Introducción breve a LaTeX

José Cuevas Barrientos
https://github.com/JoseCuevasBtos/apuntes-tex

26 de junio de 2021

Índice general

Capítulo 1

Introducción

Este texto pretende ser introductorio y en general sólo abarca cosas que creo que un lector cualquiera podría necesitar a la hora de usar LaTeX. Por su naturaleza, sólo se roza la superficie de lo que es posible en este lenguaje, si lo que busca es una documentación más detallada revise el párrafo de **Más allá...** al final de la sección §2.5.

Respecto de la instalación de LaTeX, para ello están los compiladores de MikTeX en Windows, MacTeX en Mac OS y TeXLive en Linux. Los editores recomendados son TeXMaker y overleaf (en línea). Para usuarios de Vim, recomiendo usar el plugin `vimtex`.

Historial de versiones

Fechas en formato DD-MM-AA.

06-08-20 Publicación original como artículo.

20-10-20 Corrección de errores ortográficos.

26-06-21 Actualización a formato libro.

Capítulo 2

Haciendo un documento básico desde cero

En primera vamos a ver la estructura y características generales de LaTeX. A diferencia de MS Word, LaTeX no es un generador WYSIWYG[1], sino que se escribe mediante texto en un archivo de texto y un lenguaje de marcado. No os confundas, LaTeX **no** es un lenguaje de programación, y no se parece a uno, sino que es como escribir texto común con comandos entre medio.

2.1. "Hola mundo"

Los comandos suelen tener esta estructura `\comando{opciones}` donde la información obligatoria para el comando suele ir entre llaves {} a los que llamaremos *campos obligatorios*, mientras que la información opcional suele ir entre corchetes `[]`. Todo documento comienza así:

```
\documentclass[opciones]{tipo de documento}
```

Por el momento en las opciones incluiremos el tamaño normal de letra que es 10, 11 o 12pt y el tipo de documento será `article` o `book` según lo que se desee escribir.

Luego viene un comando de la forma:

```
\begin{document}
Hola mundo.
\end{document}
```

[1] en. *what you see is what you get*; lo que ves es lo que obtienes.

Varios comandos tendran la misma estructura, a estos les llamaremos *entornos* y a la parte de `document` su "tag". El entorno `document` señala todo lo perteciente al contenido del archivo. Todo lo que esté fuera de este entorno (en particular antes de él) le diremos la *cabecera* del documento. Debería compilar este archivo básico para corroborar que su instalación de LaTeX ha sido exitosa. Varios programas (como TeXMaker) tienen compiladores incluidos, pero si se interesa en hacerlo por la terminal puede ver el apéndice.

Paquetes. Igual que los lenguajes de programación permiten añadir módulos para añadir funciones (como `<stdio.h>` en `C`, o `numpy` en `Python`), LaTeX ocupa paquetes que se incluyen en la cabecera, de momento añadiremos dos:

```
\usepackage[spanish]{babel}
\usepackage[utf8]{inputenc}
```

El primero le indica a LaTeX que estamos escribiendo en español, para traducir texto del documento como escribir "Capítulo" en lugar de "Chapter". El segundo le dice a LaTeX que lea las tildes y otros caracteres especiales (e.g. á, ñ, ç, etc.) sin problema.

Título. (Casi) todo documento tiene cosas como título, autor y fecha, para incluirlos en LaTeX debe ocupar correspondientemente los siguientes comandos en la cabecera:

```
\title{Mí título interesante}
\author{Yo}
\date{\today}
```

Nótese que en la fecha utilice el comando `\today`, este lee la fecha del día en el computador y el paquete de `babel` lo traduce al español, pero puede ocupar cualquier otra que se le antoje. Dentro del documento y antes del resto del texto escriba este comando para generar el título estándar de LaTeX:

```
\maketitle
```

Secciones. Todo texto suele dividirse en secciones. LaTeX también, y además las enumera automáticamente. En un artículo la mayor división es una sección, luego una subsección y luego una sub-subsección:

```
\chapter{Esta es mi sección}
\section{Esta mi subsección}
\subsubsection{Esta mi sub-subsección}
```

Un libro también admite todos los comandos anteriores, pero la mayor distinción es un capítulo `\chapter{...}` y se pueden agrupar los capítulos en

partes `\part{...}`, donde las últimas se enumeran con números romanos; pero a diferencia de los capítulos no son obligatorios.

Cortes de línea. Cuando hay un salto entre punto aparte y el otro párrafo eso se dice un *corte de línea*, lo que uno está acostumbrado a hacer con la tecla `Enter` de vuestros teclados. No obstante en el código de LaTeX puede notar que no genera efecto alguno, esto se debe a que en los lenguajes de marcado se da esta precaución para evitar líneas excesivamente largas en el código. Para hacer este corte hay varias formas: `\\` suele ser el más común, y no genera sangría en la línea contigua. Para hacer un corte con sangría puede usar dos veces `Enter` en el código. Un efecto similar es el de usar `\par`. La sangría se puede añadir con `\indent` y quitar con `\noindent` de ser necesario. Por último, si necesita forzar un corte de línea, para evitar la sobrecarga de caractéres por ejemplo, puede usar `\break`.

```
Texto de ejemplo.\break
Texto de ejemplo.\\
Texto de ejemplo.\par
Texto de ejemplo.
```

Texto de ejemplo.
Texto de ejemplo.
Texto de ejemplo.

Texto de ejemplo.

Caracteres especiales. Sabemos que los comandos se inician con `\`, por ende, ¿cómo se escribe dicho caracter de ser necesario en LaTeX? Esta clase de caracteres se dicen *especiales* pues cumplen una función por si solos y se pueden escribir así:

`$\backslash$`	\	`\_`	_
`\%`	%	`\#`	#
`\$`	$	`\&`	&
`\{\}`	{}		

Además debe tener en consideración que las comillas en LaTeX son también distintas para escribir algo “así” se requiere ``‘‘así’’``. Donde las dos primeras son tildes graves y las últimas son apostrofes.

2.2. Formato de texto

Estilos. Uno suele cambiar el formato, e.g. a **negritas**, o *cursivas*, para ello LaTeX ocupa:

El segundo tipo de comando es lo que se dice un *modificador*, al usarlo

`\textup{Derecho}`	`\upshape`	Derecho
`\textit{Cursivas}`	`\itshape`	*Cursivas*
`\textsl{Inclinado}`	`\slshape`	*Inclinado*
`\textsc{Versallitas}`	`\scshape`	VERSALLITAS
`\textmd{Mediano}`	`\mdseries`	Mediano
`\textbf{Negritas}`	`\bfseries`	**Negritas**
`\textrm{Romano}`	`\rmfamily`	Romano
`\textsf{Sans-serif}`	`\sffamily`	Sans-serif
`\texttt{Máquina}`	`\ttfamily`	`Máquina`

modifica todo el entorno[2], e.g., `{\scshape Texto de Ejemplo}` TEXTO DE EJEMPLO.

Tamaños. Para los tamaños de letra puedes usar un modificador o un entorno, ambos ocupan el mismo tag:

tiny scriptsize footnotesize small normalsize
large Large LARGE huge Huge

Cambiar el tamaño de letra por defecto (de 10 a 12pt por ejemplo) afecta también los otros tamaños de letra.

Alineación. LaTeX permite de dos formas el cambio de alineación de texto:

Alineación	Entorno	Modificador
Izquierda	`flushleft`	`\raggedright`
Derecha	`flushright`	`\raggedleft`
Centro	`center`	`\centering`

Colores. Para admitir colores extra se debe importar el siguiente paquete en la cabecera:

```
\usepackage{xcolor}
```

Para usarlo puedes escribir: `{\color{red} texto en rojo}` texto en rojo o `\textcolor{blue}{texto en azúl}` texto en azúl. Los nombres de colores por defecto son:

[2]En realidad modifica el entorno restante, ya que el contenido dentro del mismo entorno que viene ántes del modificador no se ve afectado.

black blue brown cyan darkgray gray green lightgray lime magenta olive orange pink purple red teal violet white yellow

Para poner colores en el fondo se utiliza `\colorbox{cyan}{Así}` Así. Además puedes utilizar un porcentaje de un color, por ejemplo, si sólo queremos usar el 50 % de negro podemos escribir `\colorbox{black!50}{esto}` esto, y si queremos usar 50 % azúl y el resto rojo `\colorbox{blue!50!red}{haga esto}` haga esto.

Por sobre eso, usted puede definir nuevos colores en la cabecera:

```
\definecolor{ejemplo}{HTML}{00c89c}
```

Donde lo de `HTML` indica que la forma de definir colores es mediante el llamado código hexadecimal o `hex` para acortar. Este suele ser el método más común de definir colores, en línea lo reconocerá pues ocupan el prefijo '`#`', e.g., #00c89c.

Varias columnas. Si usted quisiera, de forma global, tener un documento escrito en dos columnas, sólo basta agregar la opción `twocolumn` en `\documentclass`. No obstante, si desea hacerlo localmente, esto es, en una sola parte del documento, debe importar el paquete:

```
\usepackage{multicol}
```

y luego usar el entorno `multicols` seguido del número de columnas:

```
\begin{multicols}{2}
    Muchos años después, frente al pelotón de
       fusilamiento, el coronel Aureliano Buendía...
\end{multicols}
```

Muchos años después, frente al pelotón de fusilamiento, el coronel Aureliano Buendía había de recordar aquella tarde remota en que su padre lo llevó a conocer el hielo. Macondo era entonces una aldea de veinte casas de barro y cañabrava construidas a la orilla de un río de aguas diáfanas que se precipitaban por un lecho de piedras pulidas, blancas y enormes como huevos prehistóricos.

2.3. Notas, listas, figuras y tablas

Notas. LaTeX admite dos tipos comúnes de notas: al pie de la página y al margen de ella. La primera se especifica con el comando `\footnote{`

`Esta es una nota al pie de página.}`[3] mientras que las notas de margen se hacen con `\marginpar{Y ésta, una nota al margen.}`. Para revertir el lugar se puede usar `\reversemarginpar{...}`.

Y ésta, una nota al margen.

Para más información y mejor manejo de las notas al margen se recomienda revisar el paquete `marginnote`.

Listas. Se dividen en tres: numeradas, no numeradas y descripciones (similar a un glosario o diccionario). Todas se definen por un entorno y sus elementos se diferencian por el comando `\item`. Para una lista enumerada el entorno utiliza el tag `enumerate`, las no numeradas el tag `itemize` y las descripciones `description`. En las descripciones, la palabra (o frase) de título se denota en el campo opcional de `\item[Aquí...]`. Por ejemplo:

```
\begin{enumerate}
    \item Naranjas.
    \item Manzanas:
        \begin{itemize}
            \item Rojas.
            \item Verdes.
            \item Amarillas.
        \end{itemize}
    \item Bananas.
\end{enumerate}
\begin{description}
    \item[Fruta] Fruto comestible de ciertas plantas cultivadas
        ; e.g., la pera, la guinda, la fresa, etc.
    \item[Verdura] Hortaliza, especialmente la de hojas verdes.
\end{description}
```

1. Naranjas.
2. Manzanas:
    - Rojas.
    - Verdes.
    - Amarillas.
3. Bananas.

Fruta Fruto comestible de ciertas plantas cultivadas; e.g., la pera, la guinda, la fresa, etc.

Verdura Hortaliza, especialmente la de hojas verdes.

[3] Esta es una nota al pie de página.

Figuras y tablas. Por defecto, LaTeX no permite importar imágenes al documento por lo que se requiere importar el paquete:

```
\usepackage{graphicx}
```

Luego para importar la figura (asumiendo que la imágen está en la misma carpeta que el archivo) se utiliza el siguiente comando:

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

En las opciones usualmente van especificaciones sobre el tamaño, verá que sin ella la figura se importa al máximo tamaño para el cual no pierde detalle, para importarlo digamos a la mitad de su tamaño real puede usar `scale=.5`, pero usualmente es mejor especificar directamente ya sea el largo (`width`) o el alto (`height`) del archivo que debe hacerse en `cm`, `mm` o `in` (pulgadas). También podemos decirle que sea tan larga como el tamaño del texto usando `width=\textwidth`.

Para que la imagen este apartada del texto se incluye todo dentro de un entorno `figure` cuyo único propósito es ese de decirle a LaTeX que trate la imagen individualmente. Dentro de él se recomienda centrar el texto (o en este caso el contenido) y al final del entorno se recomienda añadir alguna forma de corte de línea para mejorar el formato.

Posicionamiento. Para saber donde ubicar la imagen, LaTeX utiliza los siguientes símbolos para abreviar:

`h`	Aproximadamente en el lugar donde se ubica en el código.
`t`	En el tope de la página siguiente.
`b`	En el fondo de la página actual.
`p`	En la siguiente página, destinada exclusivamente para figuras.
`!`	Fuerza la posición indicada.

Descripciones. Además las imágenes poseen descripciones cortas, estas pueden agregarse en LaTeX dentro de un entorno `figure` con el comando `\caption{Mi descripción.}`, veamos un ejemplo donde se utiliza todo lo anterior:

```
\begin{figure}[!h]
    \centering
    \includegraphics[width=10cm]{fractal.png}
    \caption{Un fractal}
\end{figure}
```

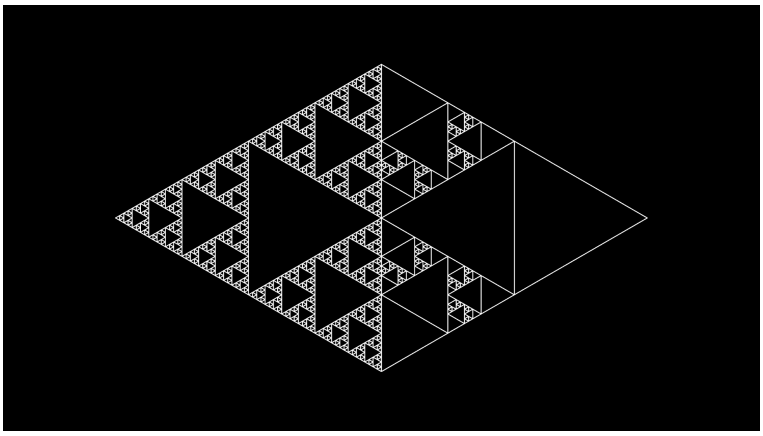


Figura 2.1: Un fractal

Tablas. Las tablas se definen así:

```
\begin{tabular}{formato}
    ...
\end{tabular}
```

Donde el *formato* se define por lo siguiente:

`l`	Celdas hacia la izquierda.
`c`	Celdas centradas.
`r`	Celdas hacia la derecha.
`p{`*longitud*`}`	Párrafo de cierta longitud.
`m{`*longitud*`}`	Como `p`, pero centrado verticalmente†.
`b{`*longitud*`}`	Como `p`, pero verticalmente al fondo†.

Los marcados por † requieren del paquete `array`. Las celdas en una columna se separan por el símbolo `&` y las columnas se separan por un corte de línea. Además `\hline` genera una línea horizontal lo que puede ser útil. Tal como uno pone las figuras en un entorno `figure`, existe el entorno `table` para tablas que sirve para posicionar y añadir descripciones; funciona exactamente igual.

```
\begin{table}[!h]
    \centering
    \begin{tabular}{|lcc|}
        \hline
        País & Contagios totales & Fallecidos totales \\
```

```
        \hline \hline
        Argentina & 201,919 & 3,667 \\
        Chile     & 361,493 & 9,707 \\
        Colombia  & 317,651 & 10,650 \\
        Peru      & 428,850 & 19,614 \\
        \hline
    \end{tabular}
    \caption{Estado Covid-19 (3 de agosto de 2020).}
\end{table}
```

País	Contagios totales	Fallecidos totales
Argentina	201,919	3,667
Chile	361,493	9,707
Colombia	317,651	10,650
Peru	428,850	19,614

Cuadro 2.1: Estado Covid-19 (3 de agosto de 2020).

Se puede agrupar un formato y repetir n veces con la abreviación `*{`*n*`}{`*formato*`}`. En el ejemplo anterior podríamos reemplazar la `cc` por `*{2}{c}`. Una celda puede usar el tamaño de varias dentro de la misma fila usando el comando `\multicolumn{`*num*`}{`*form*`}{`*contenido*`}` donde *num* es el número de columnas que abarca, y *form* el formato de ella, e.g:

```
\begin{table}[!h]
    \centering
    \begin{tabular}{|l|cc|}
        \hline
        País & Contagios totales & Nuevos contagios \\
        \hline \hline
        \multicolumn{3}{|c|}{\sffamily América} \\
        \hline
        EEUU        & 4,851,407 & +38,379 \\
        Brasil      & 2,736,298 & +2,621  \\
        \hline
        \multicolumn{3}{|c|}{\sffamily Europa} \\
        \hline
        Reino Unido & 305,623   & +928    \\
        Italia      & 248,229   & +159    \\
        \hline
    \end{tabular}
\end{table}
```

Con el paquete `multirow` puedes combinar filas también mediante el comando `\multirow{`*num*`}{`*longitud*`}{`*contenido*`}` (puedes usar `*` en el lugar de

País	Contagios totales	Nuevos contagios
América		
EEUU	4,851,407	+38,379
Brasil	2,736,298	+2,621
Europa		
Reino Unido	305,623	+928
Italia	248,229	+159

la longitud para ocupar todo lo disponible). Y puedes hacer líneas horizontales parciales desde el inicio de la i-ésima celda hasta el final de la j-ésima celda con `\cline{`i`-`j`}`, e.g.:

```
\begin{table}[!h]
    \centering
    \begin{tabular}{|l|lcc|}
        \cline{2-4}
        \multicolumn{1}{c|}{}    & País   & Contagios totales &
            Nuevos contagios \\
        \hline
        \multirow{2}{*}{América} & EEUU   & 4,851,407         &
            +38,379 \\
        \cline{2-4}
        {}                       & Brasil & 2,736,298         &
            +2,621  \\
        \hline
    \end{tabular}
\end{table}
```

	País	Contagios totales	Nuevos contagios
América	EEUU	4,851,407	+38,379
	Brasil	2,736,298	+2,621

Tablas entre páginas. Por defecto LaTeX no rompe una tabla entre páginas, sin importar cuan larga sea, para poder usar un *tabla larga* se debe importar el paquete `longtable` que provee el entorno homónimo que funciona exactamente igual que `tabular`.

2.4. Bibliografías, referencias cruzadas e índices

LaTeX es bastante popular por su forma de tratar bibliografías, por lo cual vamos a enseñar como: En primer lugar, todas las referencias bibliográficas

se guardan en otro archivo externo de extensión `.bib`. Para incluir una bibliografía debemos añadir el siguiente comando:

```
\usepackage[backend=biber]{biblatex}
\addbibresource{mis-referencias.bib}
```

Y para compilarlo hay que compilar el documento una primera vez, luego compilar la bibliografía con *biber* y luego una segunda vez. Esto se debe a que la primera vez le dice a LaTeX que referencias crear, luego el compilador las crea con su respectivo formato y la última las incluye.

Las referencias se pueden buscar facilmente en páginas como `https://scholar.google.com/` y luego seleccionando obtener el formato en BibTeX o similar. En general toda referencia suele verse así:

```
@book{gauss1966disquisitiones,
    title={Disquisitiones arithmeticae},
    author={Gauss, Carl Friedrich},
    volume={157},
    year={1966},
    publisher={Yale University Press}
}
```

Donde es bastante claro como funciona, la palabra `book` indica que la referencia es un libro, `gauss1966disquisitiones` es lo que le decimos una *etiqueta*, i.e., una cadena de signos sin espacios que se utiliza para citar dentro del documento, luego el resto es claro. Lo único importante a saber es que en el campo del autor se recomienda anotar apellido, luego coma, luego nombre; y si hay más de uno separarlos por la palabra `and`, si hay muchos y se quiere utilizar algo para decir "y otros" (et al. en latín) se utiliza un `others` (Albert Einstein et al. sería `Einstein, Albert and others` para LaTeX).

Usualmente se utiliza un solo archivo general para tener todas las referencias allí, luego, utilizando el ejemplo anterior, uno citaría un documento así:

```
\cite{gauss1966disquisitiones}
```

Si se quiere que la cita este al pie de página uno ocupa `\footcite{...}` y puedes citar varios documentos juntos separando las etiquetas por comas. Entre las referencias sólo aparecerán los trabajos citados. Para incluir un trabajo sin citarlo en la bibliografía debemos usar el comando `\nocite{...}` donde las mismas reglas se aplican. Si se quiere incluir a todas las referencias puedes escribir `\nocite{*}`. Para imprimir las bibliografías se ocupa este comando:

```
\printbibliography
```

Sub-bibliografías y palabras clave. No obstante, un problema que puede surgir al tratar bibliografías es que toman un capítulo (o sección si de un artículo se trata) entero cuando se imprimen, y un formato interesante es el de ciertos libros que usan bibliografía para cada capítulo, lo que por el momento parece una tarea imposible. Un problema similar es si queremos incluir muchos artículos a la vez, pero no todos y sólo queremos incluir los de un cierto tema. Para ambos la solución implica introducir el mismo concepto: palabras clave.

Dentro de la definición de libros en BibTEX se añade la categoría `keywords` (sí, plural) donde especifica las palabras clave separadas por comas. Tomemos el ejemplo del libro de Gauss y supongamos que le añadimos `keywords ={arithmetic}` donde tenemos varias referencias marcadas sobre el tema de `arithmetic` luego este comando:

```
\printbibliography[keyword={arithmetic}]
```

Imprimirá sólo dichos artículos. Para el problema de las sub-bibliografías añada la opción de `subbibintoc`. Y para cambiarle el nombre puede usar la opción, `heading={`*`mi título`*`}`.

Referencias cruzadas. Una de las ventajas de LATEX es que como genera automáticamente la numeración para las secciones, figuras y ecuaciones, se pueden referenciar de forma bastante sencilla, junto a uno de esos entornos se agrega el comando `\label{`*`etiqueta`*`}` y luego se llama con `\ref{`*`etiqueta`*`}` (puede requerir una compilación doble para funcionar adecuadamente), e.g:

```
\section{Bibliografías, referencias cruzadas e índices}
\label{sec:crossref}
--------------------------------------------------
...similar a lo dicho en la sección \ref{sec:crossref}.
```

...similar a lo dicho en la sección 2.4.

La parte de `sec:` es una costumbre decorativa, así como se suele usar `eq:` como prefijo para las ecuaciones, `fig:` para las figuras, etc. No es necesaria pero puede servir para ordenarse en el código fuente.

Tablas de contenidos. La tabla de contenidos no puede ser más fácil de implementar, sólo basta con el comando:

```
\tableofcontents
```

Además se pueden usar `\listoftables` y `\listoffigures` para imprimir índices de las tablas y las figuras del documento (sólo ocupa las entradas con descripciones, puedes hacer descripciones vacías para que esten numeradas).

Hipervínculos. Si ha descargado este documento notará que puede *hacer click* en la referencia cruzada anterior que lo lleva a la página de inicio de la sección, lo mismo ocurre con todas las entradas de la tabla de contenidos y con los enlaces web, para ello es tan sencillo como agregar el siguiente paquete:

```
\usepackage{hyperref}
```

Para añadir las entradas de las secciones al índice del pdf, agrega las opciones `bookmarks=true` y `bookmarksnumbered=true`. Aquí puedes añadir urls con el comando `\url{`*`url`*`}` y `\href{`*`url`*`}{`*`texto`*`}`; y referencias cruzadas extra con `\hyperref{`*`etiqueta`*`}{`*`texto`*`}`.

2.5. Configurar mi documento

Esta sección es opcional, pero se recomienda bastante para mejorar su experiencia y efeciencia con LaTeX:

(Re) definir comandos. Además de los comandos actuales, se pueden definir nuevos con `\newcommand`, este va seguido del nombre del comando y luego del uso, e.g:

```
\newcommand{\licencia}{Éste documento está bajo la
   licencia xyz (2020).}
-----------------------------------------------------
\licencia
```

Éste documento está bajo la licencia xyz (2020).

Si quiere añadirle opciones al comando entonces debe poner un campo opcional con el número de opciones y luego referirse a ellas dentro de la definición como `#1`, `#2` y así:

```
\newcommand{\licencia}[2]{Éste documento está bajo la
   licencia #2 (#1).}
-----------------------------------------------------
\licencia{2019}{uvw}
```

Éste documento está bajo la licencia uvw (2019).

Además también puedes pasar argumentos opcionales, que toman el lugar del primer argumento si están dados y cuyo valor por defecto queda definido en el segundo campo opcional:

```
\newcommand{\licencia}[2][2020]{Éste documento está bajo
   la licencia #2 (#1).}
-----------------------------------------------------
```

```
\licencia{abc}\\
\licencia[1995]{def}
```

Éste documento está bajo la licencia abc (2020).
Éste documento está bajo la licencia def (1995).

Si un comando ya existe, LaTeX tirará un error al tratar de sobreescribirlo, para poder hacerlo debes usar \renewcommand, que funciona exactamente igual. Éste último también tira error si se trata de redefinir un comando que no ha sido definido antes.

Definir entornos. Si se quiere definir entornos, se ocupa \newenvironment y \renewenvironment que funcionan exactamente igual, sólo que en lugar de poner un comando pones el tag del entorno y al final se ocupan dos campos obligatorios que determinan el código a ejecutar ántes y después del contenido del entorno. Pero todo uso de argumentos extra van en la primera parte y no pueden ir al final, e.g:

```
\newenvironment{demo}[1][$\bullet$]{#1 {\scshape
   Demostración:} }{\par\noindent}
-------------------------------------------------
\begin{demo}
    Esto se deduce de que el triángulo sea isóceles.
\end{demo}
\begin{demo}[(?)]
    Esto es obvio.
\end{demo}
```

• Demostración: Esto se deduce de que el triángulo sea isóceles.
(?) Demostración: Esto es obvio.

Plantillas. Si usted ha seguido el texto hasta ahora, probando de todo lo que se menciona, es probable que la cabecera de su documento tenga hartas líneas y genere un poco de confusión, sin contar con el hecho de que si piensa hacer otro documento tendrá que copiar y pegar todo eso. Para esto existen las llamadas *plantillas* (*template* en inglés). La manera más fácil es hacer un documento general (usualmente de nombre `template.tex`) en donde guardar todos los comandos y luego importarlo con:

```
\input{template.tex}
```

Más allá... Si quiere saber más información acerca de LaTeX le recomiendo, con absoluta seriedad, **leer los manuales** de los paquetes aquí mencionados, en ellos se suele documentar de buena manera como usarlos y configurarlos de maneras más avanzadas que he omitido para acotar un

texto que ya es bastante largo. Además varia de la información la saqué de overleaf, wikibooks y de preguntas en el foro de stack exchange.

Además, en su momento, aprendí LaTeX con bastante facilidad y complitud desde La introducción no-tan-corta a LaTeX 2_ε y el libro de Alexánder Borbón y Walter Mora que es muy completo y recomendado.

Capítulo 3

Escribir matemáticas

3.1. Básico

LaTeX admite dos formas de denotar matemáticas, en modo *entre líneas* y modo *display*. El primero se escribe así `$1+1=2$` $1+1=2$, mientras que el segundo se escribe así:

```
$$1+1=2$$
```

$$1+1=2$$

Esta distinción es importante, pues LaTeX tomará ciertas decisiones para no interferir con el resto del texto, lo que hará que el modo entre líneas tenga ciertas peculiaridades. Notemos que por defecto LaTeX entiende que ha de añadir espacios entre las operaciones, además los números no se inclinan mientras que las letras sí, e.g. `$a\cdot a=a^2$` $a\cdot a=a^2$.

LaTeX también admite subíndeces con _, e.g., `$x_1 \leq x_2$` $x_1 \leq x_2$, pero hemos de tener cuidado con los índices y las potencias pues `$a^-1$` da a^-1, esto se debe a que LaTeX sólo interpreta el primer símbolo arriba, algo similar ocurre con los índices; para usar más de uno juntos se pueden agrupar con {}, e.g. `$\pi^{-1}$` π^{-1}.

Igual que con π, LaTeX incluye comandos para varios otros símbolos usuales:

Griego			
`$\alpha A$`	αA	`$\nu N$`	νN
`$\beta B$`	βB	`$\xi\Xi$`	$\xi\Xi$
`$\gamma \Gamma$`	$\gamma\Gamma$	`$o O$`	oO
`$\delta \Delta$`	$\delta\Delta$	`$\pi \Pi$`	$\pi\Pi$

`$\epsilon \varepsilon E$`	$\epsilon\varepsilon E$	`$\rho\varrho P$`	$\rho\varrho P$
`$\zeta Z$`	ζZ	`$\sigma \Sigma$`	$\sigma\Sigma$
`$\eta H$`	ηH	`$\tau T$`	τT
`$\theta \vartheta \Theta$`	$\theta\vartheta\Theta$	`$\upsilon \Upsilon$`	$\upsilon\Upsilon$
`$\iota I$`	ιI	`$\phi \varphi \Phi$`	$\phi\varphi\Phi$
`$\kappa \varkappa K$`	$\kappa\varkappa K$	`$\chi X$`	χX
`$\lambda \Lambda$`	$\lambda\Lambda$	`$\psi \Psi$`	$\psi\Psi$
`$\mu M$`	μM	`$\omega \Omega$`	$\omega\Omega$
Operaciones			
`$\cdot$`	$\cdot$	`$\times$`	$\times$
`$\pm$`	$\pm$	`$\mp$`	$\mp$
`$\div$`	$\div$	`$\cap$`	$\cap$
`$\cup$`	$\cup$	`$\neq$`	$\neq$
`$\leq$`	$\leq$	`$\geq$`	$\geq$
`$\in$`	$\in$	`$\perp$`	$\perp$
`$\mid$`	$\mid$	`$\parallel$`	$\parallel$
`$\notin$`	$\notin$	`$\subset$`	$\subset$
`$\subseteq$`	$\subseteq$	`$\supseteq$`	$\supseteq$
`$\simeq$`	$\simeq$	`$\approx$`	$\approx$
`$\wedge$`	$\wedge$	`$\vee$`	$\vee$
`$\oplus$`	$\oplus$	`$\otimes$`	$\otimes$
`$\circ$`	$\circ$	`$\sim$`	$\sim$
`$\equiv$`	$\equiv$	`$\cong$`	$\cong$
`$\to$`	$\to$	`$\implies$`	$\implies$
Delimitadores			
`$\|$`	$\|$	`$\\|$`	$\|$
`$\{\}$`	$\{\}$	`$\lfloor \rfloor$`	$\lfloor\rfloor$
`$\lceil \rceil$`	$\lceil\rceil$	`$\langle \rangle$`	$\langle\rangle$
`$\ulcorner \urcorner$`	$\ulcorner\urcorner$	`$\llcorner \lrcorner$`	$\llcorner\lrcorner$
Otros			
`$\infty$`	∞	`$\forall$`	$\forall$
`$\Re$`	$\Re$	`$\Im$`	$\Im$
`$\nabla$`	∇	`$\exists$`	$\exists$
`$\partial$`	∂	`$\nexists$`	$\nexists$
`$\emptyset$`	$\emptyset$	`$\varnothing$`	$\varnothing$
`$\wp$`	$\wp$	`$\complement$`	$\complement$
`$\neg$`	$\neg$	`$\aleph$`	$\aleph$
`$\beth$`	$\beth$	`$\gimel$`	$\gimel$
`$\square$`	$\square$	`$\surd$`	$\surd$

`$\blacksquare$`	■	`$\triangle$`	△

Para los marcados con violeta, se requieren dos nuevos paquetes:

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

Nótese que al estar separados por coma, LaTeX entiende que son dos paquetes independientes. Si no le interesan dichos símbolos aun se recomienda importar el primero.

Encontrar símbolos. En general es una tarea practicamente imposible la de aprenderse de memoria todos los símbolos en LaTeX, sin considerar el hecho de que ciertos paquetes importan cada vez más de ellos, por lo que se recomienda con creces usar este sitio web `http://detexify.kirelabs.org/classify.html` donde puedes encontrar símbolos mediante dibujos. Bajo el comando donde aparecen se especifica si el comando funciona en modo texto o modo matemático (o ambos), y arriba especificará si requiere algún paquete.

Otra observación es que si bien `|` sirve y genera el mismo efecto en modo matemático que `\mid`, el primero se recomienda como *envoltura* (similar a los paréntesis), mientras que el segundo genera espacios como para el "divide a", e.g., `$|x|; 2\mid 4$` $|x|; 2 \mid 4$. Esta distinción aplica para el resto de delimitadores.

Puntos. Pese a que no sea obvio, en LaTeX uno posee varios tipos de puntos: centrados, bajos, verticales y diagonales (del tipo $\cdot\ddots$). Para ello se escribe el punto centrado solo como `$\cdot$` como hemos visto, pero los tres juntos se escriben `$\cdots, \dots, \vdots, \ddots$` $\cdots, \dots, \vdots, \ddots$ respectivamente.

Como recomendación se usa los puntos centrados para operaciones, e.g., `$1+2+\cdots+99+100$` $1+2+\cdots+99+100$, y los bajos para listas `$\{1,2,\dots,100\}$` $\{1, 2, \dots, 100\}$.

Fracciones. Para hacer fracciones existe el comando `\frac` que funciona como:

```
$$ \frac{1}{2} + \frac{1}{3} = \frac{5}{6}. $$
```

$$\frac{1}{2} + \frac{1}{3} = \frac{5}{6}.$$

No obstante, el comando cambia si estamos en modo texto: `$\frac{22}{7}$` $\frac{22}{7}$ y también lo hace si ponemos varias fracciones juntas:

```
$$ \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{\ddots}}}} $$
```

$$\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\ddots}}}}$$

Esto se debe a que en el numerador y denominador se pasa a modo entre líneas para no ocupar demasiado espacio, en el modo entre líneas se utiliza para que las líneas se mantengan constantes en altura, no obstante, de ser necesario puede usar `\dfrac` para forzar una fracción en modo *display* y `\tfrac` para forzar modo entre líneas.

```
$$ \dfrac{1}{1 + \dfrac{1}{1 + \dfrac{1}{1 + \dfrac{1}{\ddots}}}} $$
```

$$\dfrac{1}{1+\dfrac{1}{1+\dfrac{1}{1+\dfrac{1}{\ddots}}}}$$

```
$$ \frac{2}{5} + \frac{39}{45} = \frac{57}{45} = 1\tfrac{12}{45} $$
```

$$\frac{2}{5} + \frac{39}{45} = \frac{57}{45} = 1\tfrac{12}{45}$$

Un comando similar es `\binom` que genera coeficientes binomiales, e.g., `$\binom{3}{2}$` $\binom{3}{2}$.

Raíces. Este es muy sencillo, así se describe una raíz cuadrada `$\sqrt{17}$` $\sqrt{17}$, y así una raíz n-ésima `$\sqrt[3]{8} = 2$` $\sqrt[3]{8} = 2$.

Operadores. En general a los símbolos destinados a ir entre dos caracteres como $+$ se les dice "operación binaria", por separado, un operador es una palabra o frase que se utiliza expicitamente y, por lo general, en letras romanas derechas; e.g., `$\sin x$` $\sin x$.

Los operadores se clasifican en dos: grandes o pequeños, donde ambos ocupan el mismo tamaño para su nombre, pero los grandes tienen la gracia de que sus subíndices y superíndices van directamente abajo y arriba respectivamente del nombre, por ejemplo:

```
$$ \lim_{x \to \infty} \sin^2 x + \cos^2 x = 1. $$
```

$$\lim_{x \to \infty} \sin^2 x + \cos^2 x = 1.$$

Aquí `\sin`,`\cos` son pequeños mientras que `\lim` es grande.

Los operadores no siempre tienen que tener texto, pueden poseer otros símbolos como:

```
$$ \sum_{i=0}^n i = \frac{n(n+1)}{2}, \int_a^b f(x) dx. $$
```

$$\sum_{i=0}^{n} i = \frac{n(n+1)}{2}, \int_a^b f(x)dx.$$

Cabe notar que dicha característica de los operadores grande se pierde en el modo entre líneas, e.g., `$\prod_{k=1}^n i = n!$` $\prod_{k=1}^{n} i = n!$

`$\cos$`	cos	`$\csc$`	csc
`$\exp$`	exp	`$\ker$`	ker
`$\sinh$`	sinh	`$\arcsin$`	arcsin
`$\cosh$`	cosh	`$\deg$`	deg
`$\lg$`	lg	`$\ln$`	ln
`$\arctan$`	arctan	`$\cot$`	cot
`$\hom$`	hom	`$\log$`	log
`$\sec$`	sec	`$\tan$`	tan
`$\arg$`	arg	`$\coth$`	coth
`$\dim$`	dim	`$\sin$`	sin
`$\tanh$`	tanh		
`$\liminf$`	lím inf	`$\limsup$`	lím sup
`$\inf$`	ínf	`$\sup$`	sup
`$\min$`	mín	`$\max$`	máx
`$\lim$`	lím	`$\gcd$`	gcd
`$\Pr$`	Pr	`$\det$`	det
`$\int$`	$\int$	`$\iint$`	$\iint$
`$\iiint$`	$\iiint$	`$\idotsint$`	$\int\cdots\int$
`$\oint$`	$\oint$	`$\coprod$`	$\coprod$
`$\sum$`	$\sum$	`$\prod$`	$\prod$
`$\bigcup$`	$\bigcup$	`$\bigcap$`	$\bigcap$
`$\bigvee$`	$\bigvee$	`$\bigwedge$`	$\bigwedge$

En la lista los primeros son los pequeños y los segundos los grandes.

Definir operadores. Para definir operaciones se debe utilizar el comando `\DeclareMathOperator{com}{tag}` en la cabecera. Si se declara con un `*` antes de las llaves, entonces se definirá un operador grande, e.g:

```
\DeclareMathOperator{\mcd}{mcd}
\DeclareMathOperator*{\dom}{dom}
-----------------------------------------------
$$ \mcd(21, 70) = 7,\quad \dom_V(f) = A. $$
```

$$\mcd(21, 70) = 7, \quad \operatorname*{dom}_V(f) = A.$$

Pero esto es una definición general, aquí defines un comando para todo el documento lo que puede o no ser de su agrado, si tiene un operador que desea ocupar una única vez entonces es más recomendable usar `\operatorname`; sus efectos son los mismos:

```
$$ \operatorname{mcm}(3, 16) = 48,\quad \operatorname*{
    cod}_V(f) = B. $$
```

$$\operatorname{mcm}(3,16) = 48, \quad \operatorname*{cod}_V(f) = B.$$

También podemos utilizar el anterior para redefinir comandos, por ejemplo si no nos gusta que `$\Re, \Im$` se vean así $\Re, \Im$, podemos poner esto en la cabecera:

```
\renewcommand{\Re}{\operatorname{Re}}
\renewcommand{\Im}{\operatorname{Im}}
```

Para que se vean así $\operatorname{Re}, \operatorname{Im}$.

Espacios. El ejemplo anterior también sirve para ilustrar una característica del modo matemático, la cual es que no lee los espacios singulares del código, e.g., `$a b$` genera ab. Para añadir espacios también hay comandos singulares:

`$a\!b$`	$a\!b$
`$ab$`	ab
`$a\,b$`	$a\,b$
`$a\;b$`	$a\;b$
`$a\ b$`	$a\ b$
`$a\quad b$`	$a\quad b$
`$a\qquad b$`	$a\qquad b$

Fuentes de letras. Para el modo matemático también hay fuentes de letras:

`\mathrm`	$\mathrm{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
	$\mathrm{abcdefghijklmnopqrstuvwxyz\ 1234567890}$
`\mathbf`	$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
	$\mathbf{abcdefghijklmnopqrstuvwxyz\ 1234567890}$
`\mathsf`	$\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
	$\mathsf{abcdefghijklmnopqrstuvwxyz\ 1234567890}$
`\mathbb`	$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
`\mathcal`	$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$

`\mathfrak`	𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ 𝔞𝔟𝔠𝔡𝔢𝔣𝔤𝔥𝔦𝔧𝔨𝔩𝔪𝔫𝔬𝔭𝔮𝔯𝔰𝔱𝔲𝔳𝔴𝔵𝔶𝔷 1234567890

Es común usar `\mathbb` para conjuntos numéricos famosos como `$\mathbb{N}$` $\mathbb{N}$. A veces, éstos suelen ser tan comúnes y conocidos, que suele ser útil definir un comando para ellos:

```
\newcommand{\N}{\mathbb{N}}
\newcommand{\Z}{\mathbb{Z}}
\newcommand{\Q}{\mathbb{Q}}
\newcommand{\R}{\mathbb{R}}
\newcommand{\C}{\mathbb{C}}
```

Texto en ecuaciones. Si bien la fuente `\mathrm` es la misma del texto del documento, todavía tendríamos que lidiar con los espacios del modo matemático sin hablar de que no admiten caracteres `utf-8` (i.e., cosas como `$á$` generan errores), pero LaTeX admite texto en ecuaciones, sólo basa usar `\text`, e.g. `$x=2,\text{ mientras que }y=3$` $x = 2,\text{ mientras que }y = 3$. Esto es más útil en ecuaciones en modo *display*.

3.2. Ecuaciones numeradas, alineadas, matrices y más

Ecuaciones numeradas. Hasta ahora nuestras ecuaciones no están numeradas, ni poseen un número para añadirlas, para hacerlo debes usar el entorno `equation`. Dentro de él se permiten el uso de referencias cruzadas, respecto a esto se recomienda usar `\eqref` en lugar de `\ref` por estética, aunque sus usos son iguales, e.g:

```
...y así se deduce el llamado \textit{teorema de Pitá
   goras} \eqref{eq:pitagoras}:
\begin{equation} \label{eq:pitagoras}
    a^2 + b^2 = c^2
\end{equation}
```

...y así se deduce el llamado *teorema de Pitágoras* (3.1):

$$a^2 + b^2 = c^2 \tag{3.1}$$

Como se ve, LaTeX automáticamente enumera la ecuación como 3.1, si se quiere cambiar dicho tag se puede mediante el comando `\tag`, e.g:

```
\begin{equation}
    E = mc^2 \label{eq:einstein} \tag{\heartsuit}
\end{equation}
Y \eqref{eq:einstein} es la famosa fórmula de Einstein...
```

$$E = mc^2 \tag{♡}$$

Y (♡) es la famosa fórmula de Einstein...

Además se puede quitar los tags con `\notag`, que tal vez no es útil aún, pero lo será pronto.

Ecuaciones alineadas. Es común ver documentos donde una ecuación está alineada por signos "=" y no saber como, además notará que entre `$$ ... $$` no funcionan los cortes de línea. Para ello existe el entorno `align` y `align*`. Donde las partes son como celdas de una tabla y se separan por `&`. El primero enumera cada línea, mientras que el segundo no enumera ninguna, e.g:

```
\begin{align}
    \sum_{i=1}^n i &= 1 + 2 + 3 + \cdots + (n-1) + n \notag \\
    &= (1 + n) + (2 + (n-1)) + (3 + (n-2)) + \cdots \notag \\
    &= (n+1) + (n+1) + (n+1) + \cdots \notag \\
    &= (n+1)\cdot\frac{n}{2} = \frac{n(n+1)}{2}.
\end{align}
```

$$\begin{aligned}\sum_{i=1}^{n} i &= 1 + 2 + 3 + \cdots + (n-1) + n \\ &= (1 + n) + (2 + (n-1)) + (3 + (n-2)) + \cdots \\ &= (n+1) + (n+1) + (n+1) + \cdots \\ &= (n+1)\cdot\frac{n}{2} = \frac{n(n+1)}{2}.\end{aligned} \tag{3.2}$$

Un detalle importante es que por defecto LaTeX no rompe una ecuación alineada entre páginas, sin importar cuan larga sea, si usted quiere permitir esto debe agregar este comando a la cabecera:

```
\allowdisplaybreaks
```

Ecuación multi-linea. Ésta es diferente de la anterior, en esta se usan varias líneas, pero todos bajo un mismo número, o varias ecuaciones, pero sin alinear. Para una ecuación muy larga se puede usar el entorno `multline` y para varias ecuaciones no-alineadas el entorno `gather`. El asterisco indica si van enumeradas o no:

```
\begin{multline*}
    f(x) = 59x^{11} + 86x^{10} + 22x^9 + 83x^8 + 74x^7 +
       49x^6 + 12x^5 + 83x^4\\
    + 42x^3 + 69x^2 + 47x + 93.
\end{multline*}
y
\begin{gather*}
    g(x) = 47x^2 + 62x + 45\\
    h(x) = 18x^5 + 21x^3 + 61x - 9
\end{gather*}
```

$$f(x) = 59x^{11} + 86x^{10} + 22x^9 + 83x^8 + 74x^7 + 49x^6 + 12x^5 + 83x^4 + 42x^3 + 69x^2 + 47x + 93.$$

y

$$g(x) = 47x^2 + 62x + 45$$
$$h(x) = 18x^5 + 21x^3 + 61x - 9$$

Modo entre-línea, display y *smash*. A mucha gente le molesta el hecho de que LaTeX achica ciertos símbolos para mantener constancia en sus líneas, en cosas como $\sum_{k=1}^{n} k$ y otros operadores grandes. Ya vimos el problema con `\dfrac`, pero no vimos el caso general para `\lim` y el resto de operadores grandes. Para esto LaTeX posee los modificadores `\displaystyle` y `\textstyle` que fuerzan el modo display y entre-líneas respectivamente en cualquier entorno. Personalmente no recomiendo esto por mal uso de líneas, e.g. `$\displaystyle \sum_{k=1}^n k$` $\displaystyle \sum_{k=1}^{n} k$ provoca malos espacios; no obstante hay casos donde se podría hacer sin afectar el espacio, para ello está el comando `\smash`:

```
Línea corta. \\
Así que ahora podemos usar \smash{$\displaystyle\sum_{k=1}^n k
    $}. \\
Línea corta.
```

Línea corta.
Así que ahora podemos usar $\displaystyle \sum_{k=1}^{n} k$.
Línea corta.

Delimitadores grandes. Otro problema de tamaño es que por defecto los delimitadores no se adaptan al tamaño de símbolos grandes en LaTeX,

e.g., $(\frac{1}{2})$ que se ve mal. Para ello se recomienda usar `\left` y `\right`, e.g. `$\left(\dfrac{1}{2}\right)$` $\left(\frac{1}{2}\right)$. Estos se adaptan a cualquier símbolo, sin importar que tan grande, sólo se debe ser cuidadoso, **no se puede** usar `\left` y luego no ocupar un `\right` en la misma línea. En caso de que sólo se quiera usar un delimitador se debe poner un '`.`' en el otro, e.g:

```
$$ \int_0^2 x^2\,dx = \left.\frac{x^3}{3}\right|_0^2 = \
   frac{8}{3}. $$
```

$$\int_0^2 x^2\,dx = \frac{x^3}{3}\Big|_0^2 = \frac{8}{3}.$$

Para cambiar el tamaño de forma manual puede usar los siguientes:

`$\big($`	$\big($	`$\Big($`	$\Big($
`$\bigg($`	$\bigg($	`$\Bigg($`	$\Bigg($

Matrices. Para escribir matrices, el paquete `amsmath` incluye el entorno `matrix` para el modo matemático que funciona como una tabla:

```
$$ \mathbf{Id} :=
    \left[ \begin{matrix}
        1 & 0 \\
        0 & 1
    \end{matrix} \right] $$
```

$$\mathbf{Id} := \left[\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \right]$$

Por defecto, el entorno no posee delimitadores, pero existen variaciones de él que sí:

`matrix`	$\begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix}$	`pmatrix`	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
`bmatrix`	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	`Bmatrix`	$\begin{Bmatrix} 1 & 0 \\ 0 & 1 \end{Bmatrix}$
`vmatrix`	$\begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}$	`Vmatrix`	$\begin{Vmatrix} 1 & 0 \\ 0 & 1 \end{Vmatrix}$

Teoremas. Los teoremas en LaTeX se importan con el paquete `amsthm`, funcionan como entornos que se numeran automáticamente y poseen otras cualidades, se definen en la cabecera así:

```
\newtheorem{thm}{Teorema}
-------------------------------------------------
\begin{thm}
    La suma en $\mathbb{R}$ conmuta.
\end{thm}
```

Teorema 1. *La suma en $\mathbb{R}$ conmuta.*

En este caso el teorema se enumeró sólo, podemos desactivar la numeración con `\newtheorem*`. También el teorema partió desde el 1, pero podría haber iniciado "3.1" donde el 3 es el número de sección, o podría usar el número del capítulo en un libro, para ello le agregamos al final un campo opcional diciendo `section` o `chapter` lo que indica que la numeración se da en dicho formato. Algo interesante es que varios teoremas pueden compartir la misma numeración:

```
\newtheorem{mydef}{Definición}[section]
\newtheorem{cor}[mydef]{Corolario}
\newtheorem*{note}{Observación}
-------------------------------------------------
\begin{mydef}[Hipotenusa y catetos]
    En un triángulo rectángulo, el lado opuesto al ángulo
        recto se dice \textit{hipotenusa} y el resto
       catetos.
\end{mydef}
\begin{note}
    Creo que recuerdo un teorema acerca de eso...
\end{note}
\begin{cor}
    La hipotenusa es mayor que los catetos.
\end{cor}
```

Definición 3.2.1 (Hipotenusa y catetos)**.** *En un triángulo rectángulo, el lado opuesto al ángulo recto se dice hipotenusa y el resto catetos.*

Observación. *Creo que recuerdo un teorema acerca de eso...*

Corolario 3.2.2. *La hipotenusa es mayor que los catetos.*

¡Precaución! El paquete ya introduce un entorno para definiciones, así que no puede definir otro teorema también de tag `def`, por eso se usa `mydef`.

Por último cabe mencionar que los teoremas también permiten el uso de referencias cruzadas.

3.3. Diagramas (básico)

¿Por qué? Habiendo programas como GeoGebra, Photoshop y muchos más para hacer diagramas de forma fácil y gráfica, ¿por qué hacer diagramas en LaTeX si es código? La razón es que LaTeX fue hecho para la publicación de textos y para ello ocupa figuras vectoriales, esto genera el efecto de acercarse a los caracteres y no tener perdida de calidad. Las imágenes en LaTeX comparten esa misma cualidad y en ciertos casos, su facilidad de uso es comparable a la de un editor gráfico, y si no, hay editores que permiten exportar sus diagramas a LaTeX (GeoGebra exporta a Ti*k*Z). Sin contar el que reduce el espacio utilizado en el archivo final y que puede resultar satisfactorio poder hacer figuras sin salir de LaTeX.

Diagramas conmutativos. Creo que la manera más fácil de hacerlos es mediante una librería de Ti*k*Z, que es un muy potente y recomendado paquete para el dibujo de diagramas nativos en LaTeX. Así que añadimos:

```
\usepackage{tikz}
```

En ciertos casos usar `babel` y Ti*k*Z puede generar errores, para resolverlos basta añadir la librería homónima y para hacer diagramas conmutativos usaremos la librería `cd` así que en una sóla línea es:

```
\usetikzlibrary{babel, cd}
```

Éste provee el entorno `tikzcd` que admite argumentos opcionales. En él, los símbolos se muestran automáticamente en modo matemático y se ordenan en forma de tabla con celdas (separados por `&`). Para dibujar flechas se usa el comando `\arrow` o `\ar` para acortar. Toda la información va en un campo opcionar, comenzando por la dirección de la flecha (`l` izquierda, `r` derecha, `u` arriba, `d` abajo), luego una coma y entre comillas el contenido de la flecha.

Entre las comas, puedes poner el color de la flecha si así se quiere. Si las comillas terminan en un apostrofe entonces el contenido saldrá al otro lado de lo esperado y si se utiliza la opción `description` entonces aparece en el medio.

```
\begin{tikzcd}
    A \ar[r, "f"] \ar[rd, "f\circ g"'] & B \ar[d, "g"]\\
```

```
                                                & C
\end{tikzcd}
```

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ & \searrow^{f\circ g} & \downarrow{\scriptstyle g} \\ & & C \end{array}$$

Además se pueden abreviar los movimientos anteponiendo el movimiento en el nombre del comando, e.g., `\rar` reemplaza a `\ar[r]`. En caso de los movimientos diagonales, la componente vertical va primero, e.g., `\rdar` no se admite, pero `\drar` sí. No obstante movimientos más complicados deben ser declarados: no existe `\rrar`, ni `\ddlar` por ejemplo.

Para cambiar la separación de columnas y filas están las opciones de `col sep` y `row sep` respectivamente que pueden igualar a uno de los siguientes tamaños (de menor a mayor):

`tiny, small, scriptsize, normal, large, huge.`

Y los tipos de flechas son:

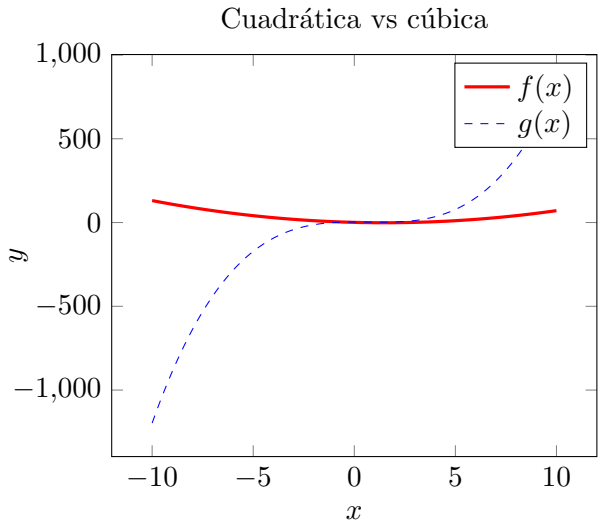
`to head`	$\longrightarrow$	`Rightarrow`	$\Longrightarrow$
`maps to`	$\longmapsto$	`Mapsto`	$\Longmapsto$
`hook`	$\hookrightarrow$	`hook'`	$\rightarrowtail$
`tail`	$\rightarrowtail$	`two heads`	$\twoheadrightarrow$
`dashed`	$\dashrightarrow$	`squiggly`	$\rightsquigarrow$
`harpoon`	$\rightharpoonup$	`harpoon'`	$\rightharpoondown$

Un programa muy útil para hacer diagramas conmutativos y que genera el código para copiar y pegar es `https://tikzcd.yichuanshen.de/`.

Gráficos de funciones. Para esto se requiere el paquete PGFPLOTS, el cual está lleno de opciones y posibles configuraciones, pero a la hora de ser aplicado es simple y elegante:

```
\begin{tikzpicture}
\begin{axis}[
        width   = 8cm,
        title   = {Cuadrática vs cúbica},
        xlabel  = {$x$},
        ylabel  = {$y$},
        samples = 100]
    \addplot[red, very thick, domain=-10:10]{x^2 - 3*x +
       1};
```

```
    \addlegendentry{$f(x)$}
    \addplot[blue, dashed, domain=-10:10]{x^3 - 2*x^2 +
        3};
    \addlegendentry{$g(x)$}
\end{axis}
\end{tikzpicture}
```



Aquí, `width` determina el tamaño del gráfico (sin contar aquel espacio extra que ocupen los números, las etiquetas de los ejes y el título del gráfico). `samples` determina cuantos puntos de ejemplo tomar (más es mejor, pero puede retrasar la compilación, 100 es recomendable). El comando `\addplot` añade un gráfico y en las opciones vimos que el dominio se determina por `domain=`*`i:f`*. Además añadimos entradas por `\addlegendentry`.

Para cambiar el grosor de cada gráfico se ocupan las siguientes opciones: `very thin`, `thin`, `thick` y `very thick`. Además se puede usar la opción `dashed` para que sea discontinua, `dotted` para que sea punteada.

Algo que hay que notar inmediatamente es que hay opciones para el entorno y para cada gráfico individualmente, las del entorno les diremos *globales*, mientras que las de cada gráfico son *locales*. Opciones globales son las etiquetas, título y otras: por ejemplo, podemos ver que el gráfico es más largo que el dominio y rango de las funciones, esto se hace mediante la opción global `enlargelimits`, si no se quiere déjela en 0. Podemos configurar el tamaño en el eje x y y de forma manual con `xmin, xmax, ymin, ymax`. Los números en los ejes se dicen *ticks* y también son configurables, los numerados

son los *ticks mayores* y los otros los *menores*; se pueden configurar con la opción global `minor tick x/y num` y `x/ytick distance`. Puedes agregar también el grid con `grid=minor/major/both/none`, y cambiar las líneas de los ejes con `axis lines=box/left/middle/center/right/none`.

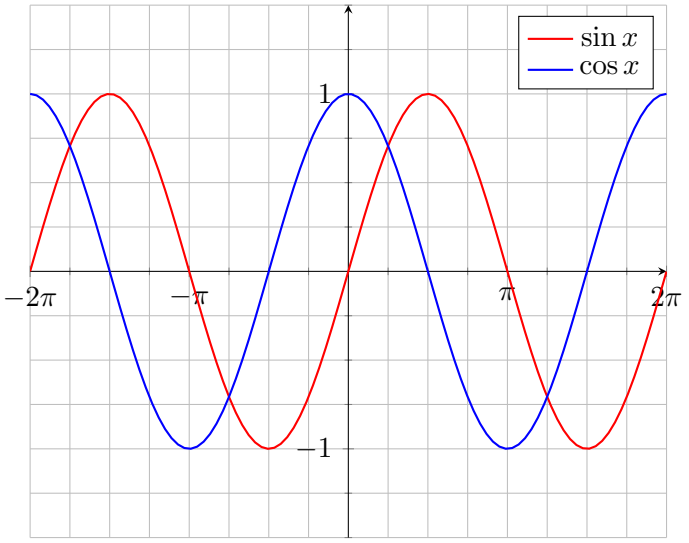
```
\newcommand{\PI}{3.141592}
\begin{axis}[
        width = 10cm,
        title = {Funciones trigonométricas},
        enlargelimits = 0,
        ymin = -1.5, ymax = 1.5,
        axis lines = middle,
        grid = both,
        xtick distance = {\PI},
        ytick distance = 1,
        minor x tick num = 3,
        minor y tick num = 3,
        domain = -2*\PI:2*\PI,
        xticklabels = {
            \empty,
            $-2\pi$,
            $-\pi$,
            \empty,
            $\pi$,
            $2\pi$},
        samples = 100
    ]
    \addplot[red,  thick]{sin(deg(x))};
    \addlegendentry{$\sin x$}
    \addplot[blue, thick]{cos(deg(x))};
    \addlegendentry{$\cos x$}
\end{axis}
```

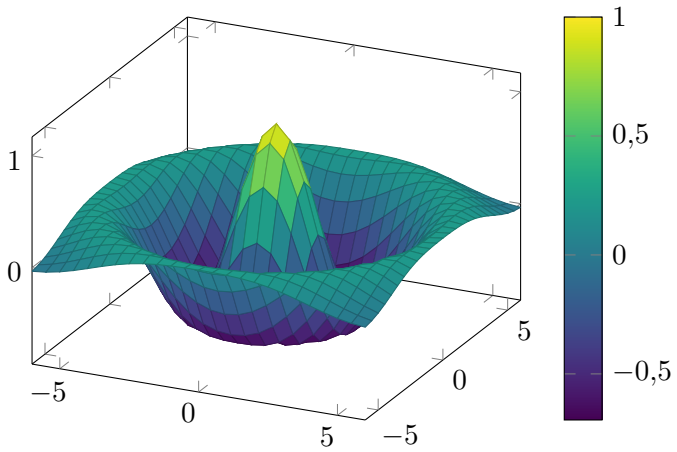
Otra razón para usar PGFPLOTS es que admite gráficos en 3d y funciona de manera muy similar:

```
\begin{axis}[
        width=8cm, samples=25,
        domain=-6:6, domain y=-6:6,
        colorbar, colormap/viridis
    ]
    \addplot3[surf]{exp(-(x^2 + y^2)/25) * cos(deg(sqrt(x
       ^2 + y^2)))};
\end{axis}
```

Aquí, `surf` es un tipo de gráfico, otro tipo es `mesh` que sólo dibuja las líneas. `colormap/viridis` determina el mapa de colores del gráfico (también

Funciones trigonométricas

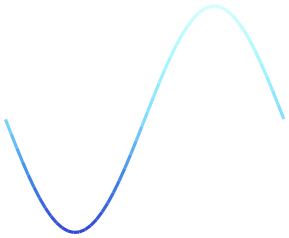




hay `hot`, `cool`, `violet`, `jet` y otros). Puedes definir tu propio mapa de colores con ésta línea en la cabecera:

```
\pgfplotsset{
    colormap={ice}{HTML(0)=(3447d4); HTML(1)=(4188ea);
       HTML(2)=(80dcff); HTML(3)=(a4f8ff); HTML(4)=(d2
       feff)}
}
-------------------------------------------------------------
```

```
\begin{axis}[
        width = 6cm, hide axis, domain=-3.14:3.14,
        colormap/ice, samples = 50
    ]
    \addplot[mesh, very thick]{sin(deg(x))};
\end{axis}
```



Los colores del mapa anterior vienen de `https://www.color-hex.com/color-palette/96010`, así que corrobora que concuerdan.

En gráficos 3d también puedes usar `view={theta}{phi}` donde *theta* y *phi* representan ángulos para la perspectiva del gráfico. Puede encontrar más ejemplos con más opciones aquí: `http://pgfplots.net/tikz/examples/all/`.

3.4. Consideraciones finales

Se recomienda chequear los manuales de Ti*k*Z y PGFPLOTS para buscar más información sobre como usarlos de manera avanzada, estoy consciente de que son manuales muy largos, pero la verdad es que entenderás bien la dinámica básica en el primer o el segundo capítulo donde todo está bien ilustrado y documentado. El resto de páginas es para ver cosas más avanzadas. Además existe un papel introductorio a Ti*k*Z muy completo: `http://cremeronline.com/LaTeX/minimaltikz.pdf`.

Además, como puede presumir, hay un montón de paquetes y contenido que no se trató en este trabajo, y lo más probable es que si quiere una herramienta específica para su área de investigación, entonces existe un paquete que simplifica su implementación, por ejemplo sé que hay librerías para facilitar el diseño de circuitos eléctricos y diagramas de Feynman.

Las partes que poseen código en el documento fueron hechas con el paquete de `listings`, el resto ocupa puras cosas que este documento ya enseña, revisa aquí el código fuente para practicar y también haz el intento de rees-

cribir páginas de libros técnicos en LaTeX, pues es sin lugar a dudas una de las mejores herramientas para los matemáticos y físicos hoy en día.

Capítulo 4

Identificación de errores

Como LaTeX debe ser compilado está expuesto a errores de escritura. El objetivo de los errores es advertir al usuario de mala sintaxis, no debe reaccionarse con miedo sino que debe leerse con detenimiento; usualmente un error indica también la línea que causa el error.

1. `comando before 0:`
   Significa que no ocupaste el comando `\documentclass` u ocupaste otro comando ántes.

2. `File 'type.cls' not found:`
   Significa que intentaste escribir `\documentclass{type}` con un formato inexistente o probablemente te equivocaste al escribir.

3. `File 'paquete.sty' not found:`
   Significa que trataste de importar dicho paquete que no existe.

4. `Option clash for package paquete:`
   Significa que usaste una opción inexistente para dicho paquete.

5. `Missing $ inserted:`
   Significa que intentaste escribir una ecuación matemática sin usar $ o $$.

6. `Undefined control sequence:`
   Significa que usaste un comando que no existe, o probablemente que escribiste mal. Otra posibilidad común es que usaste un comando de un paquete sin importarlo primero como escribir matemáticas sin `amsmath` o importar una foto sin `graphicx`.

7. `File ended while scanning use of` *`comando`*:
   Significa que abriste un comando como `\textit{...` sin cerrarlo con la llave `}` faltante.

8. `Misplaced alignment tab character &:`
   Significa que usaste & fuera de un entorno apropiado, como una tabla fuera del entorno `tabular` o `align`.

9. `File '`*`foto.jpg`*`' not found: using draft setting:`
   Significa que intentaste incluir un archivo de una foto que no existe.