje} & \bf{Programas} \\
    Java & 4 \\
    Python & 10 \\
    Haskell & 3 \\
\end{tabular}
```

Lenguaje	**Programas**
Java	4
Python	10
Haskell	3

Con el mismo formato es posible crear tablas con líneas verticales y horizontales, incluso podemos crear líneas dobles:

```
\begin{tabular}{c | c}
    \hline\hline
    \bf{Lenguaje} & \bf{Programas} \\
    \hline\hline
    Java & 4 \\
    Python & 10 \\
    Haskell & 3 \\
    \hline\hline
\end{tabular}
```

Lenguaje	**Programas**
Java	4
Python	10
Haskell	3

Podemos definir la posición en la que aparece la tabla dentro del documento, cuenta con las mismas opciones de posición que las figuras. Para ello es necesario incluir la tablar dentro del entorno `table`.

```
\begin{table}[H]
\centering
\begin{tabular}{c|c}
    \hline
    \bf{Lenguaje} & \bf{Programas} \\
    \hline
    Java & 4 \\
    Python & 10 \\
    Haskell & 3 \\
    \hline
\end{tabular}
\caption{Numero de programas}
\label{tab:programas}
\end{table}
```

Lenguaje	**Programas**
Java	4
Python	10
Haskell	3

Tabla 3: Numero de programas

Al igual que las figuras, también podemos agregar una descripción y una etiqueta que nos permita hacer referencias en alguna parte del documento.

3.8.1. Ejercicio

En la sección Ejercicios del archivo `ejemplo1.tex` realiza lo siguiente:

- Crea una tabla con dos columnas y tres filas utilizando el entorno table, con líneas verticales y horizontales, además, debes agregar una descripción y una etiqueta.
- Crea una tabla con tres columnas y dos filas utilizando el entorno table, con una línea horizontal después de la primera y última fila, además, la tabla debe de tener posición b.

4. Compilar documento en Emacs

Para compilar un documento LaTeX en Emacs debemos utilizar el siguiente comando:

```
C-c C-c
```

Si no hemos guardado los últimos cambios, el buffer nos va a preguntar `Save file \/home/usuario/ruta/ejemplo.tex"?(y/n)`, debemos de responder `y` para que LaTeX trabaje con la última versión. Posteriormente, en el buffer nos debe aparecer `Command (default LaTeX):` y tenemos que presionar la tecla Enter, para que nuestro archivo se compile. Una vez que haya terminado, nos debe aparecer en el buffer:

```
LaTeX succesfully compiled (n) pages
```

Donde `n` va a ser el número de páginas de nuestro documento. Para visualizar el archivo compilado debemos volver a teclear el comando:

```
C-c C-c
```

Esta vez el mensaje en el buffer va a ser `Command (default View):` y debemos presionar la tecla Enter, para que se abra en una ventana nueva el resultado de la compilación.

5. Bibliografía

:
Bibliografía consultada para la realización de estas notas:

1. Manual de supervivencia
2. Tutorial de LaTeX en línea
3. CTAN