

Apuntes de Latex

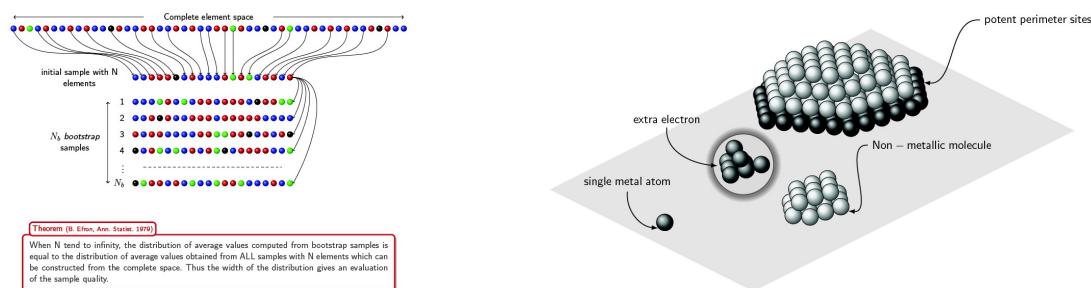
Capítulo 1

SECCIÓN 1

Qué es LATEX? Funcionamiento básico

LATEX es un sistema avanzado de composición de textos, conocido popularmente por su amplio potencial para el proceso de textos científicos. Pero sus capacidades van mucho más allá de la escritura de fórmulas matemáticas (para lo cual funciona excelentemente –ver capítulo 3–). LATEX posee amplias capacidades a la hora de configurar la apariencia general del documento (márgenes, cabeceras, división en secciones...). Asimismo, ofrece multitud de herramientas para la elaboración avanzada de índices, referencias cruzadas y bibliografía. El usuario tiene también a su disposición multitud de herramientas de maquetación para componer tanto párrafos de texto como gráficos, de forma absolutamente libre.

Ejemplos de uso de recursos LATEX



Recursos gráficos con PGF

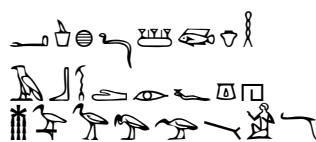
$$\int_D |\bar{\partial} u|^2 \Phi_0(z) e^{\alpha|z|^2} \quad (1)$$

$$\lim_{h \rightarrow +\infty} \int_{\Omega} |\nabla u_h| dx = |\mathbf{D}u|(\Omega) \quad (2)$$

$$P_{r-j} = \begin{cases} 0 & r-j \text{ impar,} \\ r!(-1)^{(r-j)/2} & r-j \text{ par.} \end{cases} \quad (3)$$

W er reitet so spät durch Wind ?
Es ist der Vater mit Kind

Er hat den Knaben in Arm



Fórmulas

Efectos de texto



Partituras musicales con MusixTeX

Amplias capacidades gráficas están disponibles, mediante el empleo de paquetes adicionales (*pstricks*, *pgf*, *tikz*). Además de documentos impresos, existen utilidades como *beamer* para la creación de presentaciones de alta calidad. Finalmente, el carácter de código abierto del sistema y su forma modular hace posible el utilizar multitud de recursos programados por la comunidad de usuarios de L^AT_EX para las aplicaciones más diversas, desde escritura de partituras musicales a diagramas de circuitos electrónicos.

El sistema T_EX/L^AT_EX , a diferencia de procesadores de texto como MS-Word, no posee una interfaz gráfica interactiva en la cual según se compone el texto se observa directamente el resultado (lo que se conoce como editor tipo "WYSIWYG" ó WHAT-YOU-SEE-IS-WHAT-YOU-GET). En su lugar, T_EX/L^AT_EX trabaja de forma similar a un lenguaje de programación¹, compilando un fichero fuente (con la extensión .tex) del cual se obtiene como resultado.un fichero procesado que podremos visualizar de diversas formas.

Los ficheros fuente .tex son simples archivos de texto ascii que pueden ser editados con cualquier editor de textos (aunque se sugiere trabajar desde entornos integrados como WinEdt, Texniccenter ó Kile², ésto no es imprescindible) los cuales contienen tanto el texto en sí que queremos procesar, como comandos L^AT_EX que se ocupan de formatear el texto. Todos estos comandos tienen en general (hay excepciones) la sintaxis:

\NombreComando [opciones] {argumento}

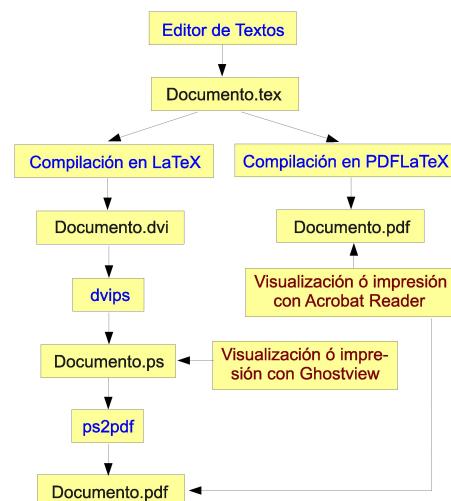
donde NombreComando es el nombre del comando en cuestión, argumento representa texto o variables L^AT_EX sobre las que actúa el comando, y opciones denotan en general variables optativas que podemos ajustar. Es muy importante tener en cuenta que el nombre del comando es sensible a mayúsculas y minúsculas, por lo que debe escribirse siempre tal y como lo encontramos en la documentación L^AT_EX. Además, no se permiten espacios entre el nombre del comando y su argumento (una fuente común de errores de sintaxis). El carácter \ tiene siempre la misión de señalar al compilador el comienzo de una instrucción.

Es esencial conocer que, a la hora de compilar un documento L^AT_EX, existen dos posibilidades (ver gráfico adjunto):

- Compilar con el programa tradicional L^AT_EX, lo cual da como resultado la creación de un fichero intermedio Documento.dvi, que contiene toda la información de formateado del documento original. Posteriormente, podemos transformar con el programa dvips nuestro archivo .dvi en un documento Postscript, el cual puede visualizarse con Ghostview ó imprimirse en una impresora postscript. Finalmente, la utilidad ps2pdf permite traducir documentos postscript al formato acrobat PDF.

¹De hecho, T_EX ES un lenguaje de programación; a lo largo del curso se explicarán algunos fundamentos de programación en T_EX

²www.winedt.com, www.texniccenter.org, kile.sourceforge.net



Funcionamiento de L^AT_EX

- Compilar con el relativamente nuevo programa PDFLATEX, que permite obtener directamente como resultado de la compilación el documento en formato **PDF**.

Aunque ambas posibilidades pueden producir en numerosos casos el mismo resultado final, es crucial mencionar que **no son equivalentes**; el carácter más moderno del compilador PDFLATEX implica que algunos recursos LATEX más antiguos pueden no estar disponibles, ó dar resultados erróneos. Por ejemplo, el conjunto de utilidades gráficas PSTricks, que hace un uso intensivo del lenguaje postscript, no soporta el uso de PDFLATEX por lo que cualquier documento que contenga tales recursos ha de compilarse necesariamente con la secuencia LATEX + dvips + ps2pdf. Por contra, el paquete **beamer** para presentaciones está exclusivamente diseñado para trabajar en PDFLATEX. *Se indicará, según vayan surgiendo estos casos especiales, cuál de las dos rutas debe emplearse.*³

Ejemplo de documento fuente LATEX y su de resultado compilado

```
\documentclass[a4,11pt]{article}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[spanish]{babel}
\renewcommand{\shorthandsspanish}{}

\title{Documento Fuente \LaTeX{}}
\author{Perico de los Palotes}
\date{}

\begin{document}

\maketitle
\tableofcontents

Ejemplo de documento \LaTeX{} de la clase \ttfamily{article} con una estructura reducida. Esta incluye secciones, subsecciones y una referencia cruzada.

\section{Primera sección}\label{primera}
Una primera sección con una fórmula y una lista.

\subsection{Fórmula}
Una ecuación: \(\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y}\)

\subsection{Listas}
Una lista de ítems señalados con una marca:

\begin{itemize}\itemsep=0pt
\item Primer ítem
\item Segundo ítem
\item Tercer ítem
\end{itemize}

\section{Segunda sección}
Esta sección complementa a la sección 1 incluyendo ejemplos de tablas escritas en \LaTeX{}.

\end{document}
```

Documento Fuente LATEX

Perico de los Palotes

Índice

1. Primera sección	1
1.1. Fórmula	1
1.2. Listas	1
2. Segunda sección	1

Ejemplo de documento LATEX de la clase `article` con una estructura reducida. Esta incluye secciones, subsecciones y una referencia cruzada.

1. Primera sección

Una primera sección con una fórmula y una lista.

1.1. Fórmula

Una ecuación: $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y}$

1.2. Listas

Una lista de ítems señalados con una marca:

- Primer ítem
- Segundo ítem
- Tercer ítem

2. Segunda sección

Esta sección complementa a la sección 1 incluyendo ejemplos de tablas escritas en LATEX.

1

³Los detalles de cómo utilizar el entorno integrado WinEdt para compilar documentos se encuentran explicados en el archivo “WinEdt Minitutorial”

SECCIÓN 2

Conceptos básicos de formateado de texto

Todo documento L^AT_EX posee la siguiente estructura:

- **Preámbulo:** Declaraciones de carácter GLOBAL que afectan a la totalidad del documento
 - `\documentclass[opciones]{tipo_de_documento}` → **OBLIGATORIA**; éste debe de ser además el primer comando del documento. Mediante ésta declaración indicamos a L^AT_EX que tipo de documento (book, article, report, letter...) queremos escribir
 - `\usepackage[opciones]{paquete}` → carga de paquetes con utilidades (para incluir gráficos, texto en color, presentaciones, etc...)
 - Otras declaraciones: Interlínea, formato de página, fuentes, etc...
- **Cuerpo:** Todo lo comprendido entre `\begin{document}` y `\end{document}`, es decir, el documento propiamente dicho.

En la página anterior puede verse un ejemplo de documento simple, tomando la forma de la clase `article`.

2.1. Reglas generales de composición de texto

Existen ciertas reglas generales a la hora de escribir texto:

- **Texto alineado y centrado:** El texto se alinea y justifica automáticamente, según medidas predeterminadas o impuestas por nosotros; **IMPORTANTE:** Toda medida predeterminada es ajustable
- **Los espacios se ignoran:** Da igual separar las palabras con 1 espacio o varios. Para aumentar el espacio de separación entre palabras se usa: \ seguido de espacio
- **Punto y aparte:** Dejar una línea en blanco equivale a cambiar de párrafo (punto y aparte). Lo mismo se obtiene con `\par`. **Ojo!** Nótese que dejar *varias* líneas en blanco es igual a dejar una: el efecto es el mismo, i.e., comienzo de un nuevo párrafo.
- **Cambio de línea:** Se puede cambiar de línea, *sin cambiar de párrafo*, usando \\

Los ejemplos en la página siguiente ilustran éstos puntos. Es importante tener en cuenta que L^AT_EX, por defecto, coloca una pequeña indentación al comienzo de cada párrafo. Para controlar ésta indentación se dispone del comando `\parindent=Xmm` (más adelante se explica el manejo de unidades de longitud). Éste comando cambia la indentación de párrafo de forma *global*, afectando a todos los párrafos tras el comando. Si se desea suprimir la indentación para un párrafo en particular, puede hacerse colocando `\noindent` al comienzo del mismo. El espacioado estándar entre párrafos (nulo por defecto) se puede modificar a través del comando `\parskip=Xmm`

Manejo de espacios, cambios de línea y de párrafo

```

En un lugar de la mancha
de cuyo nombre no quiero
acordarme, \ \\ no ha mucho tiempo
que \\ vivía un hidalgo de los de
lanza en astillero, adarga antigua,
rocín flaco y galgo corredor. \par Una
olla de algo más vaca que carnero,
salpicón las más noches, duelos y
quebrantos los sábados, lentejas
los viernes, algún
palomino de añadidura los domingos,
consumían las tres partes de su hacienda.

```

```

En un lugar de la mancha de cuyo nombre
no quiero acordarme, no ha mucho tiempo
que
vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
Una olla de algo más vaca que carnero, sal-
picón las más noches, duelos y quebrantos los
sábados, lentejas los viernes, algún
palomino de añadidura los domingos, consum-
ían las tres partes de su hacienda.

```

Control de la indentación y del espaciado entre párrafos

```
\parindent=8mm
En un lugar de la mancha de cuyo nombre no
quiero acordarme, no ha mucho tiempo que
vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
```

```
Una olla de algo más vaca que carnero,
salpicón las más noches, duelos y quebrantos
los sábados, lentejas los viernes, algún
palomino de añadidura los domingos,
consumían las tres partes de su hacienda.
```

```
\noindent
En un lugar de la mancha de cuyo nombre no
quiero acordarme, no ha mucho tiempo que
vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
```

```
\parskip=3mm
Una olla de algo más vaca que carnero,
salpicón las más noches, duelos y quebrantos
los sábados, lentejas los viernes, algún
palomino de añadidura los domingos,
consumían las tres partes de su hacienda.
```

```
En un lugar de la mancha de cuyo nombre no
quiero acordarme, no ha mucho tiempo que
vivía un hidalgo de los de lanza en astillero,
adarga antigua, rocín flaco y galgo corredor.
```

```
En un lugar de la mancha de cu-
yo nombre no quiero acordarme, no ha
mucho tiempo que vivía un hidalgo de
los de lanza en astillero, adarga anti-
gua, rocín flaco y galgo corredor.
```

```
Una olla de algo más vaca que car-
nero, salpicón las más noches, duelos
y quebrantos los sábados, lentejas los
viernes, algún palomino de añadidura
los domingos, consumían las tres par-
tes de su hacienda.
```

```
En un lugar de la mancha de cuyo nom-
bre no quiero acordarme, no ha mucho
tiempo que vivía un hidalgo de los de
lanza en astillero, adarga antigua, rocín
flaco y galgo corredor.
```

```
Una olla de algo más vaca que car-
nero, salpicón las más noches, duelos
y quebrantos los sábados, lentejas los
viernes, algún palomino de añadidura
los domingos, consumían las tres par-
tes de su hacienda.
```

```
En un lugar de la mancha de cu-
yo nombre no quiero acordarme, no ha
mucho tiempo que vivía un hidalgo de
los de lanza en astillero, adarga anti-
gua, rocín flaco y galgo corredor.
```

SECCIÓN 3

Espacios horizontales y verticales

Un elemento muy importante a la hora de construir un documento son las longitudes que se utilizan para delimitar diferentes distancias entre objetos. Existe una amplia variedad de **unidades de longitud** que podemos utilizar:

Unidades de longitud:

Medidas absolutas		
pt	punto	$1\text{pt} \approx 0.35146\text{ mm}$
pc	pica	$1\text{pc} = 12\text{ pt}$
in	pulgada	$1\text{in} = 72.27\text{ pt} = 2.54\text{ cm}$
cm	centímetro	
mm	milímetro	$1\text{ mm} = 2.845\text{ pt}$
dd	didot	$1157\text{ dd} = 1238\text{ pt}$
cc	cícero	$1\text{ cc} = 12\text{ dd}$
Medidas relativas		
em	aprox. la anchura de una ‘M’ de la fuente en curso	
ex	aprox. la anchura de una ‘x’ de la fuente en curso	

Las medidas relativas “em” y “ex” dependen del tamaño de letra en curso que estemos utilizando. Existen muchos otros ejemplos de uso de medidas relativas; en general, LATEX maneja internamente una amplia cantidad de longitudes que en principio desconocemos o que incluso son *elásticas*. Es un procedimiento frecuente, y bastante útil, el manejar longitudes relativas. Por ejemplo, si queremos delimitar la anchura de un objeto cualquiera como la mitad de la anchura del texto en la página, definida a través de la variable `\textwidth`, podemos utilizar la longitud relativa `0.5\textwidth`.

Para modificar el valor de medidas de longitud (como `\parskip`, `\parindent`, etc...) existen dos sintaxis de comando equivalentes:

`\Longitud=Xmm` `\Longitud Xmm`

Hay varias formas de añadir espacios verticales entre diversos objetos:

- `\backslash` y `\newline` —> Análogos: salto de línea simple. En este punto es bueno recordar la diferencia con `\par`: éste último *cambia de párrafo*, mientras que los anteriores simplemente terminan bruscamente un línea y pasan a la siguiente, sin comenzar nuevo párrafo.
- `\backslash[Salto]` —> Espacio vertical de longitud *Salto*
- `\vspace{Salto}` —> Análogo al anterior; nótese que no tiene efecto *al comienzo de una página*, en tal circunstancia se puede usar el comando análogo `\vspace*{Salto}`, que evita este problema
- Espacios verticales de longitud predefinida:

- `\bigskip` —> approx. 1 línea en blanco
- `\medskip` —> approx. 1/2 del espacio de una línea en blanco
- `\smallskip` —> approx. 1/4 del espacio de una línea en blanco

Para añadir espacios horizontales podemos utilizar:

- `\hspace{longitud}` —> Espacio horizontal de extensión *longitud*
- `\hspace*{longitud}` —> Igual, válido al comienzo de una línea
- Predefinidos:
 - `\,` —> un espacio entre palabras
 - `\enskip` —> medio “em”
 - `\quad` —> un “em”
 - `\quad\quad` —> dos “em”

Si queremos saltar de página, podemos utilizar tanto `\newpage` como `\clearpage`, que indican a L^AT_EX que se debe finalizar la página en curso y comenzar una nueva. Nótese que `\clearpage` posee un significado especial: en el caso de que haya *elementos flotantes* (como figuras y tablas, para los cuales L^AT_EX se encarga por si solo de buscar la ubicación más adecuada) pendientes de ubicar, tales elementos se imprimirán inmediatamente en la página siguiente, formada por sólo tablas y gráficas. El texto se reiniciará entonces otra página después.

La distancia entre líneas (interlínea) también es modificable, situando en el preámbulo (**OJO!**, y sólo en el preámbulo, fuera de ahí la instrucción no funciona) la instrucción:

`\renewcommand*{\baselinestretch}{Número}`

lo que **escala** la interlínea por la cantidad Número (un valor de 2.0 equivaldría a doble espaciado). Si se quiere cambiar el valor de la interlínea en distintas partes del documento, se puede utilizar el paquete `setspace`, con la sintaxis:

`\usepackage[espaciado]{setspace}`

lo cual hace el valor de la interlínea en todo el documento igual a **espaciado**. Los posibles valores son: `singlespacing`, `onehalfspacing` y `doublespacing`, que equivalen respectivamente a un valor de baselinestretch de 1, 1.5 y 2 (si no se incluye la opción `espaciado`, el valor por defecto es `singlespacing`). Después, en el cuerpo del documento, puede variarse a voluntad la interlínea mediante los comandos: `\singlspacing`, `\onehalfspacing` y `\doublespacing`.

SECCIÓN 4

Centrado y justificación a los márgenes del texto

Un texto dado puede ajustarse a cualquiera de los dos lados de la página ó al centro:

- Para centrar un párrafo se utiliza el *entorno* `center`

```
\begin{center}
Texto a centrar
\end{center}
```

Ejemplo:

<pre>\begin{center} El ingenioso hidalgo\\ D. Quijote de la Mancha\\[0.3cm] Miguel de Cervantes Saavedra \end{center}</pre>	El ingenioso hidalgo D. Quijote de la Mancha Miguel de Cervantes Saavedra
---	---

- Para alinear a los lados, tenemos los entornos **flushleft** y **flushright**

```
\begin{flushleft}
Probando \\
la forma de alinear \\
por la izquierda
\end{flushleft}
```

```
\begin{flushright}
Probando \\
la forma de alinear \\
por la derecha
\end{flushright}
```

Probando
la forma de alinear
por la *izquierda*

Probando
la forma de alinear
por la *derecha*

- Para textos pequeños, menores que una línea, se pueden utilizar, respectivamente:

`\leftline{Texto}`

`\centerline{Texto}`

`\rightline{Texto}`

Es interesante el hecho de que existen en L^AT_EX multitud de comandos que poseen ambas versiones, una “corta”, de tipo:

`\comando{argumento}`

donde el comando afecta a un texto pequeño (argumento), y otra “larga”, denominada **entorno** de tipo:

`\begin{entorno} Texto largo \end{entorno}`

Alternativamente, también existe otra método para éste último procedimiento, incluyendo el comando dentro de un **grupo**:

`{\comando Objeto extenso}`

donde los *delimitadores* { y } definen el grupo de objetos a los que afectará el comando.

SECCIÓN 5

Silabeo

Puede ocurrir que los algoritmos de silabeo de L^AT_EX no funcionen correctamente y que al cambiar de línea se rompa una palabra de forma inadecuada. Para evitarlo hay dos alternativas:

- Utilizar la instrucción `\hyphenation{lista de palabras}` en el preámbulo; por ejemplo, `\hyphenation{For-tran fi-che-ro}` sólo permitirá la división de las palabras “fortran” y “fichero” por los lugares indicados. Nótese que no se permiten caracteres con acentos ó símbolos en el argumento, y que no se hacen distinciones entre las letras mayúsculas y minúsculas de las palabras en la orden.
- Fuera del preámbulo, se puede utilizar la instrucción: `\-` Ésta es válido utilizarla en palabras con acentos ó símbolos, por ejemplo: `te\l\'e\f\o\l\no`

SECCIÓN 6

Escribiendo en castellano

El idioma por defecto de L^AT_EX es el inglés. Esto quiero decir que, de forma estándar, no se reconocen los caracteres especiales como Ñ, letras acentuadas, etc... Además, definiciones por defecto, como títulos para capítulos, fechas, etc, estarán en inglés. Pruébese por ejemplo lo siguiente:

```
\documentclass{article}
\begin{document}
Saludos desde \LaTeX. Haciendo una compilación de prueba, de texto
en español, para ver si todo funciona.

```

```
Escrito y compilado el día \today.
\end{document}
```

y se verá que los acentos y ñ desaparecen y que la fecha (comando `\today`) se imprime en inglés. Para solucionar tales problemas se pueden incluir los siguientes paquetes en el preámbulo del documento:

- `\usepackage[latin1]{inputenc}` — Para que L^AT_EX entienda los símbolos del teclado español. Con este paquete podemos teclear directamente símbolos del teclado que serán reconocidos por el compilador ⁴
- `\usepackage[T1]{fontenc}` — Para que utilice nuestros tipos acentuados, en vez de construirlos con METAFONT

⁴a excepción del símbolo del euro (€); véase mas adelante

- `\usepackage[spanish]{babel}` → Reglas españolas para división de sílabas, traducción de comandos, etc...
- `\renewcommand{\shorthandsspanish}{}{}` → Desactiva métodos taquigráficos en español (que pueden molestarnos)

Podemos utilizar las declaraciones anteriores como cabecera estándar para escribir textos en español. Llegado este punto, es bueno destacar que existen alternativas para escribir caracteres acentuados sin utilizar el teclado español y la codificación “latin1” en el paquete `inputenc`. Ésto puede ser útil si, por ejemplo, nos encontramos en el extranjero ó carecemos del teclado adecuado. Para ello, desactivaríamos las declaraciones `\usepackage[latin1]{inputenc}` y `\usepackage[T1]{fontenc}`, haciendo uso de las instrucciones expuestas en la siguiente tabla para conseguir caracteres acentuados:

ò	<code>\`o</code>	ó	<code>\^o</code>	ô	<code>\~o</code>	õ	<code>\~o</code>
ó	<code>\=o</code>	ó	<code>\.\o</code>	ö	<code>\\"o</code>	ç	<code>\c c</code>
ó	<code>\u o</code>	ó	<code>\v o</code>	ó	<code>\H o</code>	ó	<code>\d o</code>
ó	<code>\b o</code>	ó	<code>\t oo</code>				

IMPORTANTE: Las letras *i* y *j* necesitan un tratamiento especial, dado que no deben tener sus puntos antes de ser acentuadas. Su eliminación se consigue con los comandos `\i` y `\j`, respectivamente. Así, para obtener, por ejemplo:

Él está aquí

se debería escribir `\'{E}l est\'{a} aqu\'{i}`

Para otros símbolos pertenecientes a diversos idiomas véase la siguiente tabla:

œ	<code>\oe</code>	Œ	<code>\OE</code>	æ	<code>\ae</code>	Æ	<code>\AE</code>
å	<code>\aa</code>	Å	<code>\AA</code>				
ø	<code>\o</code>	Ø	<code>\O</code>	ł	<code>\l</code>	Ł	<code>\L</code>
ß	<code>\ss</code>						
í	<code>!‘</code>	¿	<code>?‘</code>				

SECCIÓN 7

Más sobre signos ortográficos

En esta sección describiremos diversos comandos de utilidad a la hora de escribir un texto (comillas, guiones, ordinales, etc...)

7.1. Los diez caracteres reservados

En LATEX existen 10 caracteres especiales que el sistema utiliza para distintos propósitos, a saber:

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

los cuales tienen los siguientes usos:

\ Indicador de comando.

{ } Delimitadores de grupos.

Nombra los argumentos de un comando.

& Separa columnas de una tabla.

% Se utiliza para introducir comentarios: En una línea del fichero fuente dada, todo lo que se encuentre a la derecha de éste signo es ignorado por el compilador y se entiende como comentario.

~ Se utiliza para evitar la separación de palabras: Es una conocida norma tipográfica el no separar términos complementarios, como por ejemplo Sr. Director ó A. Einstein. Utilizando la tilde como ligadura, se evita que L^AT_EX rompa éstas palabras en dos al cambiar de línea: Sr.~Director, A.~Einstein ⁵

\$ _ ^ Se utilizan en fórmulas matemáticas.

El carácter reservado significa que no podemos incluirlos en el texto normal simplemente escribiéndolos. En la siguiente tabla se muestra la sintaxis que se debe utilizar para escribirlos dentro de un documento:

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

7.2. Comillas, guiones, puntos suspensivos, grados, etc..

Comillas:

Tecleamos: << Texto >> Resulta: « Texto »

Tecleamos: ““ Texto ”” Resulta: ““ Texto ””

Tecleamos: ‘‘ Texto ’’ Resulta: ‘‘ Texto ’’

Guiones:

Tecleamos: - Resulta: -

Tecleamos: -- Resulta: –

⁵Otro método es encerrar la frase dentro de un caja: \mbox{Texto}

Tecleamos: --- Resulta: —

Tecleamos: \$-\$ Resulta: –
(signo matemático menos)

Puntos suspensivos:

Hay varias formas de introducir los puntos suspensivos:

- ... → (la más sencilla)
- \... → (sólo funciona con la opción `spanish` de `babel`)
- \dots → (comando propio de L^AT_EX, siempre disponible)
- \ldots → Distancia entre puntos suspensivos algo mayor

Ordinales y grados:

Para obtener ordinales abreviados (1^a ó 1^o), podemos hacerlo directamente desde el teclado.
Para escribir otros ordinales, puede hacerse con:

`Superíndice`

Por ejemplo, 3er produce 3^{er}. Otra versión (sólo disponible en `babel`, versión `spanish`) es:

`\sptext{Superíndice}`

que introduce un punto antes del superíndice y cambia el tamaño del superíndice si es un carácter en mayúsculas: 2.^a, 3.^{er}. Para generar correctamente el signo de grado, se utiliza: `\textdegree`

Otros signos:

El siguiente cuadro muestra como obtener otros signos diversos:

Comando	Resultado	Comando	Resultado
\dag \S	† §	\ddag \P	‡ ¶
\textbullet	•	\textvisiblespace	ؘ
\textregistered \texttrademark	® ™	\copyright \pounds	© £

Los comandos \copyright y \textregistered son casos particulares de un comando más general: \textcircled{Carácter}, que encierra Carácter dentro de un círculo.

7.3. El euro

Debido a la modernidad del símbolo del euro, la opción `latin1` del paquete `inputenc` desgraciadamente aún no reconoce éste signo. La solución para obtener el símbolo del euro está en cargar en el preámbulo el paquete `eurosym`:

```
\usepackage{eurosym}
```

tras lo cual se obtiene en símbolo € con el comando `\euro`. Puede incluso mejorarse la situación insertando la siguiente declaración en el preámbulo (por supuesto, **después** del comando `\usepackage{eurosym}`):

```
\DeclareInputText{128}{\euro}6
```

que asignaría el símbolo € del teclado la instrucción “`\euro`”, lo que ya permite utilizar el símbolo del teclado con normalidad.

SECCIÓN 8

Párrafos especiales: `quote`, `quotation`, `verse`, y más...

Los entornos `quote` y `quotation` permiten introducir citas textuales, en párrafos ligeramente más pequeños que el texto base; por ejemplo:

La inclusión de citas textuales, como la del escritor Bertolt Brecht que viene a continuación, es una tarea sencilla con `\LaTeX`.

```
\begin{quote}
```

Hay personas que luchan un día, y son buenas. Hay otras que luchan un año y son mejores. Hay quienes luchan muchos años, y son muy buenas.

Pero hay algunas que luchan toda la vida: éas son las imprescindible

```
\end{quote}
```

produce:

Con `quote`:

La inclusión de citas textuales, como la del escritor Bertolt Brecht que viene a continuación, es una tarea sencilla con `\LaTeX`.

Hay personas que luchan un día, y son buenas. Hay otras que luchan un año y son mejores. Hay quienes luchan muchos años, y son muy buenas.

Pero hay algunas que luchan toda la vida: éas son las imprescindible

Con `quotation`:

La inclusión de citas textuales, como la del escritor Bertolt Brecht que viene a continuación, es una tarea sencilla con `\LaTeX`.

Hay personas que luchan un día, y son buenas. Hay otras que luchan un año y son mejores. Hay quienes luchan muchos años, y son muy buenas.

Pero hay algunas que luchan toda la vida: éas son las imprescindible

Como se puede ver, `quotation` introduce sangrado en los párrafos de la cita, y disminuye la separación entre los mismos.

Otro entorno predefinido es el `verse`, para escribir versos. Tiene la siguiente sintaxis:

⁶En Windows. Para Linux, el código del carácter € es 164

```
\begin{verse}
verso1 \\
verso2 \\
.... \\
\end{verse}
```

Utilizando el paquete `shapepar` se pueden construir párrafos con formas muy especiales. Así por ejemplo, empleando: `\heartpar{Texto del párrafo a formatear}` puede obtenerse lo siguiente:

El 30 de marzo de
 1977, el profesor Donald E. Knuth, de
 la Universidad de Stanford, recibió las galeras
 o pruebas de imprenta de la segunda edición del segundo
 volumen de su famosa obra *The Art of Computer Programming*. La impresión que dichas pruebas causaron al autor fue
 nefasta; él mismo las calificó de tipográficamente horribles y
 tan importantes le parecieron los problemas a los que se en-
 frentaba que decidió resolverlos por sí mismo. A partir de
 las ideas de Gutenberg y utilizando las computadoras
 como herramientas, Knuth creó `TEX`, un sistema para
 escribir textos científicos (especialmente matemá-
 ticos), cómodo y transportable entre platafor-
 mas, que muchos consideran ahora como
 la aportación más importante rea-
 lizada en este campo desde
 la imprenta de Gu-
 tenberg.
 ♡

Lo anterior es una aplicación del comando general `\parshape`, que permite construir párrafos de forma arbitraria. Su sintaxis es la siguiente:

`\parshape=n i1 l1 i2 l2... in ln`

Indica que las primeras n líneas del párrafo tendrán longitudes l_1, \dots, l_n , respectivamente, y estarán sangradas i_1, \dots, i_n , respectivamente. Si el párrafo tiene más de n líneas, las condiciones para la n -ésima serán repetidas hasta final de párrafo. Para cancelar el comando, basta incluir `\parshape=0`.

SECCIÓN 9
Tipos de letra

9.1. Familias

Por defecto, L^AT_EX utiliza los tipos Computer Modern Fonts, creados por D.E. Knuth para su utilización en T_EX.⁷ Estos tipos agrupan tres familias diferentes:

- Roman (la opción por defecto)
- Sanserif (sin adornos)
- Typewriter (tipo máquina de escribir)

con las siguientes instrucciones para obtenerlas, respectivamente:

▪ \textrm{Texto}	(roman)	\rmfamily Texto
▪ \textsf{Texto}	(sanserif)	\sffamily Texto
▪ \texttt{Texto}	(typewriter)	\ttfamily Texto

Los comandos a la izquierda corresponden al modo *Texto*, es decir, para textos cortos **no más largos que un párrafo**. Los comandos a la izquierda se mantienen hasta que se declare una nueva familia, aunque lo usual suele ser incluirlos dentro de un *grupo*, en la forma siguiente:

```
Esto es roman, {\sffamily esto es sanserif},
{\ttfamily esto es typewriter}, y esto sigue siendo roman.
```

que produce:

Esto es roman, esto es sanserif, esto es typewriter, y esto sigue siendo roman.

Completamente análogo a {\sffamily Texto} sería utilizar el siguiente entorno:

```
\begin{sffamily} Texto extenso... \end{sffamily}
```

9.2. Perfiles

Para cada familia tenemos cuatro posibles perfiles, recto (opción por defecto, *italico*, *inclinado* (slanted) y *VERSAL* (letras mayúsculas pequeñas), cuyos comandos correspondientes son:

▪ \textup{Texto}	(recto)	\upshape Texto
▪ \textit{Texto}	(<i>italico</i>)	\itshape Texto
▪ \textsl{Texto}	(<i>inclinado</i>)	\slshape Texto
▪ \textsc{Texto}	(<i>VERSAL</i>)	\scshape Texto

⁷Pueden cargarse otros muchos tipos, lo cual se verá mas adelante

9.3. Grosor

Finalmente, hay dos grosores (también llamados series) para cada tipo: el normal ó medio (opción por defecto) y el grueso ó negrita. Se activan con:

- `\textmd{Texto}` (medio) `\mdseries Texto`
- `\textbf{Texto}` (grueso) `\bfseries Texto`

Todas las características explicadas (familia, perfil y grosor) pueden combinarse (aunque puntualmente alguna opción mixta no esté disponible). Por ejemplo, `\bfseries\itshape` produce ***letra negrita itálica***.

9.4. Enfatizar

Se puede resaltar texto con el comando `emph{Texto}` (ó `\em Texto`, en modo extendido) lo cual pone en itálica el texto si el ambiente es normal, ó pone normal el texto si el ambiente es itálico.

9.5. Tamaño

Tenemos a nuestra disposición los siguientes tamaños, los cuales son *relativos* a la fuente estándar del documento (que puede cambiarse, ya se verá más adelante cómo)

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

9.6. Colores

El paquete `color` permite colorear un texto. Se puede cargar como `\usepackage{color}`, lo cual permite sólo usar unos pocos colores básicos (white, black, red, blue, yellow, green)⁸

Para cambiar el color, se usan los comandos:

- `\textcolor{NombreColor}{Texto}`
- `\color{NombreColor}`

siendo la primera la versión corta, y la segunda la versión larga que tendrá efecto hasta que se cambie de nuevo el color (ó hasta que termine el grupo, si se usa:

`{\color{NombreColor} Texto extenso... }`

Todas las características de tipo, tamaño, forma, grosor, color, etc... para texto escrito pueden combinarse mediante la anidación de comandos, como muestra el siguiente ejemplo:

```
{\Large\bfseries\color{blue} Esto es letra grande, negrita y azul}
```

Esto es letra grande, negrita y azul

```
{\small\sffamily\itshape\color{red} Letra pequeña, sanserif, itálica y roja}
```

Letra pequeña, sanserif, itálica y roja

```
{\huge\textcolor{green}{\textbf{\textsc{Letra Mayúscula negrita, verde y muy grande}}}}
```

LETRA MAYÚSCULA NEGRITA, VERDE Y MUY GRANDE

9.7. El paquete soul

Cargando el paquete `soul`, podemos incorporar los siguientes efectos de resaltado de texto:

`\hl{Texto}`

Texto marcado (para ésto hace falta tener también cargado el paquete `color`)

`\ul{Texto}`

En un lugar de la mancha de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

⁸En un capítulo posterior se explicará como manejar cualquier tipo de color, además de la posibilidad de colorear páginas, cajas, etc...

```
\st{Texto}
```

En un lugar de la mancha de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, ~~re~~cín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

```
\so{Texto}
```

Texto resaltado (con letras espaciadas más de lo normal)

Apuntes de Latex

Capítulo 2: Listas y Tablas

SECCIÓN 1

Listas no numeradas: El entorno `itemize`

El entorno `itemize` permite escribir una lista de objetos, siendo adecuado para listas sencillas. Las diversas entradas de la lista se resaltan con un indicador, que por defecto es un círculo negro (`\textbullet`), aunque el uso de la opción `spanish` de `babel` puede cambiar ésta predefinición. Las entradas de la lista también se encuentran indentadas respecto al margen izquierdo, y se añade por defecto un espacio vertical antes y después de la lista. Obsérvese en el siguiente ejemplo la sintaxis del entorno y el uso del comando `\item` para introducir las entradas de la lista:

```
... párrafo anterior  
  
\begin{itemize}  
    \item Esto es un ejemplo de una lista  
    \item En el documento fuente colocamos  
        una pequeña indentación en el comando  
        item, para entender más facilmente el  
        código, aunque ésto no es imprescindible.  
    \item Si una entrada es muy larga, nótese  
        cómo el párrafo continua manteniendo una  
        indentación fija.  
\end{itemize}  
  
Párrafo posterior ...
```

... párrafo anterior

- Esto es un ejemplo de una lista
- En el documento fuente colocamos una pequeña indentación en el comando item, para entender más facilmente el código, aunque ésto no es imprescindible.
- Si una entrada es muy larga, nótese cómo el párrafo continua manteniendo una indentación fija.

Párrafo posterior ...

Las listas pueden anidarse, con entradas que posean subentradas, hasta una profundidad de cuatro subniveles (más anidamiento no está contemplado, y obtendríamos un error de compilación). Para cada subnivel se utiliza un símbolo de marcación diferente, y se añade una indentación extra con respecto al nivel anterior. Observar el siguiente ejemplo donde se anidan hasta 4 subniveles:

```
\begin{itemize}
\item 1a entrada del primer nivel
\item 2a entrada del primer nivel
  \begin{itemize}
    \item 1a entrada del segundo nivel
    \item 2a entrada del segundo nivel
      \begin{itemize}
        \item 1a entrada del tercer nivel
        \item 2a entrada del tercer nivel
          \begin{itemize}
            \item 1a entrada del cuarto nivel
            \item 2a entrada del cuarto nivel
          \end{itemize}
        \end{itemize}
      \end{itemize}
    \end{itemize}
  \end{itemize}
\end{itemize}
```

- 1^a entrada del primer nivel
- 2^a entrada del primer nivel
 - 1^a entrada del segundo nivel
 - 2^a entrada del segundo nivel
 - 1^a entrada del tercer nivel
 - 2^a entrada del tercer nivel
 - ◊ 1^a entrada del cuarto nivel
 - ◊ 2^a entrada del cuarto nivel

Como indicábamos al principio, el uso de `\usepackage[spanish]{babel}` cambia los indicadores predefinidos para cada nivel (`•, -, *, ·` para L^AT_EX estándar) a: ■, ●, ○, ◊. L^AT_EX nos permite cambiar a nuestro gusto estos marcadores, cosa que podemos hacer con la siguientes instrucciones:

- `\renewcommand{\labelitemi}{Nuevo-marcador}`
- `\renewcommand{\labelitemii}{Nuevo-marcador}`
- `\renewcommand{\labelitemiii}{Nuevo-marcador}`
- `\renewcommand{\labelitemiv}{Nuevo-marcador}`

donde **Nuevo-marcador** designa al código L^AT_EX del nuevo indicador, y `\labelitemi`, `\labelitemii`, etc... son los comandos L^AT_EX que se encargan de escribir los marcadores para los primeros, segundos, etc... niveles de enumeración. Por ejemplo, incluyendo el paquete `pifont` de símbolos especiales (colocar `\usepackage{pifont}` en el preámbulo) podemos obtener lo siguiente:

```
\renewcommand{\labelitemi}{\ding{42}}
\renewcommand{\labelitemii}{\ding{43}}
\begin{itemize}
\item 1a entrada del primer nivel
\item 2a entrada del primer nivel
  \begin{itemize}
    \item 1a entrada del segundo nivel
    \item 2a entrada del segundo nivel
  \end{itemize}
\end{itemize}
```

- ☛ 1^a entrada del primer nivel
- ☛ 2^a entrada del primer nivel
 - ☛ 1^a entrada del segundo nivel
 - ☛ 2^a entrada del segundo nivel

Nótese que si colocamos el comando `\renewcommand{\labelitemi}{Nuevo-marcador}` *futura* de un entorno `itemize`, afectará a todas las posteriores listas `itemize` del documento. Si solamente se desea que afecte a una lista en particular, las nuevas redefiniciones se pueden colocar *dentro* de entorno, de la forma siguiente:

```
\begin{itemize}
\renewcommand{\labelitemi}{\ding{42}}
\item Primera entrada del primer nivel
\item Segunda entrada del primer nivel
\begin{itemize}
\renewcommand{\labelitemii}{\ding{43}}
\item Primera entrada del segundo nivel
\item Segunda entrada del segundo nivel
\end{itemize}
\end{itemize}
```

(Un método alternativo sería incluir los comandos `\renewcommand{\labelitemi}{...}` y el entorno `itemize` correspondiente dentro de un grupo, añadiendo llaves al principio y el final)

SECCIÓN 2

Listas numeradas: El entorno `enumerate`

Para obtener listas numeradas, se utiliza (de forma análoga al `itemize`) el entorno `enumerate`, que igualmente es anidable hasta cuatro subniveles:

```
\begin{enumerate}
\item 1a entrada del primer nivel
\item 2a entrada del primer nivel
\begin{enumerate}
\item 1a entrada del segundo nivel
\item 2a entrada del segundo nivel
\begin{enumerate}
\item 1a entrada del tercer nivel
\item 2a entrada del tercer nivel
\begin{enumerate}
\item 1a entrada del cuarto nivel
\item 2a entrada del cuarto nivel
\end{enumerate}
\end{enumerate}
\end{enumerate}
\end{enumerate}
```

1. 1^a entrada del primer nivel
2. 2^a entrada del primer nivel
 - a) 1^a entrada del segundo nivel
 - b) 2^a entrada del segundo nivel
 - 1) 1^a entrada del tercer nivel
 - 2) 2^a entrada del tercer nivel
 - a' 1^a entrada del cuarto nivel
 - b' 2^a entrada del cuarto nivel

En éste caso, cada comando `\item` tiene el efecto de ir añadiendo entradas contabilizadas según valores crecientes de un contador. Éste contador, dependiendo del nivel de anidamiento del entorno `enumerate`, posee diversas *representaciones* (ó formatos); para el primer nivel se utiliza 1., 2., 3., etc..., para el segundo a), b), c), etc..., y otras representaciones distintas para los niveles tercero y cuarto. Es importante tener en cuenta que el formato de las etiquetas para cada tipo de nivel son cambiadas por la opción `spanish` de `babel`. El ejemplo anterior ilustra el resultado en tales circunstancias. Para L^AT_EX estándar se obtendría:

- 1., 2., 3., ...
- (a), (b), (c), ...

- i., ii., iii., ...
- A., B., C., ...

Existen métodos para manipular libremente tal formato de las etiquetas de enumeración, pero para describirlos necesitamos conocimientos relativamente avanzados sobre las características de los contadores en L^AT_EX, que serán abordados más adelante. De momento, propondremos una alternativa simple y elegante (aunque no muy pontente), usando el paquete `enumerate` (**para lo cual, como siempre, se debe de declarar `\usepackage{enumerate}` en el preámbulo del documento.**)

Éste paquete permite colocar un argumento optativo en el entorno `enumerate`, de la forma: `\begin{enumerate}[argumento-optativo]`, donde en el argumento se debe elegir un carácter clave: 1, i, I, a y A, (números, números romanos, y letras) sobre el cual se basará la enumeración. Lo elegante del paquete es que permite combinar el carácter clave con instrucciones L^AT_EX. Veamos un ejemplo:

Veamos ahora los pasos necesarios para escribir un documento en L^AT_EX:

```
\begin{enumerate}[\hspace*{0.5cm}%
\bfseries P{a}so 1]
\item Preparar documento fuente tex
\item Compilarlo para producir dvi
\begin{enumerate}[(a)]
\item Visualizar con visor de dvi
\item Corregir errores
\item Recompilar
\end{enumerate}
\item Convertir a .ps con dvips
\end{enumerate}
```

Veamos ahora los pasos necesarios para escribir un documento en L^AT_EX:

Paso 1 Preparar documento fuente tex

Paso 2 Compilarlo para producir dvi

- (a) Visualizar con visor de dvi
- (b) Corregir errores
- (c) Recompilar

Paso 3 Convertir a .ps con dvips

¿Para qué se ha utilizado `\hspace*{0.5cm}` en el argumento optativo? La razón reside en que, a diferencia del entorno `itemize`, en el entorno `enumerate` se suprime la sangría para el primer nivel de elementos. Otro punto importante es el porqué de la ‘a’ entre llaves en “Paso”. Es interesante ver qué ocurre si eliminamos las llaves (ejercicio).

SECCIÓN 3 – El entorno `description`

El entorno `description` se puede considerar como una generalización del entorno `itemize`, en la que las etiquetas pueden ser libremente configuradas, lo cual es adecuado para descripciones. Por ejemplo:

```
\begin{description}
\item[TeX] Un procesador de textos
\item[Word] Otro procesador de textos
\item .....
\end{description}
```

TeX Un procesador de textos

Word Otro procesador de textos

.....

Se observa que, por defecto, las etiquetas son escritas en negrita. Podemos cambiar esto sin problemas, por ejemplo:

```
\renewcommand{\descriptionlabel}[1]%
{\hspace*{0.5cm}\textsf{#1}}
\begin{description}
    \item[\TeX] Un procesador de textos
    \item[Word] Otro procesador de textos
    \item .....
\end{description}
```

\TeX Un procesador de textos

Word Otro procesador de textos

.....

lo cual pone los objetos a describir en `sanserif`, además de introducir una pequeña sangría¹ (ausente en el primer nivel, al igual que en `enumerate`).

Es importante mencionar que en cualquiera de los otros entornos (`itemize` y `enumerate`) puede en cualquier momento cambiarse la etiqueta por defecto, añadiendo la nueva etiqueta entre paréntesis tras el comando `\item`:

```
\begin{itemize}
    \item 1a entrada
    \item[$\rightarrow$] 2a entrada
    \item 3a entrada
        \begin{enumerate}
            \item 1o entrada
            \item 2a entrada
            \item[$\clubsuit$] 3a entrada
        \end{enumerate}
\end{itemize}
```

■ 1^a entrada

→ 2^a entrada

■ 3^a entrada

1. 1^o entrada

2. 2^a entrada

♣ 3^a entrada

Como también se puede ver en el ejemplo anterior, no existe ningún problema en anidar listas de distintos tipos (respetando siempre, por supuesto, el límite de 4 niveles de anidamiento).

SECCIÓN 4

Listas personalizadas: el entorno `list`

A la hora de formatear las entradas de una lista, L^AT_EX utiliza ciertos valores predeterminados para la colocación de los párrafos que conforman cada entrada, la distancia de las etiquetas a la entrada, etc... Todos estos valores (ó variables de longitud) son pueden ser modificados mediante reasignaciones de longitud con cualquiera de las sintaxis alternativas:

¹El comando `\renewcommand{\descriptionlabel}[1]{Acciones del comando}` es un ejemplo de definición de comando (en éste caso, redefinición) dependiente de un argumento; el número 1 entre corchetes indica que el comando es dependiente de 1 argumento variable, que dentro del conjunto de comandos L^AT_EX en “Acciones del comando” se denota con el símbolo clave #1. En el presente caso vemos que las acciones son dejar un espacio de 0.5cm y poner en tipo sanserif “#1”, ésto es, el argumento del comando `\descriptionlabel`. Se ampliarán éstos conceptos en el Capítulo dedicado a la programación en L^AT_EX.

- `\setlength{\Longitud}{Xmm}` (ó cm, pt, etc...)
- `\Longitud=Xmm`
- `\Longitud Xmm`

que asignan el nuevo valor `Xmm` a la variable `\Longitud`. Otro método de asignación de longitudes es el comando `\addtolength{\Longitud}{Xmm}` que *incrementa* en `Xmm` el valor de la variable de longitud. Éste método es particularmente útil en casos en que no conoczamos a priori el valor preestablecido de una longitud, y queramos modificarla sin arriesgarnos a introducir valores desproporcionados.

Para crear listas tipo `itemize` con parámetros configurables, podemos emplear el entorno `list`, con la siguiente sintaxis:

```
\begin{list}{Etiqueta}{Declaraciones}
\item Texto...
\item Texto...
\end{list}
```

donde el parámetro `Etiqueta` especifica el objeto que debe emplearse como etiqueta, y dentro del apartado `Declaraciones` debemos incluir comandos de redefinición de longitud. La Figura 1 muestra gráficamente, en el caso de un entorno `list` típico, las distintas longitudes que utiliza L^AT_EX para estructurar la lista, y que podemos modificar a voluntad. Las longitudes `\leftmargin` y `\rightmargin` definen los “márgenes” de la lista de ítems con respecto a la anchura de texto estándar del documento. Es fundamental tener en cuenta que `\leftmargin` se define como el espacio horizontal de indentación de la lista *respecto al entorno anterior*. Su valor depende del nivel de lista en el que nos encontremos; para el primer nivel, toma el valor de la longitud `\leftmargini`, para el segundo, `\leftmarginii`, y así hasta `\leftmarginiv` para el cuarto nivel. Reasignando éstas longitudes (antes de comenzar la anidación de las listas) podemos por tanto cambiar la indentación según nuestras preferencias particulares (ver ejemplos a continuación).

Por otra parte, `\topsep` permite ajustar los espacios verticales anteriores y posteriores a la lista, `\itemsep` la separación vertical entre ítems. Para ajustar la posición de las etiquetas respecto al texto de la entrada, se puede ajustar el parámetro `\labelsep`, y para la indentación de los párrafos que conforman una entrada se utiliza `\itemindent`. Los siguientes ejemplos ilustran varias posibilidades de uso de éstos parámetros:

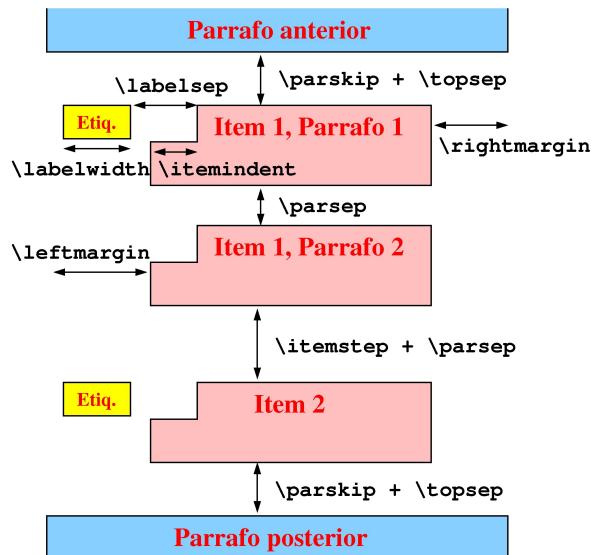


Figura 1: Variables de longitud en entornos de tipo lista

Lista estándar; nótese cómo al dejar vacío la opción para etiquetas éstas desaparecen.

```
... texto anterior
\begin{list}{}{}
  \item Primer ítem de una lista con
    valores estándar para las longitudes
  \par Segundo párrafo del primer
    ítem de la lista
  \item Segundo ítem de la lista.
  \begin{list}{}{}
    \item Primer ítem de una sublista
      anidada dentro de la lista principal
    \item Segundo ítem de la sublista
  \end{list}
\end{list}
Texto posterior ...
```

```
... texto anterior
Primer ítem de una lista con valores estándar para las longitudes
Segundo párrafo del primer ítem de la lista
Segundo ítem de la lista.

Primer ítem de una sublista anidada dentro de la lista principal
Segundo ítem de la sublista

Texto posterior ...
```

Ejemplo donde se modifica la separación con el texto circundante (`\topsep`), la indentación de los ítems (`\itemindent`), la separación entre ítems (`\itemsep`), y la separación etiqueta-ítem (`\labelsep`).

```
... texto anterior
\begin{list}{\textbullet}{%
  \addtolength{\topsep}{3mm}%
  \addtolength{\labelsep}{1mm}%
  \addtolength{\itemsep}{-2mm}%
  \setlength{\itemindent}{3mm}%
  \item Primer ítem de una lista con
    valores de formato modificados
  \par Segundo párrafo del primer
    ítem de la lista
  \item Segundo ítem de la lista.
  \begin{list}{$\blacksquare$}{%
    \item Primer ítem de una sublista
      anidada dentro de la lista principal
    \item Segundo ítem de la sublista
  \end{list}
\end{list}
Texto posterior ...
```

```
... texto anterior
• Primer ítem de una lista con valores de
  formato modificados
Segundo párrafo del primer ítem de la lista
• Segundo ítem de la lista.

■ Primer ítem de una sublista anidada
  dentro de la lista principal
■ Segundo ítem de la sublista

Texto posterior ...
```

En el ejemplo siguiente, obsérvese cómo se ajustan las indentaciones para listas de primer y segundo nivel; además, se ajustan las separaciones entre ítems a través de `\itemstep`

```
\setlength{\leftmargini}{0pt}
\setlength{\leftmarginii}{30pt}
\begin{list}{\textbullet}%
\addtolength{\itemsep}{-2mm}%
\setlength{\itemindent}{2mm}%
\item Primer ítem de una lista con
valores de formato modificados \par
Segundo párrafo del primer ítem
\item Segundo ítem de la lista.
\begin{list}{$\scriptscriptstyle \blacksquare$}%
\setlength{\itemindent}{3mm}%
\addtolength{\itemsep}{1mm}%
\item Primer ítem de una sublista
anidada dentro de la lista principal
\item Segundo ítem de la sublista
\end{list}
\end{list}
```

- Primer ítem de una lista con valores de formato modificados
- Segundo párrafo del primer ítem
- Segundo ítem de la lista.
 - Primer ítem de una sublista anidada dentro de la lista principal
 - Segundo ítem de la sublista

Finalmente, hay que tener en cuenta que algunos de éstos parámetros de configuración no son exclusivos del entorno `list` y que pueden emplearse en los entornos `itemize` ó `enumerate`, como por ejemplo `\itemstep` ó `\leftmargini`, `\leftmarginii`, etc... Véase el siguiente ejemplo en el que se ajustan las indentaciones y el espaciado entre ítems:

```
\setlength{\leftmargini}{10mm}
\setlength{\leftmarginii}{15mm}
\begin{enumerate}
\addtolength{\itemsep}{-2mm}
\item Funciones de variable compleja
\item Teorema de Cauchy-Goursat
\begin{itemize}
\addtolength{\itemsep}{-1mm}
\item Enunciado
\item Demostración
\end{itemize}
\end{enumerate}
```

1. Funciones de variable compleja
2. Teorema de Cauchy-Goursat
 - Enunciado
 - Demostración

SECCIÓN 5

Construcción de tablas: el entorno `tabular`

Para la construcción de tablas, la herramienta básica es el entorno `tabular`. La estructura fundamental de una tabla es la siguiente:

```
\begin{tabular}[Posición]{FormatoColumnas}
xxx & xxx & xxx & ... & xxx \\
xxx & xxx & xxx & ... & xxx \\
... & ... & ... & ... & ...
xxx & xxx & xxx & xxx & xxx
\end{tabular}
```

- & Separador entre columnas, que marca el fin de una casilla y el principio de la siguiente. Podemos dejar una casilla vacía con &&. Nótese que el tamaño de las columnas no tiene nada que ver con la distancia entre los separadores y el texto de la casilla: &xxx&, & xxx & ó & xxx & producirán exactamente el mismo resultado. En general, la anchura de la columna está determinada por longitudes predefinidas y, fundamentalmente, por el tamaño máximo que alcanzan los elementos de una columna (con excepción de las columnas tipo párrafo ó páncho en las que la anchura se fija de antemano).
- \\\ Cambio de fila: es la instrucción para comenzar una fila nueva. No es estrictamente necesaria para la última fila, a menos que se quiera terminar con una línea horizontal (comando: \\\hline)

Posición Argumento **optativo** que especifica la posición de la tabla respecto al texto en el que se incluye. Puede tomar los valores t (top), c (center; valor por defecto) ó b (bottom), según qué parte de la tabla se alinee con respecto a la línea de texto principal. Por ejemplo:

aquí se inserta \begin{tabular}[t]{cc} 11 & 12 \\ 21 & 22 \end{tabular} una pequeña tabla	aquí se inserta \begin{tabular}[c]{cc} 11 & 12 \\ 21 & 22 \end{tabular} una pequeña tabla	aquí se inserta \begin{tabular}[b]{cc} 11 & 12 \\ 21 & 22 \end{tabular} una pequeña tabla
aquí se inserta 11 12 una pe- 21 22 queña tabla	aquí se inserta 11 12 una pe- 21 22 queña tabla	aquí se inserta 11 12 21 22 una pe- queña tabla

FormatoColumnas Argumento **fundamental** que utilizamos para dar estructura a la tabla. Se compone de dos elementos fundamentales:

Especificadores Debe haber un especificador por cada columna de la tabla. Tenemos las siguientes opciones:

l, r, c Introduce una nueva columna justificada a la izquierda, derecha ó centro, respectivamente. La anchura de columna se determina automáticamente a partir del tamaño máximo de sus elementos.

p{Ancho} Se utiliza en caso de que tengamos un texto largo en una de las columnas, creando una columna de anchura fija *Ancho*

Separadores Son optativos, y se encargan de especificar cómo deben separarse las columnas

| Produce una barra vertical separando columnas.

@{Objeto} Suprime el espacio entre columnas e inserta en su lugar el *Objeto* declarado.

Para añadir líneas horizontales se pueden utilizar:

\hline Traza una línea horizontal a lo largo de toda la tabla

\cline{x-y} Traza una línea horizontal desde la columna x hasta la columna y, ambas inclusive

Finalmente, el comando \multicolumn{NúmeroColumnas}{FormatoColumnas}{Objeto} se puede utilizar para agrupar, dentro de una fila, el contenido de varias columnas (argumento NúmeroColumnas) en una sola (véase el ejemplo a continuación). Otros comandos útiles son \extracolsep{Longitud}, que añade un espacio adicional Longitud a la distancia entre columnas calculada automáticamente por L^AT_EX, y \setlength{\extrarowheight}{Longitud}, que añade el espacio adicional Longitud a la distancia estándar entre filas (se debe cargar el paquete array para que éste último comando funcione).

5.1. Ejemplos

Importaciones (en millones de \euro) de carne y verduras:

```
\begin{tabular}{ccc}
Pais & Carne & Verduras \\
España & 1390 & 980 \\
Francia & 1504 & 3020 \\
Italia & 2010 & 1040 \\
\end{tabular}
```

Importaciones (en millones de €) de carne y verduras:

Pais	Carne	Verduras
España	1390	980
Francia	1504	3020
Italia	2010	1040

Imaginemos que queremos:

Planeta	Distancia al sol (millones km)	
	Máxima	Mínima
Mercurio	69.4	46.8
Venus	109.0	107.6
Tierra	152.6	147.4

Podemos intentar:

```
\begin{tabular}{|l|r|r|}
\hline
& \multicolumn{2}{c}{Distancia al sol} \\
\hline
Planeta & \multicolumn{2}{c}{(millones km)} & \cline{2-3}
& Máxima & Mínima \\
\hline
Mercurio & 69.4 & 46.8 \\
Venus & 109.0 & 107.6 \\
Tierra & 152.6 & 147.4 \\
\hline
\end{tabular}
```

lo que nos daría:

Planeta	Distancia al sol (millones km)	
	Máxima	Mínima
Mercurio	69.4	46.8
Venus	109.0	107.6
Tierra	152.6	147.4

pero no queda bien!

la solución está en añadir una barra separadora | tras el argumento “c” en `\multicolumn`, con lo que quedaría:

Planeta	Distancia al sol (millones km)	
	Máxima	Mínima
Mercurio	69.4	46.8
Venus	109.0	107.6
Tierra	152.6	147.4

En éste último ejemplo se observa una de las utilidades principales del comando `\multicolumn`; aparte de servir para agrupar columnas, podemos emplearlo para cambiar el formato est醤dar de columna (tanto alineaci髇 como separadores) de una casilla de la tabla en particular. Pueden encontrarse m醩 ejemplos de esto en el documento ejemplo asociado a esta secci髇.

5.2. Parámetros de control de formato de tablas

Podemos modificar la apariencia de una tabla con los parámetros descritos a continuación. Nótese que, si queremos hacer el efecto de un cambio de parámetros *local*, deberíamos incluir la instrucción de la siguiente forma:

```
{
\setlength{\NombreParámetro}{ValorNuevo}
\begin{tabular}
...
\end{tabular}
}
```

esto es, incluyéndola dentro de un grupo que agrupe el entorno tabular.

- `\arraystretch` → Factor que controla la separación vertical entre filas; el valor por defecto es 1, igual al valor de `\baselinestretch`. Definiéndolo como 1.5 obtendremos una tabla con columnas un 50 % más altas. Debido a que `\arraystretch` es un *comando*, debemos redefinirlo con la sintaxis: `\renewcommand*{\arraystretch}{NuevoFactor}`
- `\tabcolsep` → 1/2 de la separación horizontal entre columnas, así como el espacio horizontal al comienzo y final de la tabla. Dado que es una *longitud*, se redefine mediante: `\setlength{\tabcolsep}{NuevaSeparación}`
- `\arraycolsep` → Igual que el anterior, para el entorno matemático `array`. Siendo longitud, se redefine análogamente al caso anterior. Para el resto, también longitudes, se aplica lo mismo.
- `\arrayrulewidth` → Grosor de las líneas horizontales y verticales en el entorno `tabular`. Por defecto, 0.4 pt.
- `\doublerulesep` → Separación entre rayas dobles, horizontales ó verticales.

Unos ejemplos para ilustrar lo anterior:

```
\begin{center}
{\setlength{\tabcolsep}{10pt}
\begin{tabular}{||c|c||}\hline
e & ef \\
efg & efg \\ \hline
\end{tabular}}
\par \bigskip
{\setlength{\arrayrulewidth}{3pt}
\renewcommand*\arraystretch{2}
\begin{tabular}{|c|c|} \hline
i & ij \\
ijk & ijk \\ \hline
\end{tabular}} \par\bigskip
{\setlength{\arrayrulewidth}{2pt}
\setlength{\doublerulesep}{2pt}
\begin{tabular}{||cc||} \hline
m & mn \\
mno & mnop \\ \hline
\end{tabular}}
\end{center}
```

e	ef
efg	efgh

i	ij
ijk	ijkl

m	mn
mno	mnop

Debemos destacar que, si no se carga en el preámbulo el paquete `array`, (que se describirá en un capítulo posterior) el uso de líneas gruesas crea problemas obteniéndose uniones imperfectas.

5.3. La script Excel2Latex

Si trabajamos frecuentemente con hojas de cálculo como MS-Excel, puede sernos muy útil el instalar bajo Excel la herramienta **Excel2Latex**, que proporciona una forma sencilla de convertir nuestros datos en formato Excel a entornos tabular de LATEX. Para instalarla se debe descargar del CTAN el archivo **Excel2LaTeX.xla**
(en la dirección <http://www.ctan.org/tex-archive/support/excel2latex/>)

y posteriormente copiarlo al directorio de ADDINS de office (que normalmente se encuentra en: C:\Documents and Settings\usuario\Datos de programa\Microsoft\AddIns, aunque su ubicación puede variar según la ubicación de Office y la versión de Windows). Se abre entonces una vez desde Excel (habilitando la opción de utilizar macros)² una vez, y a partir de entonces lo tendremos ya disponible como una opción más en “Herramientas”.

Para su uso, simplemente se selecciona una zona de un documento excel, se ejecuta la script “Excel2LaTeX”, y obtendremos una ventana emergente con el código LATEX (en forma de entorno tabular) que podremos entonces copiar y pegar a nuestro documento LATEX.

²Dependiendo de la instalación de Office, puede ser necesario el bajar el nivel de seguridad para las macros en las opciones de Office

Apuntes de Latex

Capítulo 3: Fórmulas matemáticas – Conceptos básicos

En éste capítulo se exponen de forma breve unas nociones básicas acerca de la escritura de expresiones matemáticas. Es importante, para disponer de todas las capacidades matemáticas de L^AT_EX en un documento, cargar con `\usepackage{...}` los paquetes **amsmath** (capacidades matemáticas extra) y **amssymb** (librería de símbolos). Como fuente de documentación adicional, se recomienda consultar la guía “Mathmode” de escritura matemática, colgada en el apartado de BIBLIOGRAFÍA de la web de la asignatura.

SECCIÓN 1

Modos matemáticos tipo texto y extendido.

A la hora de escribir expresiones matemáticas de forma elegante y precisa, T_EX dispone de un modo de escritura especial, el modo matemático. Así por ejemplo, para tener:

La ecuación de una recta en el plano cartesiano es de la forma $ax + by + c = 0$, donde a, b, c son constantes.

escribiríamos:

La ecuación de una recta en el plano cartesiano es de la forma
`$ax+by+c=0$`, donde `a, b, c` son constantes.

`$` es el comando a utilizar para entrar y salir del modo matemático *en modo texto* (es decir, cuando queremos las expresiones matemáticas escritas dentro del texto principal, con un tamaño apropiado para ello). En el ejemplo anterior vemos varias cosas importantes; primero, aunque tecleamos `$ax+by+c=0$` sin espacios, T_EX introduce espacios en la fórmula de acuerdo a *sus propias reglas* (teclar `$ ax + by + c = 0$` produciría exactamente el mismo resultado); en general, en modo matemático T_EX asigna espacios entre variables matemáticas de acuerdo con los distintos tipos de separadores (`=, +, <, ∫, ...`) que encuentra. Además, los caracteres de texto son escritos *en itálica*.

Por contra, nótese la diferencia entre:

`a, b, c` → a, b, c

`a, b, c` → a, b, c

No hay espacios entre comas en el primer caso; hemos de salir del modo matemático para introducirlos.

Si queremos escribir expresiones matemáticas resaltadas, es decir, separadas del texto principal y con un tamaño mayor, podemos utilizar:

La ecuación de una recta en el plano cartesiano es de la forma

$\$ax+by+c=0$$$

donde a , b , c son constantes.

que produciría:

La ecuación de una recta en el plano cartesiano es de la forma

$$ax + by + c = 0$$

donde a , b , c son constantes.

$\$$ es el comando a utilizar para entrar y salir del modo matemático *resaltado* (es decir, cuando queremos las expresiones matemáticas escritas *frente* al texto principal, con un tamaño mayor).

Hay tres formas análogas para delimitar cada uno de los dos tipos (texto y resaltado):

Texto	Resaltado
$\$... \$$	$\$... \$$
$\backslash(... \backslash)$	$\backslash[... \backslash]$
$\begin{math} ... \end{math}$	$\begin{displaymath} ... \end{displaymath}$

En el ejemplo siguiente puede verse más claramente la diferencia entre ambos modos:

Tenemos la equivalencia $\frac{a}{b}=\frac{c}{d}$, válida para todo a , b , c , d

Tenemos la equivalencia $\frac{a}{b}=\frac{c}{d}$ válida para todo a , b , c , d

Tenemos la equivalencia $\frac{a}{b} = \frac{c}{d}$, válida para todo a , b , c , d

Tenemos la equivalencia

$$\frac{a}{b} = \frac{c}{d}$$

válida para todo a , b , c , d

Otra alternativa para escribir fórmulas en modo resaltado es el entorno `equation`, como muestra el siguiente ejemplo:

La ecuación de una recta en el plano cartesiano es de la forma

$\begin{equation*}$

$ax+by+c=0$

$\end{equation*}$

donde a , b , c son constantes.

que produciría:

La ecuación de una recta en el plano cartesiano es de la forma

$$ax + by + c = 0$$

donde a, b, c son constantes.

¿Cuál es el efecto del “*” tras equation? Eliminándolo obtenemos lo siguiente:

La ecuación de una recta en el plano cartesiano es de la forma

$$ax + by + c = 0 \quad (1)$$

donde a, b, c son constantes.

La ecuación es entonces *numerada*. L^AT_EX utiliza un contador para numerar ecuaciones, según la sección a la que pertenezcan (en el formato `article`) ó según el capítulo y sección (en el formato `book`). En capítulos posteriores, se mostrará cómo referenciar ecuaciones, escribir ecuaciones en varias líneas, manejar teoremas, etc... Por el momento nos limitaremos simplemente a la escritura en sí de la diversa simbología matemática que soporta L^AT_EX.

SECCIÓN 2

Símbolos

Las siguientes tablas proporcionan los comandos necesarios para obtener una amplia variedad de símbolos matemáticos. Una gran parte de ellos puede obtenerse a través del icono “Σ” en el programa WinEdt, que abre una serie de pestañas con una aplica colección de símbolos. La colección completa de símbolos matemáticos puede consultarse en la “Comprehensive LaTeX symbol list”, colgada en la página de la asignatura. Es importante remarcar que, debido a que son símbolos matemáticos, su utilización en medio del texto requiere incluirlos entre signos \$.

Tabla 1: Letras griegas

α	<code>\alpha</code>	θ	<code>\theta</code>	\circ	<code>\circ</code>	τ	<code>\tau</code>
β	<code>\beta</code>	ϑ	<code>\vartheta</code>	π	<code>\pi</code>	υ	<code>\upsilon</code>
γ	<code>\gamma</code>	ι	<code>\iota</code>	ω	<code>\omega</code>	ϕ	<code>\phi</code>
δ	<code>\delta</code>	κ	<code>\kappa</code>	ρ	<code>\rho</code>	φ	<code>\varphi</code>
ϵ	<code>\epsilon</code>	λ	<code>\lambda</code>	ϱ	<code>\varrho</code>	χ	<code>\chi</code>
ε	<code>\varepsilon</code>	μ	<code>\mu</code>	σ	<code>\sigma</code>	ψ	<code>\psi</code>
ζ	<code>\zeta</code>	ν	<code>\nu</code>	ς	<code>\varsigma</code>	ω	<code>\omega</code>
η	<code>\eta</code>	ξ	<code>\xi</code>				
Γ	<code>\Gamma</code>	Λ	<code>\Lambda</code>	Σ	<code>\Sigma</code>	Ψ	<code>\Psi</code>
Δ	<code>\Delta</code>	Ξ	<code>\Xi</code>	Υ	<code>\Upsilon</code>	Ω	<code>\Omega</code>
Θ	<code>\Theta</code>	Π	<code>\Pi</code>	Φ	<code>\Phi</code>		

Tabla 2: Operadores binarios

\pm	<code>\pm</code>	\cap	<code>\cap</code>	\diamond	<code>\diamond</code>	\oplus	<code>\oplus</code>
\mp	<code>\mp</code>	\cup	<code>\cup</code>	\triangle	<code>\bigtriangleup</code>	\ominus	<code>\ominus</code>
\times	<code>\times</code>	\oplus	<code>\uplus</code>	\triangledown	<code>\bigtriangledown</code>	\otimes	<code>\otimes</code>
\div	<code>\div</code>	\sqcap	<code>\sqcap</code>	\triangleleft	<code>\triangleleft</code>	\oslash	<code>\oslash</code>
$*$	<code>\ast</code>	\sqcup	<code>\sqcup</code>	\triangleright	<code>\triangleright</code>	\odot	<code>\odot</code>
\star	<code>\star</code>	\vee	<code>\vee</code>	\lhd	<code>\lhd</code>	\bigcirc	<code>\bigcirc</code>
\circ	<code>\circ</code>	\wedge	<code>\wedge</code>	\rhd	<code>\rhd</code>	\dagger	<code>\dagger</code>
\bullet	<code>\bullet</code>	\setminus	<code>\setminus</code>	\unlhd	<code>\unlhd</code>	\ddagger	<code>\ddagger</code>
\cdot	<code>\cdot</code>	\wr	<code>\wr</code>	\unrhd	<code>\unrhd</code>	\amalg	<code>\amalg</code>
$+$	<code>+</code>	$-$	<code>-</code>				

Tabla 3: Operadores de relación

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

Tabla 4: Signos de puntuación

,

,

,

;

:

:

\colon

.

\ldotp

.

\cdotp

Tabla 5: Símbolos de flechas

\leftarrow	<code>\leftarrow</code>	\longleftarrow	<code>\longleftarrow</code>	\uparrow	<code>\uparrow</code>
\Leftarrow	<code>\Leftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\rightarrow	<code>\rightarrow</code>	\longrightarrow	<code>\longrightarrow</code>	\downarrow	<code>\downarrow</code>
\Rightarrow	<code>\Rightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>	\Downarrow	<code>\Downarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>	\updownarrow	<code>\updownarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>	\nearrow	<code>\nearrow</code>
\hookleftarrow	<code>\hookleftarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>	\searrow	<code>\searrow</code>
\leftharpoonup	<code>\leftharpoonup</code>	\rightharpoonup	<code>\rightharpoonup</code>	\swarrow	<code>\swarrow</code>
\leftharpoondown	<code>\leftharpoondown</code>	\rightharpoondown	<code>\rightharpoondown</code>	\nwarrow	<code>\nwarrow</code>
\rightleftharpoons	<code>\rightleftharpoons</code>	\leadsto	<code>\leadsto</code>		

Tabla 6: Símbolos varios

...	\ldots	...	\cdots	:	\vdots	\ddots	\ddots
\aleph	\aleph	'	\prime	\forall	\forallall	\infty	\infty
\hbar	\hbar	\emptyset	\emptyset	\exists	\existsists	\square	\Box
\imath	\imath	\nabla	\nabla	\neg	\neg	\diamond	\Diamond
\jmath	\jmath	\surd	\surd	\flat	\flat	\triangle	\triangle
\ell	\ell	\top	\top	\natural	\natural	\clubsuit	\clubsuit
\wp	\wp	\bot	\bot	\sharp	\sharp	\diamondsuit	\diamondsuit
\Re	\Re	\backslash\lvert	\backslash\lvert	\backslash\backslash	\backslash\backslash	\heartsuit	\heartsuit
\Im	\Im	\angle	\angle	\partial	\partial	\spadesuit	\spadesuit
\mho	\mho	.	.				

Tabla 7: Operadores de tamaño variable

\sum	\sum	\bigcap	\bigcap	\bigodot	\bigodot
\prod	\prod	\bigcup	\bigcup	\bigotimes	\bigotimes
\coprod	\coprod	\bigsqcup	\bigsqcup	\bigoplus	\bigoplus
\int	\int	\bigvee	\bigvee	\biguplus	\biguplus
\oint	\oint	\bigwedge	\bigwedge		

Tabla 8: Funciones

\arccos	\cos	\csc	\exp	\ker	\limsup	\min	\sinh
\arcsin	\cosh	\deg	\gcd	\lg	\ln	\Pr	\sup
\arctan	\cot	\det	\hom	\lim	\log	\sec	\tan
\arg	\coth	\dim	\inf	\liminf	\max	\sin	\tanh

Tabla 9: Delimitadores

(())	\uparrow	\uparrow	\uparrow	\Uparrow
[[]]	\downarrow	\downarrow	\downarrow	\Downarrow
{	\{	}	\}	\updownarrow	\updownarrow	\updownarrow	\Updownarrow
\lfloor	\lfloor	\rfloor	\rfloor	\lceil	\lceil	\rceil	\rceil
\langle	\langle	\rangle	\rangle	/	/	\backslash	\backslash
				\mid	\mid		

Tabla 10: Delimitadores grandes

\rmoustache	\rmoustache	\lrmoustache	\rgroup	\lgroup
\arrowvert	\arrowvert	\Arrowvert	\bracevert	\bracevert

Tabla 11: Acentos en modo matemático

\hat{a}	\hat{a}	\acute{a}	\acute{a}	\bar{a}	\bar{a}	\dot{a}	\dot{a}
\breve{a}	\breve{a}	\check{a}	\check{a}	\grave{a}	\grave{a}	\vec{a}	\vec{a}

$\ddot{a} \quad \text{\ddot{a}}$
 $\tilde{a} \quad \text{\tilde{a}}$

Tabla 12: Otras construcciones

\widetilde{abc}	\widehat{abc}
\overleftarrow{abc}	\overrightarrow{abc}
\overline{abc}	\underline{abc}
\overbrace{abc}	\underbrace{abc}
\sqrt{abc}	$\sqrt[n]{abc}$
f'	$\frac{abc}{xyz}$

Tabla 13: Delimitadores AMS

 $\ulcorner \quad \urcorner \quad \llcorner \quad \lrcorner$

Tabla 14: Flechas AMS

\dashrightarrow	\dashrightarrow	\dashleftarrow
\leftleftarrows	\leftrightarrows	\leftrightsquigarrow
\Lleftarrow	\twoheadleftarrow	\rightleftarrows
\leftarrowtail	\looparrowleft	\rightleftarrows
\leftrightharpoons	\curvearrowleft	\twoheadrightarrow
\circlearrowleft	\Lsh	\rightarrowtail
\upuparrows	\upharpoonleft	\looparrowright
\downharpoonleft	\multimap	\curvearrowright
\rightsquigarrow	\Rsh	\rightarrowtail
\rightleftarrows	\upharpoonright	\rightsquigarrow
\rightleftarrows	\twoheadrightarrow	
\rightarrowtail	\looparrowright	
\leftrightharpoons	\curvearrowright	
\circlearrowright	\Rsh	
\downdownarrows	\rightsquigarrow	
\downharpoonright		

Tabla 15: Flechas de negación AMS

 $\nleftarrow \quad \nrightarrow \quad \nLeftarrow \quad \nRightarrow$
 $\nrightarrow \quad \nLeftarrow \quad \nleftrightharpoons \quad \nleftrightarrow$

Tabla 16: Letras griegas AMS

 $\digamma \quad \varkappa$

Tabla 17: Letras hebreas AMS

$\beth \ \daleth \ \gimel$

Tabla 18: Símbolos varios AMS

\hbar	<code>\hbar</code>	\hslash	<code>\hslash</code>
\square	<code>\square</code>	\lozenge	<code>\lozenge</code>
\measuredangle	<code>\measuredangle</code>	\nexists	<code>\nexists</code>
\Game	<code>\Game</code>	\Bbbk	<code>\Bbbk</code>
\blacktriangle	<code>\blacktriangle</code>	\blacktriangledown	<code>\blacktriangledown</code>
\bigstar	<code>\bigstar</code>	\sphericalangle	<code>\sphericalangle</code>
\diagup	<code>\diagup</code>	\diagdown	<code>\diagdown</code>
\vartriangle	<code>\vartriangle</code>	\triangledown	<code>\triangledown</code>
\circledS	<code>\circledS</code>	\angle	<code>\angle</code>
\mho	<code>\mho</code>	\Finv	<code>\Finv</code>
\backprime	<code>\backprime</code>	\varnothing	<code>\varnothing</code>
\blacksquare	<code>\blacksquare</code>	\blacklozenge	<code>\blacklozenge</code>
\complement	<code>\complement</code>	\eth	<code>\eth</code>

Tabla 19: Operadores binarios AMS

$+$	<code>\dotplus</code>	\setminus	<code>\smallsetminus</code>	\Cap	<code>\Cap</code>
\barwedge	<code>\barwedge</code>	\veebar	<code>\veebar</code>	\barwedge	<code>\barwedge</code>
\boxtimes	<code>\boxtimes</code>	\boxdot	<code>\boxdot</code>	\boxplus	<code>\boxplus</code>
\ltimes	<code>\ltimes</code>	\rtimes	<code>\rtimes</code>	\leftthreetimes	<code>\leftthreetimes</code>
\curlywedge	<code>\curlywedge</code>	\curlyvee	<code>\curlyvee</code>	\circleddash	<code>\circleddash</code>
\circledcirc	<code>\circledcirc</code>	\centerdot	<code>\centerdot</code>	\intercal	<code>\intercal</code>
\Cup	<code>\Cup</code>	\boxminus	<code>\boxminus</code>	\divideontimes	<code>\divideontimes</code>
\rightthreetimes	<code>\rightthreetimes</code>	\circledast	<code>\circledast</code>		

Tabla 20: Operadores de relación AMS

\leq	<code>\leqq</code>	\leqslant	<code>\leqslant</code>	\lessdot	<code>\lessdot</code>
\lessapprox	<code>\lessapprox</code>	\approxeq	<code>\approxeq</code>	\lesseqgtr	<code>\lesseqgtr</code>
\lessgtr	<code>\lessgtr</code>	\lesseqgtr	<code>\lesseqgtr</code>	\backsim	<code>\backsim</code>
\risingdotseq	<code>\risingdotseq</code>	\fallingdotseq	<code>\fallingdotseq</code>	\sqsubset	<code>\sqsubset</code>
\subseteq	<code>\subseteq</code>	\Subset	<code>\Subset</code>	\precapprox	<code>\precapprox</code>
\curlyeqprec	<code>\curlyeqprec</code>	\precsim	<code>\precsim</code>	\Vvdash	<code>\Vvdash</code>
\trianglelefteq	<code>\trianglelefteq</code>	\vDash	<code>\vDash</code>	\Bumpeq	<code>\Bumpeq</code>
\smallfrown	<code>\smallfrown</code>	\bumpeq	<code>\bumpeq</code>	\gtrsim	<code>\gtrsim</code>
\geqslant	<code>\geqslant</code>	\eqslantgtr	<code>\eqslantgtr</code>	\gtrless	<code>\gtrless</code>
\gtrdot	<code>\gtrdot</code>	\ggg	<code>\ggg</code>	\circeq	<code>\circeq</code>
\gtreqless	<code>\gtreqless</code>	\eqcirc	<code>\eqcirc</code>	\supseteqq	<code>\supseteqq</code>
\thicksim	<code>\thicksim</code>	\thickapprox	<code>\thickapprox</code>	\curlyeqsucc	<code>\curlyeqsucc</code>
\sqsupset	<code>\sqsupset</code>	\succcurlyeq	<code>\succcurlyeq</code>		

$\approx \succapprox$	$\triangleright \vartriangleright$	$\trianglerighteq \trianglerighteq$
$\shortmid \shortmid$	$\parallel \shortparallel$	$\between \between$
$\varpropto \varpropto$	$\blacktriangleleft \blacktriangleleft$	$\therefore \therefore$
$\blacktriangleright \blacktriangleright$	$\because \because$	$\lessapprox \lessapprox$
$\lll \lll$	$\doteqdot \doteqdot$	$\backsimeq \backsimeq$
$\preccurlyeq \preccurlyeq$	$\vartriangleleft \vartriangleleft$	$\smallsmile \smallsmile$
$\geqq \geqq$	$\gtrapprox \gtrapprox$	$\gtreqless \gtreqless$
$\triangleq \triangleq$	$\Supset \Supset$	$\succsim \succsim$
$\Vdash \Vdash$	$\pitchfork \pitchfork$	$\backepsilon \backepsilon$

Tabla 21: Negación de operadores de relación AMS

$\not\less \nless$	$\not\leq \nleq$	$\not\leqslant \nleqslant$
$\not\leq \lneq$	$\not\leq \lneqq$	$\not\leq \lvertneqq$
$\not\approx \lnapprox$	$\not\approx \nprec$	$\not\approx \npreceq$
$\not\approx \precnapprox$	$\not\sim \nsim$	$\not\sim \nshortmid$
$\not\vdash \nvDash$	$\not\vdash \nvDash$	$\not\triangleleft \ntriangleleft$
$\not\subseteq \nsubseteq$	$\not\subseteq \subsetneq$	$\not\subseteq \varsubsetneq$
$\not\subseteq \varsubsetneqq$	$\not> \ngtr$	$\not> \ngeq$
$\not\geq \ngeq$	$\not\geq \gneq$	$\not\geq \gneqq$
$\not\sim \gnsim$	$\not\approx \gnapprox$	$\not\sim \nsucc$
$\not\approx \nsucceq$	$\not\approx \succnsim$	$\not\approx \succnapprox$
$\not\parallel \nshortparallel$	$\not\parallel \nparallel$	$\not\parallel \nvDash$
$\not\triangleright \ntriangleright$	$\not\triangleright \ntrianglerighteq$	$\not\triangleright \nsupseteq$
$\not\supseteq \supsetneq$	$\not\supseteq \varsupsetneq$	$\not\supseteq \supsetneqq$
$\not\leq \nleqq$	$\not\sim \lnsim$	$\not\approx \precnsim$
$\not\mid \nmid$	$\not\triangleleft \ntrianglelefteq$	$\not\subseteq \subsetneqq$
$\not\geq \ngeqlant$	$\not> \gvertneqq$	$\not\approx \nsucceq$
$\not\cong \ncong$	$\not\vdash \nVDash$	$\not\supseteq \supsetneqq$
$\not\supseteq \varsupsetneqq$		

Tabla 22: Alfabetos matemáticos

Paquete requerido

ABCdef	$\mathit{\mathrm{ABCdef}}$
\textit{ABCdef}	$\mathit{\mathrm{ABCdef}}$
\textit{ABCdef}	$\mathit{\mathrm{ABCdef}}$
\mathcal{ABC}	$\mathcal{\mathrm{ABC}}$
\mathcal{ABC}	$\mathcal{\mathrm{ABC}}$
\mathfrak{ABCdef}	$\mathfrak{\mathrm{ABCdef}}$
\mathbb{ABC}	$\mathbb{\mathrm{ABC}}$
\mathscr{ABC}	$\mathscr{\mathrm{ABC}}$

euscript con la opción: mathcal
eufrak
amsfonts ó amssymb
mathrsfs

SECCIÓN 3

Subíndices y superíndices

Los superíndices ó exponentes se producen con el símbolo ^

El teorema de Fermat establece que para $n > 2$, no hay enteros x, y, z que cumplan:

$$x^n + y^n = z^n$$

se produce escribiendo:

El teorema de Fermat establece que para $n > 2$, no hay enteros x, y, z que cumplan:

$$\$\$x^ny + y^nz = z^{n+1}$$$$

Debe tenerse en cuenta que, si el superíndice tiene más de un carácter de longitud, debe utilizarse {superíndice} para agrupar el superíndice; por ejemplo:

$$$(x^m)^n=x^{m\{n\}}$ → $(x^m)^n = x^{mn}$$$

pero si tecleamos x^{mn} se obtiene x^{mn} .

También podemos tener superíndices de superíndices, agrupándolos de la siguiente manera:

Los números de la forma $2^{\{2^n\}+1}$, donde n es un número natural, se denominan números de Fermat

Los números de la forma $2^{2^n} + 1$, donde n es un número natural, se denominan números de Fermat

La forma en que los agrupamos es crítica; probando:

$$2^{2^n+1} \rightarrow 2^{2^n} + 1$$

$$\{2^2\}^n \rightarrow 2^{2^n} + 1$$

obtenemos resultados diferentes (compárese en especial el tamaño de la n).

Para producir subíndices véase el siguiente ejemplo:

La sucesión (x_n) definida por

$$\$\$x_1=1, \quad x_2=1, \quad x_n=x_{\{n-1\}}+x_{\{n-2\}}; \quad (n>2)$$$$

se llama sucesión de Fibonacci.

La sucesión (x_n) definida por

$$x_1 = 1, \quad x_2 = 1, \quad x_n = x_{n-1} + x_{n-2} \quad (n > 2)$$

se llama sucesión de Fibonacci.

(nótese como introducimos espacios con el comando \quad). Al igual que en el caso de los superíndices, se pueden obtener sub-subíndices con un agrupamiento adecuado.

Con facilidad, podemos agrupar juntos sub- y superíndices; por ejemplo: (x_n^2) y (x^2_n) producen el mismo resultado: (x_n^2) . De nuevo, ha de tenerse cuidado con el modo

de agrupamiento; compárense los siguientes casos:

$$\begin{aligned} \$x_m^n &\longrightarrow x_m^n \\ \$\{x_m\}^n &\longrightarrow x_m^n \\ \$\{x^n\}_m &\longrightarrow x^n_m \end{aligned}$$

SECCIÓN 4

Raíces

La raíz cuadrada se introduce con el comando `\sqrt{Argumento}`. Así, `\sqrt{2}` produce $\sqrt{2}$. Este comando tiene un argumento opcional, para escribir raíces cúbicas, cuartas, ó n-ésimas:

$$\sqrt[4]{5}, \sqrt[5]{4} \longrightarrow \sqrt[4]{5}, \sqrt[5]{4}$$

El tamaño del signo de raíz se ajusta automáticamente al tamaño del argumento; ésta característica permite *anidar* raíces con facilidad, por ejemplo:

La sucesión

```
$$
2\sqrt{2}, , \quad 2^2\sqrt{2-\sqrt{2}}, , \quad 2^3\sqrt{2-\sqrt{2+\sqrt{2}}}, ,
\quad 2^4\sqrt{2-\sqrt{2+\sqrt{2+\sqrt{2}}}}, , \dots
$$
converge a  $\pi$ .
```

La sucesión

$$2\sqrt{2}, \quad 2^2\sqrt{2-\sqrt{2}}, \quad 2^3\sqrt{2-\sqrt{2+\sqrt{2}}}, \quad 2^4\sqrt{2-\sqrt{2+\sqrt{2+\sqrt{2}}}}, \dots$$

converge a π .

Para obtener lo anterior, nótese como se ha hecho uso de los comandos `\,`, `\;`, abreviaturas de `\thinspace` y `\thickspace`, respectivamente. En modo matemático, también puede utilizarse `\colon` (`\medspace`), que produce un espacio intermedio. Otra alternativa, si queremos *reducir* espacios, es utilizar los comandos:

`\negthinspace` (ó su abreviatura `\!`)

`\negmedspace`

`\negthickspace`

que introducen espacios análogos, pero de longitud *negativa*.

SECCIÓN 5

Delimitadores

Llamamos delimitadores a signos de la forma (), { }, etc... Una de las capacidades más potentes del modo matemático es el ajuste automático del tamaño del delimitador al tamaño del argumento que contiene. Por ejemplo, escribiendo simplemente:

$$\left[a + \left(\frac{b}{c} \right) \right] = \frac{ac+b}{c}$$

se obtiene:

$$a + \left(\frac{b}{c} \right) = \frac{ac+b}{c}$$

Obtenemos delimitadores de tamaño adecuado utilizando `\left(... \right)`, en vez de simplemente (...). Véase la diferencia:

$$a + \left(\frac{b}{c} \right) = \frac{ac+b}{c}$$

La lista de símbolos al final del capítulo ofrece una lista de todos los delimitadores disponibles.

Un punto interesante de la pareja `\left` y `\right` es que, a pesar de que *siempre* han de ir conjuntados, no es necesario que los delimitadores a los que se aplican sean iguales (podemos abrir con paréntesis y cerrar con llaves). Incluyendo un punto, se puede incluso eliminar la apertura y el cierre; por ejemplo:

$$\left. \begin{array}{l} u_x = v_y \\ u_y = -v_x \end{array} \right\} \text{ Ecuaciones de Cauchy-Riemann}$$

se obtiene con:

```
\begin{equation*}
\left. \begin{aligned}
u_x &= v_y \\
u_y &= -v_x
\end{aligned} \right\} \quad \text{Ecuaciones de Cauchy-Riemann}
\end{equation*}
```

< Nótese cómo en el ejemplo anterior se utiliza el comando `\text{...}` para incluir un trozo de texto normal dentro de una fórmula matemática>

A veces los delimitadores producidos automáticamente con `\left` y `\right` son demasiado grandes ó pequeños. Por ejemplo:

```
\begin{equation*}
(x+y)^2 - (x-y)^2 = \left( (x+y) + (x-y) \right) \left( (x+y) - (x-y) \right) = 4xy
\end{equation*}
```

produce:

$$(x+y)^2 - (x-y)^2 = ((x+y) + (x-y))((x+y) - (x-y)) = 4xy$$

Utilizando los modificadores `\bigl` y `\bigr` en su lugar:

```
\begin{equation*}
(x+y)^2-(x-y)^2=\bigl((x+y)+(x-y)\bigr)\bigl((x+y)-(x-y)\bigr)=4xy
\end{equation*}
```

tenemos:

$$(x+y)^2 - (x-y)^2 = \bigl((x+y) + (x-y)\bigr)\bigl((x+y) - (x-y)\bigr) = 4xy$$

Existen otros modificadores de tamaño predefinido, mayores que `\bigl`, que por tamaño creciente se ordenan como: `\Bigl`, `\biggl` y `\Biggl` (con versiones análogas para "r").

SECCIÓN 6

Matrices y determinantes

Para escribir datos en forma matricial, *dentro del modo matemático* se puede utilizar el entorno `array`, que funciona de forma similar al `tabular`:

```
\begin{array}{[Posición]}{FormatoColumnas}
A11 & A12... & A1N \\
A21 & A22... & A2N \\
.... \\
\end{array}
```

Por ejemplo:

```
\[\begin{array}{crl}
x & & & & & & & & \\
& & & & & & & & \\
x+y & & & & & & & & \\
& & & & & & & & \\
x^z & & & & & & & & \\
& & & & & & & & \\
(x+y)z' & & & & & & & & \\
& & & & & & & & \\
\end{array}\]
```

produce:

x	3	$m + n^2$
$x+y$	5	$m - n$
x^z	$\sqrt{75}$	m
$(x+y)z'$	100	$1 + m$

(nótese como debemos iniciar el modo matemático antes de comenzar `array`)

Basándonos en `array`, podemos construir una matriz utilizando los delimitadores `\right` y `\left`, un determinante con `\right|` y `\left|`, etc... Un método alternativo es usar los entornos específicos `pmatrix`, `bmatrix`, `Bmatrix`, `vmatrix` y `Vmatrix`, análogos a `array`, y que respectivamente añaden automáticamente los delimitadores `()`, `[]`, `{ }`, `||` y `|||`. Además, no requieren de especificación del formato de columnas; a diferencia de `array`, éstos entornos siempre producen columnas *centradas*. Por ejemplo:

```
\begin{pmatrix}
A_1 & A_2 & A_3 & A_4 \\
B_1 & B_2 & B_3 & B_4 \\
C_1 & C_2 & C_3 & C_4 \\
D_1 & D_2 & D_3 & D_4
\end{pmatrix}
```

produce:

$$\begin{pmatrix} A_1 & A_2 & A_3 & A_4 \\ B_1 & B_2 & B_3 & B_4 \\ C_1 & C_2 & C_3 & C_4 \\ D_1 & D_2 & D_3 & D_4 \end{pmatrix}$$

Dentro de estos entornos es posible utilizar el comando:

```
\hdotsfor[Factor]{NúmeroDeColumnas}
```

que produce un línea de puntos suspensivos en la matriz que abarca tantas columnas como se especifique en NúmeroDeColumnas. El argumento (opcional) Factor escala la separación entre puntos (el valor por defecto es 1).

Para escribir matrices en modo “texto”, se utiliza el entorno `smallmatrix` (que no produce delimitadores!!!); por ejemplo:

Dado que $\begin{vmatrix} a & h & g \\ h & b & f \\ g & f & c \end{vmatrix} = 0$, la matriz $\begin{pmatrix} a & h & g \\ h & b & f \\ g & f & c \end{pmatrix}$ no es invertible.

Dado que

```
$
\left|\begin{smallmatrix}
a & h & g \\
h & b & f \\
g & f & c
\end{smallmatrix}\right| = 0,
\$,
la matriz
$,
\left(\begin{smallmatrix}
a & h & g \\
h & b & f \\
g & f & c
\end{smallmatrix}\right)
\$,
no es invertible.
```

SECCIÓN 7

Puntos suspensivos

Para introducir puntos suspensivos en modo matemático tenemos una amplia colección de comandos. El comando `\dots` produce puntos suspensivos cuya ubicación vertical obedece a determinadas reglas, según le sigan signos $+$, $-$, ó \times (puntos centrados), una coma (puntos abajo), etc... Los comandos `\ldots` → \dots y `\cdots` → \cdots producen *siempre* puntos abajo y centrados, respectivamente. Además, los comandos `\vdots` → \vdots y `\ddots` → \ddots producen puntos suspensivos verticales ó diagonales, que son útiles en la escritura de matrices.

Por otra parte, existe otra serie de comandos más especializados:

- `\dotsc` → Para puntos separados por comas
- `\dotsb` → Para puntos separados por operadores binarios ($+$, $-$, etc...)
- `\dotsm` → Para puntos separados por multiplicaciones implícitas
- `\dotsi` → Para puntos separados por signos integrales
- `\dotso` → Otros puntos suspensivos

SECCIÓN 8

Fracciones y binomios

La forma general de un fracción se obtiene con el comando:

```
\frac{numerador}{denominador}
```

Para formas binomiales, se utiliza el comando análogo:

```
\binom{numerador}{denominador}
```

para el cual se carece de barra horizontal, y que incluye paréntesis. Por ejemplo, con:

```
\begin{equation*}
1 - \binom{n}{1} \frac{1}{2} + \binom{n}{2} \frac{1}{2^2} - \dotsb \\
-\binom{n}{n-1} \frac{1}{2^{n-1}} = 0
\end{equation*}
```

se obtiene:

$$1 - \binom{n}{1} \frac{1}{2} + \binom{n}{2} \frac{1}{2^2} - \dots - \binom{n}{n-1} \frac{1}{2^{n-1}} = 0$$

SECCIÓN 9
Unos símbolos sobre otros

Podemos subrayar ó poner una línea sobre el argumento con los comandos:

`\underline{Objeto}` → Coloca una línea *bajo* Objeto

`\overline{Objeto}` → Coloca una línea *sobre* Objeto

Asimismo, `\underbrace{Objeto}_{Indice}` y `\overbrace{Objeto}^{Indice}` colocan llaves *bajo* ó *sobre* un objeto, pudiéndose incluso añadir el argumento Indice *bajo* la llave:

$$\overbrace{x + \underbrace{y + z}_{2} + w}^4$$

`\[\overbrace{x+\underbrace{y+z}_{2}+w}^4 \]`

De carácter más general son los comandos:

`\underset{Deabajo}{Objeto}` y `\overset{Encima}{Objeto}`,

que colocan los símbolos Encima y Debajo, respectivamente, encima y debajo de Objeto. El comando `\stackrel{Encima}{RelaciónBinaria}` puede utilizarse para poner argumentos encima de signos = y similares. Por último, es útil conocer el comando

`\sideset{Derecha}{Izquierda}Operador`:

$$b \prod_{j=1}^{k+1} c^d$$

`\[\sideset{_a^b}{_c^d}\prod_{j=1}^{k+1} \]`

Para colocar flechas, se dispone de la siguiente colección de comandos:

<code>\overleftarrow{Objeto}</code>	<code>\overrightarrow{Objeto}</code>
<code>\underleftarrow{Objeto}</code>	<code>\underrightarrow{Objeto}</code>
<code>\overleftrightarrow{Objeto}</code>	<code>\underleftrightarrow{Objeto}</code>
<code>\xleftarrow[Deabajo]{Encima}</code>	<code>\xrightarrow[Deabajo]{Encima}</code>

Los tres primeros comandos colocan la flecha debajo ó encima del objeto, y el último se utiliza para poner objetos encima ó debajo de una flecha (que es autoextensible, dependiendo de la longitud de los objetos que tenga encima/debajo). Por ejemplo:

`\[\underleftarrow{z+w} \neq \underrightarrow{z+q} \neq \overleftrightarrow{zw} \]`

`\[\xleftarrow[T]{a+b} \quad \xleftarrow[a+b+c+d]{T} \]`

$$\begin{array}{c} \xleftarrow[z+w]{z+q} \neq \xleftarrow[T]{zw} \\ \xleftarrow[T]{a+b} \quad \xleftarrow[a+b+c+d]{T} \end{array}$$

SECCIÓN 10

Operadores de tamaño variable: integrales, sumatorios,...

Los signos de integral y sumatorio se obtienen, respectivamente, con los comandos `\int` y `\sum`. El tamaño de éstos signos depende, al igual que ocurría con las fracciones, del modo matemático que se esté utilizando, texto ó resaltado. Por ejemplo:

Euler demostró que la serie

$$\sum_{n=1}^{\infty} \frac{1}{n^2}$$
 converge, pero además que:

$$\begin{aligned} \begin{aligned} & \text{\begin{equation*}} \\ & \quad \sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \\ & \text{\end{equation*}} \end{aligned} \end{aligned}$$

Euler demostró que la serie $\sum_{n=1}^{\infty} \frac{1}{n^2}$ converge, pero además que:

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

Se observa también que la posición de los subíndices del sumatorio cambia si estamos en modo texto (sub/superíndices a un lado) ó en modo resaltado (sub/superíndices debajo y encima). Éste comportamiento es común a casi todos los operadores de tamaño variable (\sum , \prod , \coprod , etc..) y también a funciones (como la expresión para el límite: `\lim`).

En el caso de que queramos, *para el modo texto*, cambiar la posición de los sub/superíndices (para que aparezcan abajo/arriba), podemos utilizar el comando `\limits` inmediatamente a continuación del comando de operador de tamaño variable (`\sum`, `\prod`, etc...). Por contra, si deseamos que, *en modo resaltado*, los sub/superíndices aparezcan a un lado, se debe utilizar el comando `\nolimits`; por ejemplo:

Euler demostró que la serie $\sum \limits_{n=1}^{\infty} \frac{1}{n^2}$ converge, pero además que:

$$\begin{aligned} \begin{aligned} & \text{\begin{equation*}} \\ & \quad \sum \nolimits_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \\ & \text{\end{equation*}} \end{aligned} \end{aligned}$$

Euler demostró que la serie $\sum_{n=1}^{\infty} \frac{1}{n^2}$ converge, pero además que:

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

Todo lo anterior es válido para cualquier signo de tamaño variable, excepto los de tipo integral; para éstos, tanto en modo texto como en resaltado, los límites se colocan a un lado:

```
Así,  $\lim_{x \rightarrow \infty} \int_0^x \frac{\sin x}{x} dx = \frac{\pi}{2}$  y por definición,
\begin{equation*}
\int_0^\infty \frac{\sin x}{x} dx = \frac{\pi}{2}
\end{equation*}
```

Así, $\lim_{x \rightarrow \infty} \int_0^x \frac{\sin x}{x} dx = \frac{\pi}{2}$ y por definición,

$$\int_0^\infty \frac{\sin x}{x} dx = \frac{\pi}{2}$$

Si queremos límites abajo/arriba, el comando `\limits` cambia su ubicación.

SECCIÓN 11

Varias líneas de subíndices

En caso de que se quiera colocar varias líneas de subíndices, se puede utilizar el comando `\substack`, como en el siguiente ejemplo:

$$p_k(x) = \prod_{\substack{i=1 \\ i \neq k}}^n \left(\frac{x - t_i}{t_k - t_i} \right)$$

se obtiene con:

```
\begin{equation*}
p_k(x) = \prod_{\substack{i=1 \\ i \neq k}}^n \left( \frac{x - t_i}{t_k - t_i} \right)
\end{equation*}
```

Se observa que todas las líneas de subíndices aparecen centradas. Si se quiere justificarlas a la izquierda, se debe utilizar el entorno:

```
\begin{subarray}{l}
```

Líneas

```
\end{subarray}
```

como por ejemplo:

```
\sum_{\substack{1 \leq i \leq 100 \\ i < j < 8}} P(i, j)
```

$$\sum_{\substack{1 \leq i \leq 100 \\ i < j < 8}} P(i, j)$$

SECCIÓN 12

Integrales múltiples

Para utilizar integrales múltiples, el reiterar signos de integración con `\int` resulta inadecuado, ya que el espacio entre signos no es correcto. La forma correcta es utilizar los comandos `\iint` (int. dobles), `\iiint` (int. triples) e `\iiiiint` (int. cuádruples). El comando `\idotsint` produce la abreviatura con puntos suspensivos apropiada para integrales múltiples n-dimensionales:

```
\[ \iint f(x,y) dx dy \quad \iiint f(x,y,z) dx dy dz \quad \dots \int_M dx_1 \dots dx_n ]
```

$$\iint f(x,y) dx dy \quad \iiint f(x,y,z) dx dy dz \quad \int_M dx_1 \dots dx_n$$

Finalmente, `\oint` produce el símbolo de integral cerrada \oint (véanse las tablas en "The comprehensive L^AT_EX symbol list" para versiones más complejas del símbolo integral).

SECCIÓN 13

Nombres de funciones

La forma correcta de escribir una función genérica, como por ejemplo, $f(x)$, es en itálica, la forma estándar en modo matemático. Sin embargo, existen funciones especiales como \cos , \lim , \log , etc..., que poseen un nombre específico para designarlas. Estas funciones se escriben habitualmente en fuente de tipo "roman". Existen las siguientes funciones disponibles, que se obtienen a través del comando `\NombreFunción`

```
\arccos \cos \csc \exp \ker \limsup \min \sinh
\arcsin \cosh \deg \gcd \lg \ln \Pr \sup
\arctan \cot \det \hom \lim \log \sec \tan
\arg \coth \dim \inf \liminf \max \sin \tanh
```

Además, si tenemos el paquete `babel` con la opción `spanish` cargado, se pueden utilizar los nombres castellanizados de algunas funciones:

sen	\sen	arcsen	\arcsen	tg	\tg	arctg	\arctg
-----	------	--------	---------	----	-----	-------	--------

Algunos de éstos comandos (`\lim`, `\det`, ...) funcionan de forma similar a los símbolos \sum , \prod , etc..., y al igual que éstos, puede cambiarse su comportamiento con el comando `\limits`:
 Esto es un ejemplo de fórmula tipo texto `\lim_{x \rightarrow \infty} 3x+1`
`\[\text{y esto de resaltada:} \quad \lim_{x \rightarrow \infty} 3x+1 \]`
 cambiando con el comando `limits`: `\lim\limits_{x \rightarrow \infty} 3x+1`

Esto es un ejemplo de fórmula tipo texto $\lim_{x \rightarrow \infty} 3x + 1$

y esto de resaltada: $\lim_{x \rightarrow \infty} 3x + 1$

cambiando con el comando `limits` tenemos: $\lim_{x \rightarrow \infty} 3x + 1$ (en modo texto)

SECCIÓN 14

Fuentes en modo matemático

Se ha mencionado al comienzo que las letras que son parte de una fórmula aparecen en itálica. Para seleccionar un tipo de letra diferente se tienen los siguientes comandos:

- `\mathbb{Texto}` → $\mathbb{A} \mathbb{B} \mathbb{C} \mathbb{D} \mathbb{E}$ (sólo mayúsculas)
- `\mathbf{Texto}` → $\mathbf{A} \mathbf{B} \mathbf{C} \mathbf{D} \mathbf{E}$
- `\mathcal{Texto}` → $\mathcal{A} \mathcal{B} \mathcal{C} \mathcal{D} \mathcal{E}$ (sólo mayúsculas)
- `\mathfrak{Texto}` → $\mathfrak{A} \mathfrak{B} \mathfrak{C} \mathfrak{D} \mathfrak{E}$ (mayúsculas y minúsculas)
- `\mathit{Texto}` → $A B C D E$
- `\mathnormal{Texto}` → $A B C D E$ (similar a la itálica)
- `\mathrm{Texto}` → $A B C D E$
- `\mathsf{Texto}` → $A B C D E$
- `\mathtt{Texto}` → $A B C D E$

Podemos obtener otro tipos de letras caligráficas cargando el paquete `mathrsfs`. Con éste paquete cargado, el nuevo comando `\mathscr{Texto}` produce el siguiente conjunto de letras caligráficas: $\mathcal{A} \mathcal{B} \mathcal{C} \mathcal{D} \mathcal{E}$ (sólo mayúsculas).

Si en vez de cargar el paquete `mathrsfs`, cargamos el `eucal` con la opción `mathscr` (`\usepackage[mathscr]{eucal}`), el comando `\mathscr{Texto}` produce ahora letras tipo «Euler script» en lugar de caligráficas:

$\mathcal{A} \mathcal{B} \mathcal{C} \mathcal{D} \mathcal{E}$ (también sólo para mayúsculas)

Cuando tratamos con signos matemáticos en vez de con letras, los cambios de tipo de letra anteriores no siempre funcionan adecuadamente. Ésto es especialmente inconveniente cuando se desea poner en negrita una fórmula; por ejemplo, `$\mathbf{a+b=c}$` produce: $\mathbf{a+b=c}$, con los signos + e = no resaltados. Para estas situaciones se dispone del comando:

`\boldsymbol{Objeto}`

que proporciona símbolos en negrita:

`$\boldsymbol{a+b=c}$` → $a + b = c$

(además, puede verse que este comando sirve asimismo para obtener letras en negrita itálica, ya que `\mathbf{Texto}` pone el texto en negrita romana).

Para ciertos símbolos especiales, el comando `\boldsymbol{Objeto}` puede no funcionar; por ejemplo, compárese:

`\oint f` → $\oint f$

`\boldsymbol{\oint f}` → $\oint f$

El comando `\boldsymbol` no tiene efecto para la integral cerrada `\oint`. Es éstos casos, se utiliza el comando `\pmb{Objeto}` (poor man's bold), que proporciona a cada símbolo el aspecto de estar en negrita reescribiéndolo con pequeños desplazamientos:

`\pmb{\oint f}` → $\oint f$

(con una buena lupa puede verse que la letra 'f' negrita no es idéntica con `\boldsymbol` y con `\pmb`).

SECCIÓN 15

Fórmulas a color

El paquete `color` soporta sin problemas la inclusión de color dentro de expresiones matemáticas a través del comando `\textcolor{NombreColor}{Texto}`. Éste comando puede emplearse sin problemas dentro del modo matemático. Por ejemplo:

Texto... `\textcolor{green}{\int_0^\infty f(x)d(x)} = g(x) + C` ...Texto
produce:

Texto... $\int_0^\infty f(x)d(x) = g(x) + C$...Texto

Ó también:

`$$\int_0^\infty \textcolor{blue}{f(x)d(x)} = \textcolor{red}{g(x)} + C$$`
que produce:

$$\int_0^\infty f(x)d(x) = g(x) + C$$

Apuntes de Latex

Capítulo 4: Clases de documentos y su estructura

Índice

1. Tipos de Documento	2
2. Unidades de estructura	4
3. Generación de títulos	6
4. Estilos de página	7
5. Parámetros de una página	8
5.1. El paquete geometry	9

Índice de Tablas

1. Opciones de la clase de documento	3
2. Unidades de estructura	4
3. Comandos para nombres de unidades de estructura	6

Índice de Figuras

1. Formato de página para la clase book	8
2. Formato de página para la clase article.	9

SECCIÓN 1

Tipos de Documento: Las clases `article` y `book`

El comando `\documentclass[opcion1, opcion2, ...]{NombreClase}` determina el tipo (clase) general de documento que vamos a escribir; además, opcionalmente podemos cambiar opciones específicas de formato de documento (tipo de papel, tamaño de letra, etc...). Existen dos clases de documento fundamentales en el L^AT_EX básico (ó plain L^AT_EX):

- `book` —> Escritura de libros y documentos extensos
- `article` —> Documentos más breves

Además de éstas dos clases básicas, existen otras muchas (a emplear mediante el uso de paquetes externos) adaptadas a propósitos específicos, como `beamer` (presentaciones), `a0poster` (posters en tamaño a0), etc... Por otro lado, un amplio número de editoriales publican clases de documentos propias con especificaciones adaptadas al formato de sus publicaciones. Así, por ejemplo, si se pretende escribir un manuscrito para su publicación como artículo en el “Physical Review” (editado por la Sociedad Americana de Física) podemos utilizar la clase `revtex` (disponible por defecto en TeXLive) para componer el borrador del artículo. Obtendremos así un manuscrito en el que el formato de página, la numeración de secciones, etc..., siguen las líneas de diseño de la revista.

La elección de una clase determina principalmente el aspecto general de las páginas del documento, así como su estructuración en secciones, subsecciones, etc... Las diferencias fundamentales entre las clases `book` y `article` son las siguientes:

- La clase `book` crea páginas “pares” e “impares”, con diferentes especificaciones de márgenes, mientras que la clase `article` crea páginas uniformes con el texto centrado
- La clase `book` permite la división en capítulos, secciones, subsecciones, etc..., mientras que la `article` carece de capítulos y el documento sólo se subdivide en secciones.
- La clase `article` proporciona el entorno especial
`\begin{abstract}`
Texto del abstract
`\end{abstract}`
para la inclusión de un pequeño resumen del documento tras el título y autores
- Las cabeceras y pies de página tienen un diseño más complejo en la clase `book` (en la `article`, tan sólo se indica el número de página en el pie).

Todos estos (y más) comportamientos por defecto pueden cambiarse especificando las opciones de la clase de documento en el argumento optativo (entre corchetes) del comando `\documentclass`. La Tabla 1 detalla la lista de posibles opciones, su significado, y sus valores por defecto en las clases `book` y `article`.

Otras opciones diversas son:

`portrait|landscape`

Opción	Valor por defecto		Descripción
	Book	Article	
10pt	X	X	Especifican el tamaño de texto normal para todo el documento
11pt			
12 pt			
letterpaper	X	X	Tamaño del paper a utilizar
legalpaper			
executivepaper			
a4paper			
a5paper			
b5paper			
final	X	X	Si elegimos draft (borrador), aparecen marcas negras en las líneas más anchas de lo normal (mensajes Overfull \hbox)
draft			
oneside		X	Deciden si el documento se preparará distinguiendo entre páginas “a derecha” y “a izquierda” (twoside) ó con formato uniforme para todas las páginas (oneside). Tal diferenciación afecta a márgenes, cabeceras de página, etc...
twoside	X		
onecolumn	X	X	Texto a una ó dos columnas
twocolumn			
openright	X		Para la opción twoside, openright especifica que todos los capítulos empezarán en una página “a derecha”, mientras que con “openany”, los capítulos se iniciarán en la página siguiente, independientemente de su lado.
openany			
notitlepage		X	Con titlepage, el título se sitúa en una página aparte; con notitlepage, el texto sigue inmediatamente al título.
titlepage	X		

Tabla 1: Opciones de la clase de documento

Orientación del papel

`leqno`

Por defecto, los números de ecuación se sitúan a la derecha. Esta opción permite situarlos a la izquierda

`fleqn`

Por defecto, las ecuaciones se escriben centradas. Esta opción las sitúa a la izquierda

Los ejemplos del capítulo 4 (ver archivos fuente y documentos pdf en la web de la asignatura) ilustran algunas de éstas posibilidades (páginas a una ó dos columnas, clase article vs. book, etc...). Es interesante observar, en el ejemplo 4-3, cómo funciona la opción `oneside`; reemplazándola por `twoside` puede verse que se dejan páginas en blanco para colocar el comienzo de cada capítulo, índice de contenidos ó bibliografía en páginas a la derecha.

SECCIÓN 2

Unidades de estructura

Tanto `book` como `article` permiten subdividir un documento mediante diferentes unidades de estructura, organizadas jerárquicamente. La tabla 2 muestra las distintas unidades disponibles para las clases `book` y `article`, respectivamente, así como los comandos necesarios para declararlos:

Nombre	Clase <code>article</code>	Clase <code>book</code>
Parte	<code>\part(optativa)</code>	<code>\part(optativa)</code>
Capítulo		<code>\chapter</code>
Sección	<code>\section</code>	<code>\section</code>
Subsección	<code>\subsection</code>	<code>\subsection</code>
Subsubsección	<code>\subsubsection</code>	<code>\subsubsection</code>
Parágrafo	<code>\paragraph</code>	<code>\paragraph</code>
Subparágrafo	<code>\ subparagraph</code>	<code>\ subparagraph</code>

Tabla 2: Jerarquía de las unidades de estructura según la clase de documento

La sintaxis concreta para cada uno de estos comandos es la siguiente:

`\NombreComando[TextoToc]{Título}` ó `\NombreComando*[Título]`

Donde `NombreComando` representa respectivamente `part`, `chapter`, `section`, `etc...`, y `Título` es el título que queremos darle al capítulo, sección, etc... Este título aparecerá al comienzo de la sección en particular, así como en la tabla de contenidos que genera L^AT_EX con la información obtenida de las distintas subdivisiones declaradas. La tabla de contenidos se escribe con el comando `\tableofcontents`. Es necesario compilar el documento DOS veces (según la configuración, WinEdt puede ocuparse de ésto automáticamente) para generarla correctamente; la

razón de ésto reside en que, en la primera compilación, L^AT_EX detecta las secciones y guarda la información en un fichero Documento.toc. En la segunda compilación, se utiliza tal información para construir la tabla de contenidos del documento.

El argumento *optativo* TextoToc se utiliza cuando se quiere que Título no aparezca en la tabla de contenidos, sino TextoToc en lugar de él. Ésto es útil en caso de títulos de sección largos; el título completo aparecerá solamente al comienzo de la sección, y un título abreviado en la tabla de contenidos, lo cual mejora la apariencia de ésta. Tal procedimiento puede aplicarse también a las leyendas de tablas ó figuras; el comando \caption{Título} (ver capítulo 5 de los apuntes de la asignatura) admite también la sintaxis \caption[TextoToc]{Título}. Consultar el ejemplo 4-3 que ilustra el uso de esa opción. Para incluir en el documento índices de tablas ó de figuras, podemos usar respectivamente los comandos

\listoftables ó \listoffigures

Este documento hace uso de éstas capacidades (ver primera página); nótese que todos éstos comandos pueden situarse en el lugar del documento que deseemos (normalmente al principio).

Las versiones con asterisco de los comandos de estructura se utilizan cuando se desea que la unidad no sea numerada, ni aparezca en la tabla de contenidos (por ejemplo, para escribir prefacios, u otros elementos varios de un libro).

Las diversas unidades de estructura de un documento son automáticamente numeradas de forma acorde con su jerarquía. En un documento tipo article, las secciones se numeran con el formato:

1. Titulo 2. Titulo etc...

Las subsecciones como:

1.1 Titulo 1.2 Titulo etc...

Y las sub-subsecciones como:

1.1.1 Titulo 1.1.2 Titulo etc...

(en el caso de un documento book, el formato añade además el número de capítulo, ésto es, para el capítulo 1, la primera sección es la 1.1, la segunda la 1.2, etc...)¹.

Por defecto, la numeración termina aquí, y los párrafos y subpárrafos se destacan colocando el título del párrafo en negrita dentro del mismo párrafo, como muestra el siguiente ejemplo:

```
\paragraph*{Ejemplo de párrafo}
En un lugar de la mancha, de cuyo nombre
no quiero acordarme, no ha mucho tiempo
que vivía un hidalgo de los de lanza en
astillero, adarga antigua, rocín flaco...
```

```
\subparagraph*{Ejemplo de subpárrafo}
En un lugar de la mancha, de cuyo nombre
no quiero acordarme, no ha mucho tiempo
que vivía un hidalgo de los de lanza en
astillero, adarga antigua, rocín flaco...
```

Ejemplo de párrafo En un lugar de la mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco...

Ejemplo de subpárrafo En un lugar de la mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco...

(nótese cómo, para distinguir los subpárrafos, se añade una pequeña indentación). Si se desea

¹Para cambiar el tipo de numeración, se puede consultar el Capítulo 8 de los apuntes de la asignatura (Programación en L^AT_EX); si se desea mejorar la apariencia de los encabezamientos de sección ó capítulo, puede utilizarse el paquete titlesec

que los párrafos y subpárrafos aparezcan numerados (bajo la jerarquía de las sub-subsecciones, ésto es, como 1.1.1.1, 1.1.1.2, etc...) se puede emplear el siguiente comando en el preámbulo:

```
\setcounter{secnumdepth}{4} ó \setcounter{secnumdepth}{5}
```

dependiendo de que queramos numerar sólo hasta los párrafos (secnumdepth=4) ó hasta los subpárrafos (secnumdepth=5).² La modificación de `secnumdepth` permite aumentar la profundidad de la numeración de las unidades de estructura, pero sin embargo no afecta al hecho de que éstas sean incluídas ó no en la tabla de contenidos. Si se necesita colocar párrafos y subpárrafos en la tabla de contenidos, debemos modificar también `tocdepth` (cuyo valor por defecto es también de 3):

```
\setcounter{tocdepth}{4} ó \setcounter{tocdepth}{5}
```

En un documento, la tabla de contenidos, índices de tablas ó figuras, capítulos, bibliografía, etc..., viene encabezada por un título, por defecto en inglés. Cada uno de los nombres para éstos títulos viene almacenado en un comando L^AT_EX distinto. La siguiente tabla especifica los nombres de comando, junto con sus valores por defecto:

Comando	Valor por defecto	Comando	Valor por defecto
<code>\abstractname</code>	Abstract	<code>\indexname</code>	Index
<code>\appendixname</code>	Appendix	<code>\listfigurename</code>	List of Figures
<code>\bibname</code>	Bibliography	<code>\listtablename</code>	List of Tables
<code>\chaptername</code>	Chapter	<code>\partname</code>	Part
<code>\contentsname</code>	Contents	<code>\refname</code>	References

Tabla 3: Comandos y valores por defecto (en L^AT_EX estándar inglés) para las diversas unidades de estructura

Si cargamos el paquete `babel` con la opción `spanish`, los nombres por defecto en inglés de las distintas unidades de estructura cambian automáticamente a una versión española (Table por Cuadro, Chapter por Capítulo, etc...). Podemos no obstante cambiar también tales definiciones, de la siguiente forma:

```
\renewcommand{\Comando}{NombreNuevo}
```

Así por ejemplo, `\renewcommand{\listtablename}{Lista de Tablas}` renombraría en parámetro `\listtablename`. Es importante saber que, en caso de haber cargado `babel`, debemos hacer ésta modificación **inmediatamente después** del comando `\begin{document}`, y no antes; la razón reside en que muchas de las modificaciones introducidas por `babel` son activadas al comenzar el documento, y no cuando se carga el paquete.

SECCIÓN 3

Generación de títulos

Para construir la página del título, se pueden definir una serie de elementos con los que L^AT_EX construirá la cabecera del documento:

²El valor por defecto de la variable `secnumdepth` es 3

- `\title{Título}` Título del documento
- `\author{Autor1 \and Autor2 \and ...}` Lista de autores
- `\date{Fecha}` Fecha puede ser cualquier elemento: la fecha (`\today`), dejarse vacío, o texto cualesquiera
- `\thanks{Agradecimiento}` Se puede incluir en el argumento de cualquiera de los anteriores, lo que produce una nota a pie de página con agradecimientos ó comentarios varios.

Finalmente, `\maketitle` se encarga de imprimir la página del título con todo lo especificado anteriormente. En el caso de que no nos guste el formato estándar que L^AT_EX produce, existe la alternativa de usar el entorno:

```
\begin{titlepage}
Texto diverso
\end{titlepage}
```

que produce una página de título conteniendo el `Texto diverso` que especifiquemos.

Para la clase `article` (no está disponible para `book`), tenemos además la posibilidad de introducir un pequeño resumen (`abstract`) con el entorno:

```
\begin{abstract}
Texto
\end{abstract}
```

El texto de nuestro resumen aparecerá en un párrafo centrado de anchura algo menor que la del texto principal.

SECCIÓN 4

Estilos de página

El contenido del encabezamiento y pie de una página está determinado por el estilo de página elegido. Podemos elegir entre:

- **`empty`** Cabecera y pie vacíos
- **`plain`** Cabecera vacía y pie con número de página centrado; ésta es la opción por defecto para la clase `article`
- **`headings`** La cabecera contiene el número de página (por la parte externa) y un texto determinado por la clase de documento (número y título de capítulos y secciones, por ejemplo). El pie está vacío. Esta es la opción por defecto para la clase `book`.

Estos estilos de página pueden seleccionarse con el comando `\pagestyle{Estilo}` en el preámbulo, con lo cual afectarían a todo el documento. También es posible hacer modificaciones puntuales, restringidas a sólo una página, con el comando `\thispagestyle{Estilo}`

Si se utiliza la opción `headings`, podemos tener problemas de formato en el caso de capítulos y secciones con títulos largos, que excedan la anchura de la cabecera de página. En tal caso se pueden utilizar (justo después de los comandos `\chapter` ó `\section`)³ los comandos:

```
\chaptermark{MarcaCabecera} \sectionmark{MarcaCabecera}
```

```
\subsectionmark{MarcaCabecera}
```

que cambian el texto de las cabeceras de página, empleando `MarcaCabecera` en vez del título de capítulo, sección, etc... Podemos así crear encabezamientos con títulos abreviados.

SECCIÓN 5

Parámetros de una página

Todos los parámetros que controlan la colocación de texto en la página (anchura, altura, tamaño de márgenes, etc...) son modificables. La Figura 1 ilustra la definición de tales longitudes y sus valores estándar para la clase `book` (en `a4paper`)

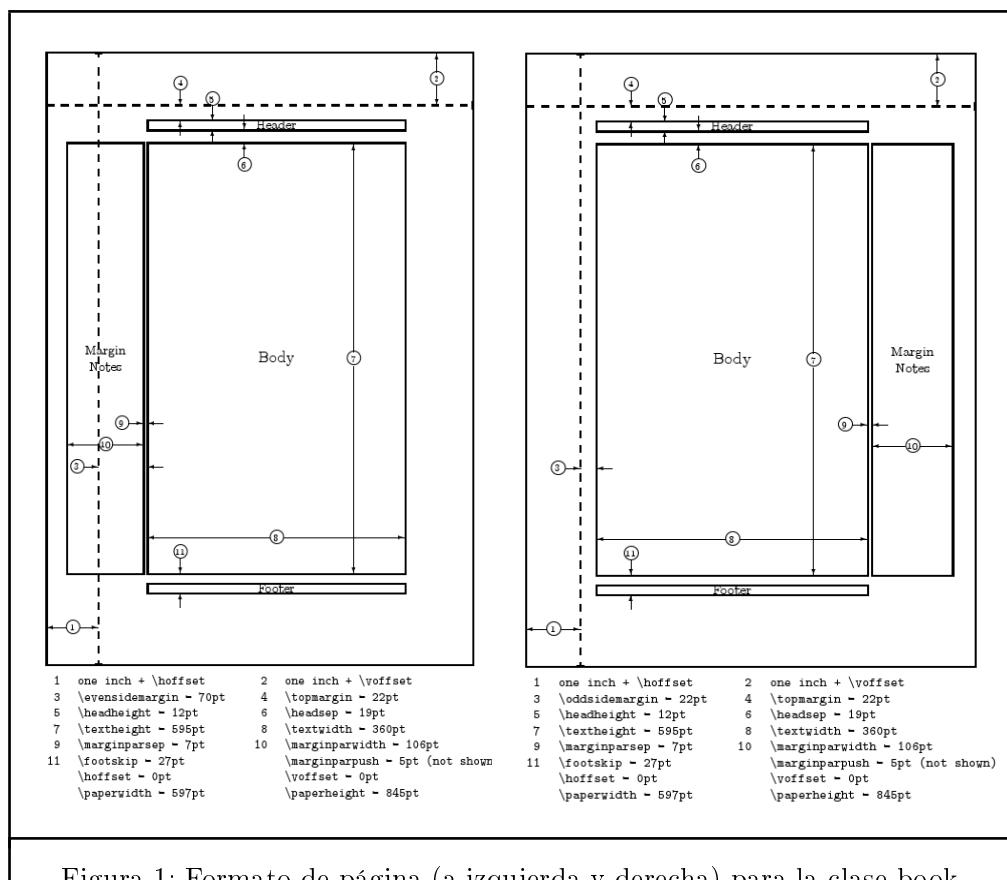


Figura 1: Formato de página (a izquierda y derecha) para la clase `book`.

³En ciertos casos especiales, debido a la forma especial en la que L^AT_EX configura las páginas, puede ser necesario repetir los comandos *antes y después* del comando de unidad de estructura +

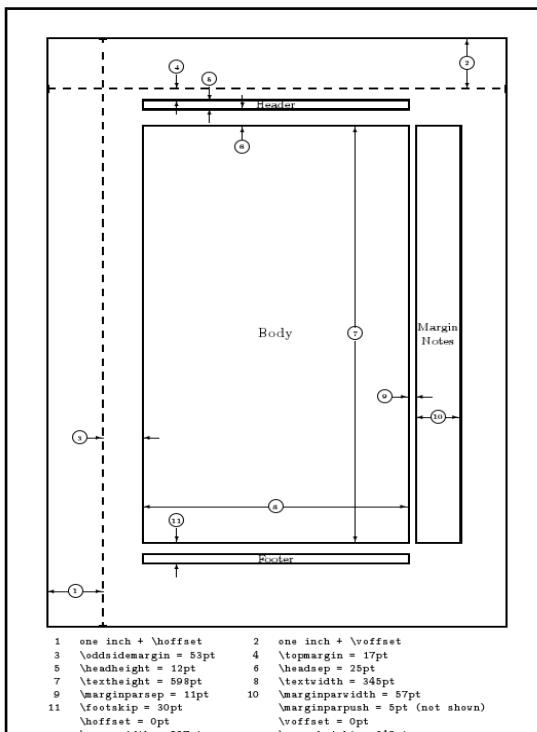


Figura 2: Formato de página para la clase article.

mientras que la Figura 2 muestra los valores estándar para `article` en `a4paper`.

Existen varios métodos equivalentes para modificarlas:

- `\Longitud=xxcm`
- `\Longitud xxcm`
- `\setlength{\Longitud}{xxcm}`

Asimismo, puede ser interesante utilizar el comando:

`\addtolength{\Longitud}{xxcm}`
que **incrementa** la longitud dada en una cierta cantidad. Por ejemplo, si queremos modificar el ancho de texto en 4 cm, pero manteniendo el texto centrado, puede usarse:

`\addtolength{\textwidth}{4cm}`
`\addtolength{\hoffset}{-2cm}`

De la misma manera, podemos variar la distancia de la cabecera al borde superior de la página (que L^AT_EX sitúa siempre a una distancia de una pulgada del borde del papel real) ajustando `\topmargin`, de la cabecera al texto con `\headsep`, etc...

Finalmente, en casos en los que nos interese incrementar ligeramente la longitud de una página dada (por ejemplo, cuando queda una sola línea huérfana en la página siguiente) se utiliza el comando:

`\enlargethispage{Longitud}`

que alarga en la cantidad `Longitud` la página.

5.1. El paquete `geometry`

El paquete `geometry` proporciona una forma simple e intuitiva de ajustar los parámetros de colocación del texto en un documento; además, nos permite variar libremente el tamaño del papel, siendo muy útil su empleo para la producción de posters (tamaño DIN-A0), pequeños folletos, etc...

El ajuste de los diferentes parámetros debe hacerse como argumentos optativos en la carga del paquete con `\usepackage{geometry}`. Así por ejemplo, con:

```
\usepackage[papersize={841mm,1189mm},lmargin=2cm,
rmargin=2cm,top=2cm,bottom=2cm]{geometry}
```

definiríamos un tamaño de papel DIN-A0 (841mm x 1189mm), y ajustaríamos los márgenes izquierdo, derecho, superior e inferior a 2cm, respectivamente (para más información acerca de las opciones del paquete, consultar su documentación).

Un ejemplo de la aplicación de éste formato tipo póster puede consultarse en el archivo “*poster.tex*” que se encuentra en la sección “Ejemplos”. Las ideas de formato y maquetación contenidas en él pueden utilizarse como base para la creación de similares plantillas de documento adaptadas a la producción de pósters con propósitos diversos. En el preámbulo, además de especificar las opciones de tamaño de papel y márgenes, se ajustan otros parámetros relacionados con el manejo de espacios, tamaño de tipos de letra, etc..., adaptados del paquete *a0poster* (no se empleó directamente éste paquete ya que sólo permite compilar documentos con *LATEX* + *dvips* + *ps2pdf*; la cabecera del ejemplo permite compilación con *PDFLATEX*).

Otro ejemplo de uso de éste paquete es la cabecera de documento presentada a continuación. Su idea es producir documentos con tamaño de papel pequeño, con especificaciones adaptadas a su visualización en lectores de libros electrónicos (e-readers). La variante del ejemplo (que además utiliza el paquete *titlesec* –ver capítulo 11 de los apuntes–) define un tamaño de papel ajustado a las dimensiones del lector Papyre con pantalla de 6 pulgadas. Dado el reducido tamaño de la pantalla, por motivos de aprovechamiento de espacio los márgenes se reducen a la mínima expresión. Un ejemplo de la apariencia final del documento es el archivo “*Apuntes3-papyre.pdf*”, descargable en el apartado “Apuntes”, donde se ha utilizado tal cabecera de documento para producir una versión visualizable en e-reader del capítulo 3 de los apuntes.⁴

```
\documentclass[10pt]{article}

%%% Carga de paquetes
\usepackage{...}
\usepackage{...}
\usepackage{...}

\parindent=3mm
\parskip=2mm

%%% Definición de cabecera con titlesec (ver Cap.11 de los apuntes)
\usepackage[calcwidth]{titlesec}
\newpagestyle{estiloA}[\large]{\headrule
\sethead{\ Sección \thesection }{\sectiontitle}{\thepage\ }%
\pagestyle{estiloA}
\renewcommand{\makeheadrule}{%
\makebox[0pt][l]{\rule[.9\baselineskip]{1.0\ linewidth}{1.0pt}}%
\rule[-.4\baselineskip]{1.0\ linewidth}{1.2pt}}}

%%% Formato de comienzo de sección con titlesec
\titleformat{\section}[frame]
```

⁴ Debido a las restricciones de espacio, en la producción de la nueva versión se debió adaptar el tamaño de tablas y otros elementos gráficos al reducido ancho de papel

```
{\normalfont}{\filcenter\small  
 \ SECCIÓN \thesection \ }  
{7pt}{\Large\bfseries\filcenter}  
  
%%% Carga de geometry con opciones ajustadas al tamaño del e-reader  
\usepackage[papersize={95mm,125mm},lmargin=1.5mm,%  
rmargin=1.5mm,top=7mm,bottom=1.5mm,headsep=3mm]{geometry}  
  
\begin{document}  
.....  
\end{document}
```

Apuntes de L^AT_EX

Capítulo 5: Inclusión de Gráficos y Elementos Flotantes

Índice de cuadros

1.	Parámetros optativos de los entornos flotantes	4
----	--	---

1. Tipos de formatos gráficos; conversión de formato

Esencialmente existen dos formas diferentes de almacenar en un fichero un gráfico: **Mapas de bits** y **Formato vectorizado**, el primero consiste en una gran tabla en la que se informa sobre el color de cada uno de los pixels (puntos) del gráfico, cuando se quiere visualizar simplemente se copia la tabla de colores en una pantalla (o impresora). Este tipo de formato es el que usan las cámaras fotográficas. Tiene el inconveniente que si queremos ampliar el gráfico esos puntos (que originalmente son del tamaño del punto más pequeño que se puede mostrar en la pantalla) acaban transformándose en cuadros de color apreciables a simple vista y por tanto la imagen pierde calidad. Las extensiones más usuales de gráficos de este tipo son bmp, jpg (jpeg), gif, png y tiff y muchos programas generan este tipo de archivos.

En el formato vectorizado en lugar de guardar la información como un mapa de colores se almacenan una serie de instrucciones que permiten al ordenador regenerar el gráfico cada vez que se quiere mostrar en pantalla (o enviar a una impresora), de esa forma cuando se amplia el tamaño el gráfico se escala adecuadamente manteniendo la calidad original. Formatos de este tipo son wmf, ps, eps, pdf . Los formatos vectorizados pueden guardar, como parte del gráfico, mapas de bits, por tanto en ese caso esa parte del gráfico sufrirá los mismos problemas comentados anteriormente.

Dependiendo del tipo de formato de los ficheros gráficos incluidos en nuestro documento, se debe utilizar una ú otra de las distintas opciones de compilación en L^AT_EX:

L^AT_EX + dvips + ps2pdf → Postscript

PDFL^AT_EX → jpg, gif, bmp, pdf

Si deseamos incluir varios archivos de diversos tipos (mapa de bits/vectorial) en el mismo documento, debemos primero convertir algunos de forma que todos finalmente se encuentren en el mismo tipo de formato.

Para la conversión de gráficos, se recomienda utilizar programas de manipulación de gráficos como CorelDraw, Gimp ó ImageMagick, que permiten convertir tanto mapas de bits a postscript como a la inversa (se recomienda Gimp por su potencia y facilidad de uso, además de ser gratuito).

2. Inclusión de gráficos: el paquete graphicx

Para la inclusión de gráficos ó fotografías, se debe cargar en el preámbulo el paquete `graphicx` (`\usepackage{graphicx}`) y utilizar el comando `\includegraphics[NombreFichero]` en el lugar donde queremos que aparezca el gráfico. Los siguientes ejemplos ilustran las diversas operaciones de transformación que podemos aplicar a un gráfico mediante la especificación de diversos parámetros optativos (a través de la sintaxis `\includegraphics[opción1,opción2,...]{NombreFichero}`):

- Especificación de anchura ó altura: Los parámetros `width` y `height` permiten ajustar la anchura ó altura a un valor determinado; si especificamos los dos, la imagen puede deformarse:

```
\includegraphics[width=2cm]{knuth.jpg}
\includegraphics[height=2cm]{knuth.jpg}
\includegraphics[width=3cm,height=2cm]
{knuth.jpg}
```



- El parámetro `scale` permite escalar globalmente las dimensiones de la imagen:

```
\includegraphics[scale=0.2]{knuth.jpg}
\includegraphics[scale=0.4]{knuth.jpg}
```



- Mediante `viewport` podemos cambiar las dimensiones de la “caja” en la que se incluye la imagen; combinado con `clip`, podemos recortar la imagen. La sintaxis es `viewport = xmin ymin xmax ymax`

```
\includegraphics[scale=0.5,%
viewport=0 0 130 155,clip]{knuth.jpg}
\includegraphics[scale=0.5,%
viewport=20 0 93 155,clip]{knuth.jpg}
\includegraphics[scale=0.7,%
viewport=20 50 93 155,clip]{knuth.jpg}
```

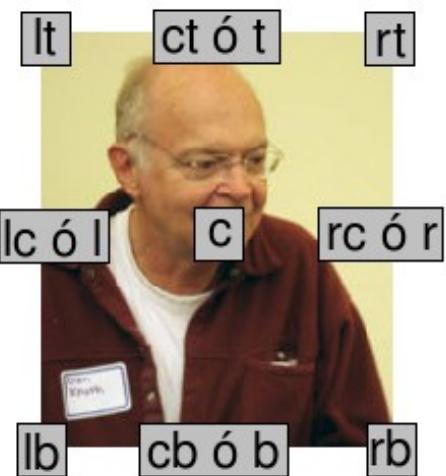


- El parámetro `trim` es análogo a `viewport`, sólo que especifica las dimensiones que deben ser recortadas a los lados izquierdo, derecho, inferior y superior, en vez de las coordenadas absolutas de la “bounding box” de la imagen

```
\includegraphics[scale=0.5,%
trim=0 30 0 30,clip]{knuth.jpg}
\includegraphics[scale=0.5,%
trim=20 50 20 0,clip]{knuth.jpg}
```



- Los parámetros `angle` y `origin` permiten rotar un gráfico un ángulo dado alrededor de un origen (`lb` por defecto) especificando a través del parámetro `origin`. La siguiente figura ilustra los diferentes valores que puede tomar el parámetro `origin` y el punto asociado sobre el que se produce la rotación. Los siguientes ejemplos describen el uso de éstos parámetros. Es importante destacar que las operaciones de rotación y escalamiento de las figuras NO SON CONMUTATIVAS; el orden en el que las especifiquemos pueden cambiar el resultado



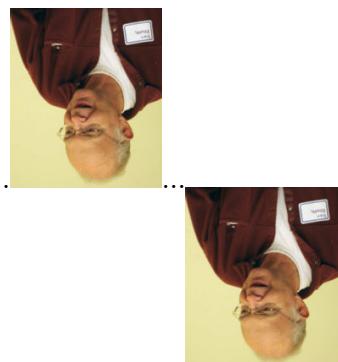
```
\includegraphics[angle=90,width=2cm]{knuth.jpg}
\includegraphics[width=2cm,angle=90]{knuth.jpg}
```



```
... \includegraphics[origin=c,
angle=45,width=2cm]{knuth.jpg}
... \includegraphics[origin=lb,
angle=45,width=2cm]{knuth.jpg} ...
\includegraphics[width=2cm,
origin=lb,angle=45]{knuth.jpg} ...
```



```
... \includegraphics[origin=c,
angle=180,width=2cm]{knuth.jpg}
... \includegraphics[origin=b,
angle=180,width=2cm]{knuth.jpg} ...
```



3. Elementos flotantes; los entornos `figure` y `table`

Para incluir un elemento de considerables dimensiones dentro de un documento, como una tabla ó una figura, L^AT_EX proporciona dos tipos de entorno:

<pre>\begin{figure} \end{figure}</pre>	<pre>\begin{table} \end{table}</pre>
--	--

ó

que permiten que, en el caso de que el compilador deba incluir los elementos dentro del entorno (típicamente, una tabla escrita con `tabular` ó un gráfico incluido con `\includegraphics`) en

Parámetro	Significado
h	Sitúa el elemento flotante <i>preferentemente</i> (es decir, si es posible) en la situación exacta donde se incluye éste
t	Sitúa el elemento en la parte de arriba de la página
b	Sitúa el elemento en la parte de abajo de la página
p	Sitúa el elemento en una página aparte dedicada sólo a elementos flotantes; en el caso del formato <code>article</code> , ésta se sitúa al final del documento, mientras que para al <code>book</code> es colocada al final de cada capítulo

Tabla 1: Parámetros optativos de los entornos flotantes

un lugar donde no existe espacio suficiente para ello (al final de una página, por ejemplo), tenga la libertad de “recolocar” ó “hacer flotar” el elemento a otra parte del documento, continuando con la escritura del texto normal; ésto permite que, en el caso crítico de una figura incluida hacia el final de una página, ésta se mueva por ejemplo hasta el principio de la página siguiente evitando el efecto antiestético de un hueco vacío.

El siguiente ejemplo muestra el comportamiento por defecto del entorno `table`; en el presente documento, se ha incluído EN ÉSTE LUGAR el siguiente código...

```
\begin{table}
\begin{center}
\begin{tabular}{|c|p{0.8\textwidth}|}
\hline
Parámetro & \multicolumn{1}{c}{Significado} \\ \hline
\texttt{h} & Sitúa el elemento flotante \emph{preferentemente} \\
(es decir, si es posible) en la situación exacta donde se incluye éste & \\
\texttt{t} & Sitúa el elemento en la parte de arriba de la página \\
\texttt{b} & Sitúa el elemento en la parte de abajo de la página \\
\texttt{p} & Sitúa el elemento en una página aparte dedicada sólo a \\
elementos flotantes; en el caso del formato \texttt{article}, \\
ésta se sitúa al final del documento, mientras que para al book es \\
colocada al final de cada capítulo \\
\hline
\end{tabular}
\end{center}
\caption{Parámetros optativos de los entornos flotantes}
\label{tabla_parametros}
\end{table}
```

...y la tabla aparece en la parte de arriba de la página. La posición preferida del elemento flotante se especifica a través de los parámetros descritos en la Tabla 1, según la sintaxis:

```
\begin{table}[parametros] ... \end{table}
```

donde podemos especificar uno ó varios parámetros según nuestras preferencias; así por ejemplo, `ht` equivale a pedir la situación en en lugar del documento donde se incluya el entorno, y, si no es posible, en la parte de arriba de la página. La opción por defecto (que tiene lugar cuando no se especifica ninguna, como en el ejemplo) es `\begin{figure}[tbp]`

Los entornos `table` y `figure` realizan funciones exactamente análogas, con la única diferencia siendo el nombre de la leyenda (figura ó tabla) que aparece bajo la tabla. Tal leyenda se incluye con el comando

```
\caption{Texto de la leyenda}
```

que produce que se imprima “Tabla N: Texto de la leyenda”¹ ó “Figura N: Texto de la leyenda” centrado bajo la tabla ó figura (que no está centrada por defecto, para ello debemos utilizar el entorno `center`, como en el ejemplo de la Tabla 1).

A lo largo de un documento L^AT_EX va enumerando las tablas y figuras, pudiendo imprimirse un índice de tablas ó de figuras a través de los comandos `\listoftables` ó `\listoffigures`, respectivamente (un ejemplo puede encontrarse en éste mismo documento).

L^AT_EX, además de enumerar automáticamente las tablas ó figuras, permite establecer referencias cruzadas a ellas (ver Capítulo 6 de los apuntes). El comando `\label{tabla_parametros}` tras el comando `caption` permite etiquetar la tabla (con el nombre `tabla_parametros`, por ejemplo) para más adelante referenciarla en el texto como

```
a través de los parámetros descritos en la Tabla \ref{tabla_parametros}, ...
```

lo cual produce, tras la compilación, el resultado “Tabla 1”. Es importante acostumbrarse a aprovechar la potencia de L^AT_EX para manejar referencias cruzadas empleando los comandos `\label` ... `\ref` para las citas a tablas/figuras en el texto, debido a que, cualquier revisión posterior del documento (añadiendo figuras ó tablas extra) mantiene correctamente la numeración de las mismas al referenciarlas.

4. Rotando y escalando texto

El paquete `graphicx` incluye algunos comandos que permiten escalar y rotar *cualquier objeto* L^AT_EX

- `\scalebox{escala horizontal}[escala vert]{argumento}`
- `\resizebox{ancho}{alto}{argumento}`
- `\rotatebox[opciones]{ángulo}{argumento}`

Ejemplos:

```
\scalebox{4}[4]{pepito}  
\scalebox{4}{pepito}  
\scalebox{-4}[4]{pepito}
```

producen diversos escalamientos de la palabra “pepito” (nótese como un cambio de signo produce una imagen reflejada):

Con `\resizebox`, en vez de aplicar un factor de escala especificamos dimensiones horizontales y verticales del objeto:

```
\resizebox{3cm}{2cm}{pepito}
```

¹Cuando se utiliza la opción `spanish` de `babel`, se traduce “Table” por “Cuadro”; si deseamos cambiar el nombre a “Tabla” debemos emplear el comando: `\renewcommand{tablename}{Tabla}` situándolo *justo a continuación* de `\begin{document}`

genera



Podemos asimismo generar rotaciones sobre un objeto mediante el comando:

```
\rotatebox[origin=X]{angulo}{Objeto}
```

donde la variable X tiene el mismo significado (origen de rotación) y toma los mismos valores que la variable opcional `origin` del comando `\includegraphics` (ver sección 2), es decir, `c,t,b,lc,lr`, etc... Por ejemplo:

En este ejemplo `\rotatebox[origin=c]{33}{PATATA}` esta rotado 33 grados respecto al centro

produce:

En este ejemplo  esta rotado 33 grados respecto al centro

```
\rotatebox{90}{\Large \ \ \ Meses \ \ } \
\begin{tabular}[b]{|c|c|c|} \
\cline{2-3}
\multicolumn{1}{c}{} & \multicolumn{2}{c}{Producción} \\ \
& Fabrica 1 & Fabrica 2 \\ \
Enero & 5.5 & 6.7 \\
Febrero & 5.2 & 5.8 \\
Marzo & 5.0 & 4.3 \\
Abril & 6.4 & 7.1 \\
\end{tabular}
```

produce:

Meses	Producción	
	Fabrica 1	Fabrica 2
Enero	5.5	6.7
Febrero	5.2	5.8
Marzo	5.0	4.3
Abril	6.4	7.1

Apuntes de L^AT_EX

Capítulo 6: Referencias, Bibliografía, Notas al Pie y al Margen

1. Referencias cruzadas

En L^AT_EX podemos referenciar cualquier objeto numerado, que hayamos etiquetado convenientemente con anterioridad. Los comandos para ello son:

\label{etiqueta} → Coloca una etiqueta en un punto dado del texto
\ref{etiqueta} → Referencia a la etiqueta colocada con el comando \label

donde el comando \label puede utilizarse dentro de cualquier unidad de estructura capítulo, sección, ...) ó entorno numerado (figura, tabla, elemento de lista, ...). El comando \ref imprime el **número** de la unidad de estructura guardado en **etiqueta**.

Por ejemplo, incluimos aquí una figura con el comando `includegraphics` (dentro de un entorno `figure`), colocamos el comando `\label{knuth}` dentro del entorno `figure`, y lo referenciamos con:

Referencia a la figura `\ref{knuth}`, lo que produce:

Referencia a la Figura 1.

De la misma manera, podemos decir que estamos en la **sección 1** incluyendo tras el comando `\section` una etiqueta:

`\label{refcruz}`

y referenciándola con

`\ref{refcruz}`



Figura 1: D. E. Knuth,
el creador de TeX

Podemos también incluir `\label{etiqueta}` utilizar el comando en una página cualquiera, y referencia ese número de página con el comando:

`\pageref{etiqueta}`

Por último, podemos también etiquetar las ecuaciones con `\label{etiqueta}`, para citarlas más adelante con `\eqref{etiqueta}`; por ejemplo:

Sea la ecuación:

```
\begin{equation}
ax + by + c = 0 \label{prueba_ecuacion}
\end{equation}
```

que es referenciada como ecuación `\ref{prueba_ecuacion}`

produce:

Sea la ecuación:

$$ax + by + c = 0 \quad (1)$$

que es referenciada como ecuación 1

Es de capital importancia, a la hora de compilar documentos con referencias cruzadas, hacerlo DOS veces (una para generarlas, y otra para que el compilador las lea de un fichero auxiliar).

2. Bibliografía

Para incluir referencias de bibliografía en un documento se utiliza el entorno `thebibliography`, con la siguiente sintaxis:

```
\begin{thebibliography}{99} --> para poder incluir hasta 999 citas
\bibitem{ref1} Titulo, autor, año, etc...
\bibitem{ref2} Titulo, autor, año, etc...
.....
\end{thebibliography}
```

Para citar referencias en el texto del documento, se utiliza el comando `\cite{etiqueta}`, donde `etiqueta` se debe corresponder con la etiqueta de la entrada `bibitem` que queremos referenciar.

3. Notas al pie y al margen

El comando:

```
\footnote[Texto de la nota al pie]
```

permite incluir “`Texto de la nota al pie`” como nota al pie en un documento.

Asimismo, el comando:

```
\marginpar[TextoIzqda]{TextoDerecha}
```

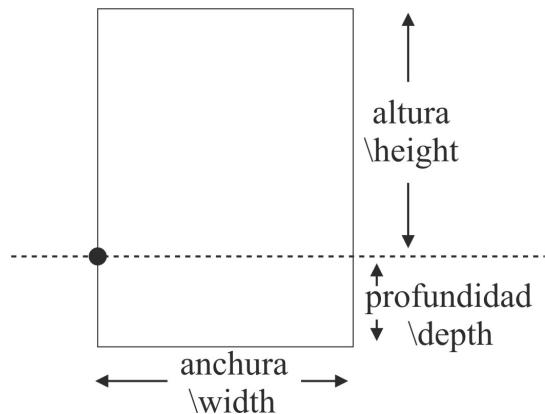
crea una nota al margen con el texto dado dentro del argumento del comando. El primer argumento, *optativo*, se utiliza en caso de que queramos incluir notas al margen colocadas a la izquierda, en vez de a la derecha, que es la opción por defecto (si estamos utilizando la opción de documento `oneside`; en caso de tener `twoside`, las notas al margen se colocan por defecto en el margen *externo*).

Apuntes de L^AT_EX

Capítulo 7: Manipulación de Cajas; Escritura a Varias Columnas

1. Cajas y marcos

El modo de trabajo de L^AT_EX se basa en cajas, que compone de modo igual al cajista de una imprenta; cada carácter es una caja, con la cual se construyen cajas más grandes (palabras), líneas, etc... Todas las cajas se alinean respecto a un punto de referencia (la línea base).



Las cajas se caracterizan por tres elementos (longitudes): altura sobre la línea base (`\height`), profundidad (`\depth`) (por ejemplo, la longitud del rabillo de la letra "p") y anchura (`\width`). La suma de altura y profundidad se denomina `\totalheight`. En general, podemos distinguir entre tres tipos de cajas:

L-R: (left-right) Cajas simples que se escriben de izquierda a derecha.

Par: Cajas de varias líneas, de anchura controlable

Rule: Línea gruesa ó delgada que se puede utilizar para separar elementos

1.1. Cajas L-R

Disponemos de varios comandos para crear cajas de ésta clase, cuya utilidad reside en la posibilidad de tratar a las cajas construidas como **objetos rígidos** (es decir, que *no se rompen por saltos de línea ó párrafo*) que podemos mover arriba, abajo, ó a ambos lados:

- Cajas sin marco:
 - `\mbox{Material}` Versión abreviada del comando siguiente; crea una caja que contiene a `Material`, de dimensiones ajustadas alas dimensiones propias de `Material`.
 - `\makebox[Ancho][Posición]{Material}` Extensión del comando anterior, donde los argumentos optativos `Ancho` y `Posición` denotan, respectivamente, el ancho de la caja y la posición de `Material` dentro de ella; éste último puede tomar los valores

`l`, `r`, `c`, `s`, correspondiendo a `left`, `right`, `center`, y `stretched` (estirado). Con la opción `s`, los elementos de `Material` se separan lo más posible, hasta agotar el ancho de la caja.

- Cajas enmarcados:

- `\fbox[Material]` Análogo a `\mbox`, produciendo una caja enmarcada.
- `\framebox[Ancho][Posición]{Material}` Versión enmarcada de `\makebox`.
- `\frame[Material]` Produce una caja que enmarca `Material` con una separación nula respecto a `Material`, y cuyo punto de referencia es la línea base (a diferencia de `\framebox`; véanse los ejemplos).

Ejemplo:

```
Colocamos la palabra
\fbox[2\width]{hola}
en el centro de un marco de ancho
el doble de dicha palabra. También,
con \makebox, podemos prescindir del
marco: \makebox[2\width]{hola}.
Ahora colocamos
\fbox[2\width][r]{hola} en
el mismo marco pero a la derecha.
Utilizando la opción ‘‘s’’, las
palabras se separan:
\fbox[2\width][s]{se separan}.
El siguiente ejemplo ilustra la
diferencia entre \fbox:
\fbox{caja} y \frame: \frame{caja}
```

Colocamos la palabra `hola` en el centro de un marco de ancho el doble de dicha palabra. También, con `\makebox`, podemos prescindir del marco: `hola`. Ahora colocamos `hola` en el mismo marco pero a la derecha. Utilizando la opción “`s`”, las palabras se separan: `[se separan]`. El siguiente ejemplo ilustra la diferencia entre `\fbox`: `[caja]` y `\frame`: `[caja]`

El grosor de la raya para el marco de la caja y la separación entre el marco y el objeto enmarcado se controlan a través de las longitudes `\fboxrule` y `\fboxsep` (por defecto, 0.4 y 3pt, respectivamente. Por ejemplo:

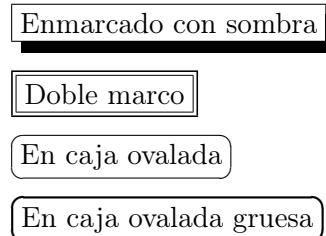
```
\fbox{hola} \
{\fboxrule=1pt \fboxsep=6pt
\fbox{hola}} \
{\fboxrule=2pt \fboxsep=1pt
\fbox{hola}}
\\[4mm]
Ahora producimos cajas dobles:
\\[4mm]
\fbox{\fbox{hola}} \
{\fboxrule=2pt \fbox{\fboxrule=1pt
\fbox{hola}}}
```

Ahora producimos cajas dobles:

El paquete `fancybox` introduce comando similares al comando `\fbox[Material]`:

```
\shadowbox{xxx}      \doublebox{xxx}      \ovalbox{xxx}      \ovalbox{xxx}
que producen cajas de apariencia diversa:
```

```
\shadowbox{Enmarcado con sombra} \\
\doublebox{Doble marco} \\
\ovalbox{En caja ovalada} \\
\Ovalbox{En caja ovalada gruesa}
```



Para todos estos comandos, podemos utilizar asimismo `\fboxsep`. También, para controlar el ancho de sombra en `\shadowbox`, podemos modificar la longitud `\shadowsize` (4pt por defecto).

Ejercicio 1:

Caja shadowbox con distancia a marco de 5mm, sombra de 3mm y grosor de marco de 1mm

1.2. Cajas tipo párrafo (Par)

Los comandos anteriores permiten escribir texto cortos (de no más de una línea). Para textos de más de una línea ó párrafo, existen dos herramientas que permiten crear cajas a modo de “pequeñas páginas” dentro de la página ambiente:

```
\parbox[Posición][Alto][PosRel]{Ancho}{Material} y
\begin{minipage}[Posición][Alto][PosRel]{Ancho}
Material
\end{minipage}
```

siendo ambos análogos, la única diferencia es que `\parbox` toma la forma de *comando* y *minipage* la de un entorno. El significado de los argumentos es el siguiente:

Ancho Parámetro **obligatorio** que establece la anchura de la caja

Material Lo que se desea incluir en la caja (texto ó cualquier otro elemento)

Posición Parámetro **optativo** para posicionar la caja con respecto a la línea base. Se elige entre `t`, `c`, `b` (`c` por defecto)

Alto Parámetro **optativo** para modificar la altura de la caja con respecto a su altura natural (`\height`)

PosRel Parámetro **optativo** para modificar la posición relativa de **Material** dentro de la caja (en caso de que ésta sea más alta); se elige entre `t`, `b`, `c`

Por ejemplo:

```
\noindent ... texto \fbox{%
\begin{minipage}[b][1.5\height]{%
[t]{0.25\textwidth} texto incluido dentro de una caja construida con el entorno minipage. Nótese como por defecto \parindent es 0pt dentro de las minipage%
\end{minipage}}%
y aqui continua el texto normal
```

texto incluido dentro de una caja construida con el entorno minipage. Nótese como por defecto \parindent es 0pt dentro de las minipage

... texto y aqui continua el texto normal

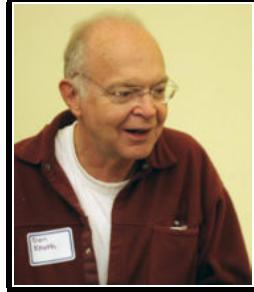
(nótese, en el ejemplo anterior, cómo podemos enmarcar la minipágina con \fbox)

Ejercicio 2:

Colocamos una foto (knuth.jpg), con anchura 0.2 veces textwidth en paralelo a un párrafo de anchura 0.25 veces textwidth (para que todo quede proporcionado).



Colocamos una foto (knuth.jpg), con anchura 0.2 veces textwidth en paralelo a un párrafo de anchura 0.25 veces textwidth (para que todo quede proporcionado).



1.3. Rayas (rule)

Se pueden imprimir cajas rellenas de tinta, de cualquier grosor y longitud con el comando:

```
\rule[Elevación]{Ancho}{Alto}
```

donde **Ancho** y **Alto** definen la anchura y altura de la caja; **Elevación** es un parámetro opcional que permite desplazar verticalmente la raya. Si es positivo/negativo, se desplaza hacia arriba/abajo, respectivamente. Por ejemplo:

```
Una raya gruesa \rule{1cm}{2pt}
que sube \rule[5pt]{1cm}{2pt}
\par
Una raya normal \rule{1cm}{0.5pt}
que baja \rule[-5pt]{1cm}{0.5pt}
```

Una raya gruesa _____ que sube _____
Una raya normal _____ que baja _____

El siguiente “truco” para el manejo de cajas merece ser estudiado con detalle. Podemos superponer dos rayas metiendo una de ellas en una caja de anchura nula, con el material de la caja justificado a la izquierda. Ésta sería la forma de conseguir que el “cursor” de L^AT_EX no se desplace, permitiéndonos sobreescribir texto.

```
Raya \makebox[0pt][l]{\rule[3pt]{1cm}{1pt}\rule{1cm}{1pt}} doble
```

produce: Raya ===== doble

Ejercicio 3:

Raya triple en medio de texto  Raya triple en medio de texto

2. Rellenando espacios entre cajas

Para separar cajas, podemos utilizar los siguientes comandos, que son útiles a la hora de calcular automáticamente la separación entre objetos (`\hspace`, no es muy satisfactorio, ya que en principio desconocemos los espacios libres):

- `\hfill` Introduce *horizontalmente* espacio vacío entre dos objetos hasta que aparezcan separados entre sí lo máximo que permita la anchura de la caja que los contiene (la anchura de la página, si no se ha especificado ninguna). Puede ser utilizado repetidamente entre parejas de objetos, obteniéndose entonces una separación uniforme entre los mismos.
- `\rulefill` Análogo a `\hfill`, salvo que rellena con una raya horizontal el espacio entre objetos.
- `\dotfill` Análogo a `\hfill`, rellena espacio con una línea de puntos
- `\vfill` Análogo *vertical* del comando `\hfill`; introduce espacio vacío *verticalmente* hasta separarlos lo más posible dentro de la caja.

Puede ser de utilidad para emplear éstos comandos el comando `\null`, que introduce una marca para calcular espacios sin introducir texto.

Ejemplo:

```

ppp \hfill ppp \hfill ppp
\fill
ppp \dotfill ppp
\dotfill ppp \dotfill ppp
\vfill
ppp \rulefill ppp \rulefill ppp
\vfill
\null \hfill ppp \hfill ppp
\hfill ppp \hfill \null

```

ppp	ppp	ppp
pppp pppp pppp pppp		
pppp _____ pppp _____ pppp		
ppp	ppp	ppp

3. Manipulación de cajas

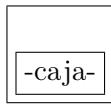
Para mover cajas en sentido horizontal, es apropiado utilizar el comando
`\hspace*{Desplazamiento}`

(es importante utilizar el asterisco, ya que la versión sin asterisco no funciona al comienzo de una línea). Para desplazar cajas en sentido vertical, respecto a la línea base, se puede utilizar:
`\raisebox{Elevación}{[Alto]}{[Profundidad]}{[Material]}`

donde **Material** representa lo que vamos a mover, **Alto** y **Profundidad** son la altura y profundidad de la caja, y **Elevación** la longitud que se desplaza verticalmente la caja.

Ejemplo:

```
Subimos \raisebox{1ex}{un poco}
un texto\\ Bajamos
\raisebox{-1ex}{un poco}
un texto\\ Bajamos un poco una
\raisebox{-1ex}{\fbox{caja enmarcada}}\\
Modificamos el alto y profundidad
\fbox{\raisebox{-1ex}[20pt][10pt]%
\fbox{-caja-}} de una caja enmarcada.
Vemos como se modifica la interlinea
```

Subimos un poco un texto
Bajamos un poco un texto
Bajamos un poco una **caja enmarcada**
Modificamos el alto y profundidad

de una caja enmarcada. Vemos
como se modifica la interlinea

4. Guardando y reutilizando cajas

Si una caja va a ser utilizada repetidamente, L^AT_EX nos proporciona un m para almacenarla y posteriormente utilizarla cuantas veces deseemos. Se comienza declarando el nombre de la caja con:

```
\newsavebox{\NombreCaja}
```

A continuacin, se define el contenido de la caja, utilizando cualquiera de los siguientes comandos:

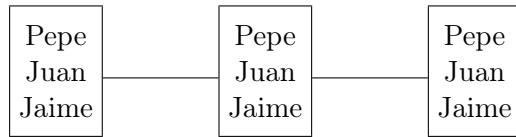
- `\sbox{\NombreCaja}{Material}`
- `\savebox{\NombreCaja}[Ancho][Posicin]{Material}`
- `\begin{lrbox}{\NombreCaja}
 Material
\end{lrbox}`

los cuales, son, respectivamente, adaptaciones de los comandos `\mbox`, `\makebox` y del entorno `minipage`. Finalmente, para recuperar la caja predefinida e imprimirla, se utiliza el comando:

```
\usebox{\NombreCaja}
```

El ejemplo siguiente ilustra todo el procedimiento:

```
\newsavebox{\Caja}
\begin{lrbox}{\Caja}
\fbox{\parbox[c][1.5cm][c]{1cm}{%
\begin{center}
Pepe \\ Juan \\ Jaime
\end{center}}}
\end{lrbox}
\usebox{\Caja}\hrulefill
\usebox{\Caja}\hrulefill
\usebox{\Caja}
```



5. Escribiendo a varias columnas

En las clases de documento `book` y `article` se escribe por defecto a una columna. Si queremos escribir a dos columnas, podemos incluir la opción `twocolumn` en los argumentos optativos de la clase de documento:

```
\documentclass[11pt,a4paper,twocolumn]{article}
```

lo cual produce que, por defecto, *todo el documento* se imprima a dos columnas.

También podemos cambiar, dentro del documento, de una a dos columnas y viceversa, con los comandos:

```
\twocolumn      \onecolumn
```

Sin embargo, el resultado no es muy satisfactorio, ya que, al cambiar de formato con ellos, se salta de página, sin dejar completa la anterior. Para mezclar estilos de forma más elegante se puede utilizar el paquete `multicol`, que se explica a continuación. Llegados a éste punto, es importante remarcar la diferencia entre los entornos `figure` y `figure*` (ó `table` y `table*`). En el primero de los casos (sin asterisco) la figura ó tabla se incluye ocupando una sola de las dos columnas, mientras que la versión con asterisco hace que la figura ó tabla ocupe las dos columnas. Esto es conveniente en el caso de tablas ó figuras demasiado anchas¹.

6. El paquete `multicol`

El paquete `multicol` (no olvidar cargar en el preámbulo para su uso) permite escribir textos hasta en 10 columnas en la misma página. Una vez cargado el paquete, podemos escribir a varias columnas abriendo el entorno `multicols` (fijarse en la ‘s’ final del nombre):

```
\begin{multicols}{Numero}[Cabecera][Anchura]  
Texto  
\end{multicols}
```

donde `Numero` indica el numero de columnas que se deseja, y el argumento opcional `Cabecera` permite poner una cabecera común al texto en multicolumna. También se puede añadir otro argumento opcional, `Anchura`, que especifica la altura mínima que debe quedar hasta el final de una página para poder comenzar el entorno. Si el espacio libre es menor que esa cantidad, se iniciará una nueva página antes de empezar a escribir a varias columnas.

Ejemplo:

```
\begin{multicols}{3}[Fragmento del Quijote]  
En un lugar de la Mancha...  
\end{multicols}
```

Fragmento del Quijote

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lan-

za en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches,

duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes

¹En el caso de escritura a una columna, no hay diferencia entre versiones con ó sin asterisco

de su hacienda. El resto della concluían sayo de velarte, calzas de velludo para las fiestas con sus pantuflas de lo mismo, los días de entre semana se honraba con su vellori de lo más fino. Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte, y un mozo de cam-

po y plaza, que así ensillaba el rocín como tomaba la podadera. Frisaba la edad de nuestro hidalgo con los cincuenta años, era de compleción recia, seco de carnes, enjuto de rostro; gran madrugador y amigo de la caza. Quieren decir que tenía el sobrenombre de Quijada o Quesada (que en esto hay

alguna diferencia en los autores que desde caso escriben), aunque por conjetas verosímiles se deja entender que se llama Quijana; pero esto importa poco a nuestro cuento; basta que en la narración dél no se salga un punto de la verdad.

Podemos personalizar el entorno `multicols` cambiando los valores de las longitudes:

- `\columnsep` -> separación entre columnas
- `\columnseprule` -> grosor de la línea que separa las columnas (por defecto, 0pt, es decir, invisible)
- `\multicolsep` -> espacio libre que se deja antes y despues de iniciar el entorno `multicols`, y que lo separa del texto circundante.

Finalmente, el comando `\columnbreak` dentro del entorno `multicols` inicia una nueva columna sin terminar la anterior.

Ejercicio 4:

Usando como ayuda el texto fuente (faltan eñes y acentos para evitar problemas de compatibilidad de codificación al cortar y pegar el texto del PDF a WinEdt):

WASHINGTON.- La aceleracion hacia la guerra ha cruzado el punto de no retorno. El presidente de EEUU ha lanzado esta madrugada un claro y definitivo ultimatum: "Sadam Husein y sus hijos deben abandonar Irak en 48 horas". Si el lider iraqui no toma el camino del exilio en ese plazo, el ataque seria inevitable e inminente. Durante un discurso de 15 minutos televisado a la nacion, George W. Bush manifesto su decepcion con el funcionamiento de Naciones Unidas, en cuyo Consejo de Seguridad no logro sacar adelante una nueva resolucion para autorizar el uso de la fuerza contra Irak...

7,6 millones de hogares españoles sufren para llegar a fin de mes, un 10% mas que el pasado año.

España ya no va tan bien. O al menos va algo peor que hace un año. La Encuesta de Presupuestos Familiares difundida hoy por el INE revela que 7,6 millones de hogares en España tienen problemas para llegar a fin de mes, un 10% mas que el pasado año. Los que mejor capean el temporal son los riojanos y los que peor, los murcianos.

El Parlamento palestino aprueba la figura del primer ministro exigida por EE UU.

El Parlamento palestino aprueba la figura del primer ministro exigida por EE UU. La Camara Legislativa palestina ha aprobado finalmente la enmienda a la Ley Basica, vital para incluir la figura de un primer ministro en el organigrama de la Autoridad Nacional Palestina (ANP). Asi, se allana el camino para que el unico candidato de Arafat, Abu Mazen, desempene esta funcion.

Producir el siguiente resultado:

Mi Periódico



WASHINGTON.- La aceleración hacia la guerra ha cruzado el punto de no retorno. El presidente de EEUU ha lanzado esta madrugada un claro y definitivo ultimátum: "Sadam Husein y sus hijos deben abandonar Irak en 48 horas". Si el líder iraquí no toma el camino del exilio en ese plazo, el ataque sería inevitable e inminente. Durante un discurso de 15 minutos televisado a la nación, George W. Bush manifestó su decepción con el funcionamiento de Naciones Unidas, en cuyo Consejo de Seguridad no logró sacar adelante una nueva resolución para autorizar el uso de la fuerza contra Irak...

7,6 millones de hogares españoles sufren para llegar a fin de mes, un 10% mas que el pasado año.

España ya no va tan bien. O al menos va algo peor que hace un año. La Encuesta de Presupuestos Familiares difundida hoy por el INE revela que 7,6 millones de hogares en España tienen problemas para llegar a fin de mes, un 10% más que el pasado año. Los que mejor capean el temporal son los riojanos y los que peor, los murcianos.

Anidamos aquí entornos multicol:



España ya no va tan bien. O al menos va algo peor que hace un año. La Encuesta de Presupuestos Familiares difundida hoy por el INE revela que 7,6 millones de hogares en...

España ya no va tan bien. O al menos va algo peor que hace un año. La Encuesta de Presupuestos Familiares difundida hoy por el INE revela que 7,6 millones de hogares en...

El Parlamento palestino aprueba la figura del primer ministro exigida por EE UU.

El Parlamento palestino aprueba la figura del primer ministro exigida por EE UU. La Cámara Legislativa palestina ha aprobado finalmente la enmienda a la Ley Básica, vital para incluir la figura de un primer ministro en el organigrama de la Autoridad Nacional Palestina (ANP). Así, se allana el camino para que el único candidato de Arafat, Abu Mazen, desempeñe ésta función.

El Parlamento palestino aprueba la figura del primer ministro exigida por EE UU. La Cámara Legislativa palestina ha aprobado finalmente la enmienda a la Ley Básica, vital para incluir la figura de un primer ministro en el organigrama de la Autoridad Nacional Palestina (ANP). Así, se allana el camino para que el único candidato de Arafat, Abu Mazen, desempeñe ésta función.

Apuntes de LATEX

Capítulo 8: Nociones de Programación LATEX

El compilador \TeX contiene aproximadamente 300 secuencias de control (comandos) llamadas *primitivas*. Éstas son operaciones de bajo nivel que no pueden ser descompuestas en acciones más simples. El resto de lo que propiamente se llama \TeX , unas 600 instrucciones, son “macros”, es decir, comandos definidos a partir de las 300 primitivas, haciendo uso de las capacidades de compilador (es decir, lenguaje de programación) de \TeX . Asimismo, el procesador de textos LATEX es otro conjunto de macros construidas a partir de comandos \TeX . En éste capítulo se introducirán las herramientas básicas de programación disponibles en \TeX , útiles para definir nuevos comandos ó entornos, modificar parámetros, automatizar tareas, en definitiva personalizar nuestro documento.

1. Nuevos Comandos y Entornos

1.1. Comandos

En ésta sección describiremos cómo utilizar el comando `\newcommand` para definir nuevos comandos LATEX que puedan ayudarnos a simplificar el realizar tareas repetitivas. Repasaremos ahora el uso de este tipo de comandos, desde una perspectiva más formal. Para la definición de un nuevo comando se dispone de tres posibilidades:

```
\newcommand{\NombreComando}[NumArg]{ArgDefecto}{Definición}
\renewcommand{\NombreComando}[NumArg]{ArgDefecto}{Definición}
\providecommand{\NombreComando}[NumArg]{ArgDefecto}{Definición}
```

donde `\NombreComando` es el nombre que queremos asignar al nuevo comando, `NumArg` indica el número de argumentos que va a tener (comprendido entre 1 y 9), `ArgDefecto` es el valor por defecto de un argumento optativo (el primero de ellos), y `Definición` contiene la definición del comando, donde los distintos argumentos se denotan como `#1, #2, etc...`.

Entre estas tres versiones existen diferencias importantes. `\newcommand` se utiliza para definir *nuevos* comandos, por lo que debemos estar seguros de que el comando a definir no existe. `\renewcommand` se utiliza para *redefinir* comandos ya existentes, reescribiendo y borrando la definición anterior del comando. Finalmente, `\providecommand` define el nuevo comando *sólo en el caso de que el comando no exista*; en caso contrario la nueva definición carece de efecto.

Para cada una de estas tres posibilidades existen versiones con y sin asterisco; las versiones con asterisco (`\newcommand*{\NombreComando}[NumArg]{ArgDef}{Def}, etc...`)

no permiten que los argumentos puedan extenderse a más de un párrafo, mientras que las versiones sin asterisco (`\newcommand{\NombreComando}[NumArg]{ArgDef}{Def}`, etc...) permiten que los argumentos se extiendan a más de un párrafo.

Ejemplos:

- Imaginemos que queremos que un texto aparezca con tipo de letra sansserif e itálico; podemos entonces definir el comando `\nuevotipo`, dependiente de un parámetro (el texto a cambiar de tipo):

```
\newcommand{\nuevotipo}[1]{{\itshape\sffamily #1}}
```

tras lo cual, escribiendo `\nuevotipo{texto sansserif}` obtendríamos *texto sansserif*.

- Cambiemos ahora el ejemplo anterior; supongamos que, además, se quiere que, por defecto, el texto aparezca en tamaño `\large`, aunque ésto último sea también una opción modificable; definiríamos entonces:

```
\newcommand{\nuevotipo}[2][\large]{{#1\itshape\sffamily #2}}
```

lo cual hace que escribiendo `\nuevotipo{texto sansserif}` resulte *texto sansserif*, mientras que con `\nuevotipo[\small]{texto sansserif}` obtendríamos *texto sansserif*

- Veamos ahora otro ejemplo útil para la escritura de expresiones matemáticas; imaginemos que la expresión (x_1, x_2, \dots, x_n) aparece frecuentemente en nuestro documento. Podemos entonces definir:

```
\newcommand{\vect}{(x_1,x_2,\dots,x_n)}
```

con lo cual, cada vez que escribamos `\vect` (el nombre del nuevo comando) se imprimirá (x_1, x_2, \dots, x_n) . Todos los nuevos comandos conviene situarlos *en el preámbulo*.

- Ahora compliquemos un poco el ejemplo con la introducción de *argumentos variables*. Si por ejemplo escribimos:

```
\newcommand{\vect}[1]{{#1_1,#1_2,\dots,#1_n}}
```

(añadiendo un argumento, que se sustituye en la fórmula con "#1"), escribiendo `\vect{x}` obtendríamos (x_1, x_2, \dots, x_n) , con `\vect{a}` se tendría (a_1, a_2, \dots, a_n) , etc...

- Añadiendo más argumentos, podemos obtener construcciones más complejas, por ejemplo, definiendo:

```
\newcommand{\vect}[2]{{#1_1,#1_2,\dots,#1_{#2}}}
```

`$\vect{x}{n}$` daría como resultado (x_1, x_2, \dots, x_n) mientras que con `$\vect{a}{p}$` se obtendría (a_1, a_2, \dots, a_p) .

- Practiquemos ahora la definición de comandos con argumentos optativos, que toman un determinado valor por defecto. Por ejemplo, construyamos:

```
\newcommand{\nuevovector}[2][x]{{#1_1,#1_2,\dots,#1_{#2}}}
```

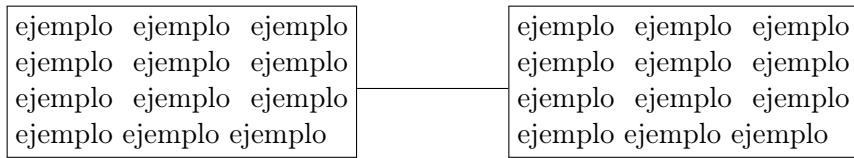
donde la "x" entre paréntesis es el valor por defecto del argumento opcional (siempre

el número 1). Entonces, escribiendo: `\nuevovector{n}` ó `\nuevovector{p}` obtendríamos (x_1, x_2, \dots, x_n) y (x_1, x_2, \dots, x_p) respectivamente, mientras que añadiendo un argumento optativo cambiaríamos su valor por defecto de “x”:
`\nuevovector[a]{n}` $\longrightarrow (a_1, a_2, \dots, a_n)$.

Ejercicio: Escribir un comando dependiente de dos argumentos que calcule la derivada parcial respecto de una función f respecto a una variable x, de forma que se puedan elegir tanto f como x:

$$\frac{\partial f}{\partial x}$$

Ejercicio: Definir un nuevo comando, dependiente de dos argumentos (párrafos de texto) que los coloque enmarcados, con anchura `0.3\textwidth` por defecto (cada párrafo), aunque modificable, unidos por una línea de longitud 2cm, y todo ello centrado;



1.2. Entornos

También es posible definir nuevos entornos, o redefinir entornos ya existentes; para ello se dispone de los siguientes comandos:

```
\newenvironment{NombreEntorno}[NumArg][ArgDef]{DefEntrada}{DefSalida}
\renewenvironment{NombreEntorno}[NumArg][ArgDef]{DefEntrada}{DefSalida}
```

que funcionan de un modo similar a los comandos del tipo `\newcommand`, en cuanto a que admiten argumentos (hasta 9), opcionalmente con el primero de ellos optativo. La diferencia reside en que en el argumento `DefEntrada` se indican las órdenes que se deben ejecutar *antes de entrar en el entorno*, y en el argumento `DefSalida` la que se deben ejecutar *al salir del entorno*. Una vez definido el nuevo entorno, se debe utilizar de la siguiente forma:

```
\begin{NuevoEntorno}{Arg1}...{ArgN}
Texto y comandos
\end{NuevoEntorno}
```

Al igual que en el caso de los comandos, existen versiones sin y con asterisco, con el mismo significado, es decir, que respectivamente admiten ó no argumentos de más de un párrafo.

Por ejemplo, construyamos un entorno que cree una minipágina de anchura variable (por defecto media página), centrada, y con el texto en negrita:

```
\newenvironment{mientorno}[1][0.5]{\begin{center}\begin{minipage}{#1\textwidth}\bfseries}{\end{minipage}\end{center}}
```

y tras definir éste nuevo entorno, tecleando:

```
\begin{mientorno}
Ejemplo de texto con una anchura estándar de media página, centrado,
y en tipo de letra negrita
\end{mientorno}
```

obtenemos:

**Ejemplo de texto con una anchura es-
tándar de media página, centrado, y en
tipo de letra negrita**

ó, si queremos emplear el argumento optativo y reducir la anchura del texto a 0.3 veces la anchura de texto (`\textwidth`):

```
\begin{mientorno}[0.3]
Ejemplo de texto con una anchura de un tercio de página, centrado, y
en tipo de letra negrita
\end{mientorno}
```

**Ejemplo de texto con
una anchura de un ter-
cio de página, centra-
do, y en tipo de letra
negrita**

Es importante tener en cuenta que los argumentos de un entorno sólo pueden utilizarse en la definición de entrada (`DefEntrada`). Si los necesitamos en la definición de salida, podemos utilizar el “truco” de guardarlos convenientemente, empleando un comando `\newcommand` para ello. En el siguiente ejemplo, creamos un entorno `cita` para escribir citas, dando el nombre del autor como argumento:

```
\newenvironment{cita}[1]{\newcommand{\autor}{#1}%
\begin{quote}\itshape‘‘}{’’\end{quote}\centerline{\autor}}
```

Tras lo cual, por ejemplo:

```
\begin{cita}{Andres Fernández}
Nuestras vidas son los ríos que van a parar al mar, que es el morir
\end{cita}
```

produce:

“ *Nuestras vidas son los ríos que van a parar al mar, que es el morir* ”

Andres Fernández

2. Compilación por trozos: `\input` e `\include`

Imaginemos que estamos escribiendo un documento largo (un libro, por ejemplo). Es conveniente, a la hora de depurar errores, escribir y compilar cada parte por separado. Para ello `LATeX` proporciona dos posibilidades:

- El comando `\input{Fichero.tex}` produce que el compilador, al encontrar esta instrucción, lee el fichero indicado en el argumento y continúa compilando dicho fichero. En el argumento del comando podemos dar, si el fichero no se encuentra en el directorio actual, el camino hasta él. Debe tenerse cuidado de que instrucciones clave como `\documentclass` ó `\begin{document}` no se dupliquen. Entonces, para escribir un libro, por ejemplo, podemos tener un documento con la siguiente estructura:

```
\documentclass[opciones]{book}
\usepackage{paquete1}
.....
\begin{document}
% \input{capitulo1.tex}
% \input{capitulo2.tex}
% \input{capitulo3.tex}
.....
\end{document}
```

y, a la hora de depurar errores, descomentar individualmente cada una de las líneas `\input{fichero.tex}`. También es posible utilizar este comando para otros usos, por ejemplo, incluir listas de instrucciones `\newcommand` y personalizaciones diversas que podamos querer hacer comunes a varios documentos.

- Una alternativa más cómoda es utilizar, en vez de `\input`, el comando `\include{Fichero}` (es esencial omitir la extensión .tex en éste caso). Entonces, en el preámbulo se puede colocar el comando `\includeonly{Fichero1,Fichero2,...}`, que hace que sólo se incluyan en la compilación los ficheros que aparecen en el argumento. Es importante mencionar que al comenzar y terminar, la orden `\include` induce un salto de página (más exactamente, un comando `\clearpage`, que además “expulsa” elementos flotantes pendientes), por lo que esta alternativa es conveniente utilizarla sólo para incluir capítulos de un libro ó tesis.

3. Conceptos básicos sobre contadores y longitudes

3.1. Contadores

En su funcionamiento habitual, L^AT_EX utiliza un amplio número de contadores con el fin de enumerar distintos elementos de un documento: páginas, secciones, tablas, figuras, etc... Cada contador tiene un *nombre* que permite identificarlo; así, `page` es el contador que identifica páginas, `chapter` capítulos, etc... En lo sucesivo, denotaremos ese nombre como *NombreContador*. Cada contador lleva asociados una serie de elementos de diferente significado: *nombre*, *valor* (siempre un número entero) y *formato*, éste último pudiendo tomar variadas formas: (I, II, III..., a, b, c...)

Se dispone de los siguientes formatos de contador:

\arabic{NombreContador}	1, 2, 3, 4...
\alph{NombreContador}	a, b, c, d... (nota 1)
\Alph{NombreContador}	A, B, C, D... (nota 1)
\roman{NombreContador}	I, II, III, IV... (nota 2)
\Roman{NombreContador}	I, II, III, IV...
\fnsymbol{NombreContador}	*, **, ***, ****... (nota 3)

Nota 1: El valor del contador no puede superar 27 (número de letras en el abecedario)

Nota 2: El resultado mostrado es el que se obtiene con babel, opción spanish. Sin ello, se obtendría i, ii, iii, ...

Nota 3: Igualmente, el resultado mostrado es el obtenido con babel y spanish; en caso contrario, se utilizan las marcas inglesas: *, †, ‡... En ambos casos, el valor no puede ser superior a 6

Asociado a cada contador existe un comando, llamado **representación del contador**, que permite imprimir el valor del contador *NombreContador* en alguno de los formatos descritos; el comando es:

\the*NombreContador*

Cuando L^AT_EX define un nuevo contador, le asigna inicialmente la representación correspondiente al formato \arabic; si queremos cambiarla, podemos redefinirla mediante el comando \renewcommand*; veamos unos ejemplos de lo que se puede hacer:

Este ejemplo muestra cómo obtener el número de la página en curso; ésta página es la número \thepage, en la representación original.\\\renewcommand*{\thepage}{\roman{page}}Ahora esta cambiada a números romanos:
Página \thepage \\Finalmente, algo más elaborado: \\\renewcommand*{\thepage}{[Sección \% \thesection \ -- Página \arabic{page}]}Estamos en: \thepage

Este ejemplo muestra cómo obtener el número de la página en curso; ésta página es la número 6, en la representación original.

Ahora esta cambiada a números romanos: Página VI
Finalmente, algo más elaborado:
Estamos en: [Sección 3 – Página 6]

Ejercicio: Cambiar la representación de las secciones en éste documento a números romanos (I, II, III...), y la de las subsecciones al formato: I-a, II-b, etc...

Podemos cambiar los valores de un contador con los siguientes comandos:

- \setcounter{NombreContador}{Valor} —> Asigna al contador *NombreContador* el valor entero *Valor*, con independencia del valor anterior
- \addtocounter{NombreContador}{Valor} —> Incrementa *NombreContador* con la cantidad *Valor*, que puede ser positiva ó negativa.

Ejemplo:

```

Esta es la sección \thesection. Pero
podemos añadirle 2 fácilmente:\\
\addtocounter{section}{2}
ahora estamos en la sección \thesection
Mejor lo dejamos como estaba, porque
si no (comprobar comentando la
línea siguiente) las restantes
secciones quedarían numeradas
incorrectamente (esto es, el efecto
de estos cambios de numeración
es \emph{global}):
\addtocounter{section}{-2}

```

Esta es la sección 3. Pero podemos añadirle 2 fácilmente:
ahora estamos en la sección 5. Mejor lo dejamos como estaba, porque si no (comprobar comentando la línea siguiente) las restantes secciones quedarían numeradas incorrectamente (esto es, el efecto de estos cambios de numeración es *global*):

Podemos recuperar el valor *numérico* de un contador, independientemente de su representación, con el comando:

```
\value{NombreContador}
```

lo cual es útil para la gestión e contadores, como veremos a continuación.

Se definen nuevos contadores con la instrucción:

```
\newcounter{NuevoContador}[ContadorExistente]
```

que introduce un contador de nombre *NuevoContador*, y le asigna cero como valor inicial. El argumento *ContadorExistente* es *optativo*, y sirve para subordinar *NuevoContador* al contador ya existente *ContadorExistente*, de la misma forma que, por ejemplo, el contador *subsection* está subordinado al contador *section*: incrementar en una unidad el contador *section* implica que el contador *subsection* se reinicia a cero automáticamente.

Veamos un ejemplo de cómo introducir un nuevo contador, con el fin de enlazar varias listas *enumerate* manteniendo la numeración (*enumi* es el contador estándar L^AT_EX para los ítems de primer nivel en entornos *enumerate*):

```

Las primeras lecciones
son las siguientes:
\begin{enumerate}
\item Números reales
\item Números complejos
\setcounter{conserva}{\value{enumi}}
\end{enumerate}
Mas adelante, se estudiarán
temas más complicados:
\begin{enumerate}
\setcounter{enumi}{\value{conserva}}
\item Continuidad
\item Derivación
\end{enumerate}

```

Las primeras lecciones son las siguientes:

1. Números reales
2. Números complejos
3. Continuidad
4. Derivación

Debe mencionarse que al crear un nuevo contador se crea automáticamente el comando *\theNuevoContador*, con la definición *\arabic{NuevoContador}* por defecto.

Cuando se modifica un contador con los comandos *\setcounter* y *\addtocounter*, los contadores subordinados no se ponen a cero; para obtener ése efecto, se dispone de los comandos:

`\stepcounter{NombreContador}` → Incrementa `NombreContador` en una unidad, y reinicia todos los contadores subordinados a él

`\refstepcounter{NombreContador}` → Lo mismo que el anterior, pero declarando también como valor del comando `\ref` el texto generado por `\theNuevoContador` cuando se utilizan referencias cruzadas con los comandos `\label` y `\ref`

Veamos un pequeño ejemplo que ilustra como utilizar estos comandos; definimos:

```
\newcounter{prg}[section]\newcounter{linea}[prg]
\newcommand*\lin{\%
\addtocounter{linea}{1}\thelinea\quad}
\renewcommand*\theprg{\arabic{section}.\arabic{prg}}
\newenvironment*{programa}{%
\refstepcounter{prg}
\begin{center}Programa~\theprg\end{center}
\ttfamily\obeylines\obeyspaces\par}
```

Ahora, como ejercicio, escribir un pequeño documento `article`, con un par de secciones y un par de entornos `programa` en cada una de ellas, utilizando dentro del entorno `programa` el comando `\lin` para ir enumerando líneas de código de un programa. Asimismo, incluir una referencia cruzada a uno de los programas (Ver archivo ejercicio2.pdf adjunto)

3.2. Longitudes

Al igual que con los contadores, L^AT_EX es también capaz de crear y modificar variables de tipo Longitud. Las longitudes que habitualmente utiliza L^AT_EX pueden tomar dos tipos de valores:

Rígidos: Toman un valor determinado; por ejemplo:

`\quad = 11.747 pt, \thinspace = 1.958 pt, \hoffset = -28.45274pt`¹

Elásticos: Toman un valor que L^AT_EX puede modificar dentro de unos límites, a fin de optimizar la composición del documento. Por ejemplo, `\bigskip`, `\medskip` y `\smallskip`

El comando `\bigskip` se define como:

```
\vspace{12pt plus 4pt minus 4pt}
```

lo cual quiere decir que L^AT_EX debe introducir un espacio vertical de 12 pt, aunque tiene la libertad de incrementarlo ó reducirlo en 4pt, según convenga a fin de distribuir el espacio de forma homogénea. Alguna de las holguras `plus` ó `minus` pueden estar ausentes en la definición, pero si ambas aparecen deben estar en ése orden.

¹Hay que tener cuidado en no confundir el concepto de *longitud* y del *valor* que toma una longitud; `\hoffset` es una longitud, mientras que `\quad` y `\thinspace` son *comandos* que dejan en blanco un espacio horizontal de valor rígido

Los comandos `\bigskip`, `\medskip` y `\smallskip`, respectivamente, tienen asociadas longitudes elásticas con valores almacenados en `\bigskipamount`, `\medskipamount` y `\smallskipamount`, por los que tales comandos se definirían de hecho como:

```
\bigskip → \vspace{\bigskipamount}
\medskip → \vspace{\medskipamount}
\smallskip → \vspace{\smallskipamount}
```

y donde cada una de éstas longitudes elásticas toma valores:

```
\bigskipamount :: 12.0pt plus 4.0pt minus 4.0pt
\medskipamount :: 6.0pt plus 2.0pt minus 2.0pt
\smallskipamount :: 3.0pt plus 1.0pt minus 1.0pt
```

Puede obtenerse el valor de cualquier longitud con el comando:

```
\the\NombreLongitud
```

donde `NombreLongitud` es el nombre de la longitud; éste comando siempre expresa las longitudes en unidades pt, con el punto como separador decimal.

Al igual que ocurría con los contadores, los valores de una longitud pueden modificarse. Existen dos comandos para ello:

- `\setlength{\NombreLongitud}{Valor}` Asigna a la longitud `\NombreLongitud` un valor igual al argumento `Valor`, que debe ser una longitud (ésto es, expresada en unidades cm, pt, etc...). Puede ser un valor tanto rígido como elástico (por ejemplo, `5mm plus 1mm minus 2mm`). También es posible que `Valor` sea una variable de longitud (`\textwidth`) con quizás un factor multiplicativo (`0.5\textwidth`, por ejemplo).

Una forma alternativa de asignar a `\NombreLongitud` un valor es utilizar la sintaxis:

```
\NombreLongitud=Valor ó bien: \NombreLongitud Valor
```

- `\addtolength{\NombreLongitud}{Valor}` Suma a la longitud `\NombreLongitud` la cantidad `Valor`, que puede ser positiva ó negativa.

Al contrario que lo que ocurría con los contadores, cuyas asignaciones tienen carácter *global* (es decir, trascienden el grupo dentro del cual han sido declaradas, y tienen efecto en todo el resto del documento), las asignaciones de longitud tienen por defecto carácter *local*; si se realizan dentro de un grupo, el valor anterior a la asignación se recupera a la salida del grupo. En el caso de que deseemos un efecto global, puede ser aconsejable realizar tales asignaciones en el preámbulo del documento.

Se pueden definir nuevas longitudes con el comando:

```
\newlength{\NuevaLongitud}
```

que crea una nueva longitud llamada `\NuevaLongitud`; es importante que `\NuevaLongitud` no sea ni un comando ni una longitud L^AT_EX ya existentes, en cuyo caso obtendríamos un mensaje de error. Por defecto, las nuevas longitudes son creadas con un valor inicial 0.0 pt.

Para la gestión de valores de longitud son útiles los siguientes comandos:

```
\settowidth{\NombreLongitud}{\Objeto}  
\settoheight{\NombreLongitud}{\Objeto}  
\settodepth{\NombreLongitud}{\Objeto}
```

que calculan, respectivamente, la anchura (width), altura (height) y profundidad (depth) de un objeto, asignando el valor resultante a la longitud `\NombreLongitud`.

Ejemplos:

Imaginemos que queremos medir la longitud asociada al comando `\quad`. Para ello podemos definir una nueva longitud:

```
\newlength{\longi}
```

a continuación, asociamos a `\longi` la anchura del espacio asociado al comando `\quad`:

```
\settowidth{\longi}{\quad}
```

tras lo cual, el comando `\the\longi` muestra el valor 10.88788pt

Ahora creamos otra longitud: `\newlength{\longitud}`

que empleamos para medir la anchura, altura, y profundidad de la palabra Pajarita

```
\noindent La anchura de la  
palabra {\Large Pajarita}  
es \settowidth{\longitud}%  
{\Large Pajarita} \the\longitud\\  
su altura es \settoheight{\longitud}%  
{\Large Pajarita} \the\longitud\\  
su profundidad \settodepth{\longitud}%  
{\Large Pajarita} \the\longitud\\
```

La anchura de la palabra Pajarita
es 48.91916pt
su altura es 9.91672pt
su profundidad 2.80008pt

```

\newlength{\longA}
\settowidth{\longA}{xxxxx}
\begin{center}
xxxxx\\
xxxxx\hspace{\longA}xxxxx\\
xxxxx\hspace{\longA}xxxxx%
\hspace{\longA}xxxxx\\
xxxxx\hspace{\longA}xxxxx\\
xxxxx
\end{center}

```

XXXXX
 XXXXX XXXXX
 XXXXX XXXXX XXXXX
 XXXXX XXXXX
 XXXXX

3.2.1. Longitudes elásticas `fil`

En este apartado describiremos dos *unidades de longitud* elásticas:

• `fil`

• `fill`

que L^AT_EX utiliza para introducir espacios de longitud variable. Ambas proporcionan dos diferentes grados de “elasticidad infinita”; `fil` es una unidad de longitud elástica infinitamente más grande que cualquier longitud rígida, mientras que `fill` es infinitamente más grande que `fil` (y por tanto, que cualquier longitud rígida).

Basados en éstas unidades de longitud, existe una variedad de comandos:

- `\fill` Es una *longitud*, de valor `0pt plus 1fill`
- `\stretch{n}` Es una *longitud* de valor `0pt` y holgura un número `n` de unidades `fill` (entero ó decimal). Así, `\fill` equivale a `\stretch{1}`

De éste modo, los comandos `\hfill` y `\vfill` equivalen a `\hspace{\fill}` y `\vspace{\fill}`, respectivamente. La utilidad del comando `\stretch{n}` está en la posibilidad de separar objetos con espacios proporcionales a diversas cantidades. Véase el siguiente ejemplo:

```

Colocamos un texto centrado:\[2mm]
\vrule\hspace{\stretch{1}}Texto
centrado\hspace{\stretch{1}}\vrule\par
Ahora colocamos un texto con
el doble de espacio a
un lado que al otro:\par
\noindent\vrule\hspace{\stretch{1}}%
Texto\hspace{\stretch{2}}\vrule\par
Otro ejemplo, con la distancia
entre T1 y T2 igual a tres veces
la distancia a los márgenes:\par
\noindent\vrule\hspace{\stretch{1}}%
T1\hspace{\stretch{3}}T2
\hspace{\stretch{1}}\vrule

```

Colocamos un texto centrado:

| Texto centrado |

Ahora colocamos un texto con el doble de espacio a un lado que al otro:

| Texto |

Otro ejemplo, con la distancia entre T1 y T2 igual a tres veces la distancia a los márgenes:

| T1 T2 |

(para imprimir la barra vertical de referencia al comienzo y final de línea en el ejemplo anterior, hemos utilizado el comando `\vrule`; podemos poner una “marca” en blanco con los comandos `\mbox{}` ó `\null`).

- `\hfill` y `\vfill` (ya descritos)

- **\hfil** y **\vfil** Análogos a los anteriores, pero empleando para la elasticidad una unidad **fil** en lugar de **fill**.

El siguiente ejemplo ilustra la diferencia entre las unidades **fil** y **fill**:

<pre>\noindent A \hfil B \hfil C \\ D \hfill E \hfill F \par</pre>	 	A B C D E F
--	--	--

¿Porqué cambian las posiciones de B y C en la primera línea? La respuesta está en que, antes de cortar una línea, L^AT_EX introduce un espacio de elasticidad variable, a fin de evitar que las líneas cortas se estiren hacia la derecha. Éste espacio se controla a través de la longitud **\parfillskip**, que por defecto tiene el valor **0pt plus 1fil**. Por tanto, en el primer ejemplo se equilibrان los espacios asociados a tres comandos **\hfil**. En el segundo caso, esto no sucede, dado que **\hfill** corresponde a un grado de elasticidad infinitamente más grande.

- **\hfilneg** y **\vfilneg** Equivalen, respectivamente, a **\hspace{0pt plus -1fil}** y a **\vspace{0pt plus -1fil}**, y permiten **cancelar** el efecto de los comandos **\hfil** y **\vfil**; por ejemplo:

<pre>\parindent=0pt \parfillskip=0pt \newcommand*\centrar[1]{% \vrule\hfil #1\hfil\vrule} \centrar{\Centrado}\par \centrar{\Centrado anulado}\hfilneg\par \centrar{\hfilneg Centrado anulado}</pre>	 Centrado Centrado anulado Centrado anulado
---	--

- **\hss** Equivale a **\hspace{0pt plus 1fil minus 1fil}**, e interviene en la definición de los comandos **\leftline**, **\rightline** y **\centerline**
- **\vss** Análogo vertical, que equivale a **\vspace{0pt plus 1fil minus 1fil}**

Los siguientes comandos (algunos de ellos ya mencionados anteriormente) tienen un efecto similar a **\hfill**, con la diferencia de que en el espacio intermedio introducen diversos símbolos de extensión variable (en dirección horizontal):

- `\hrulefill` → Raya

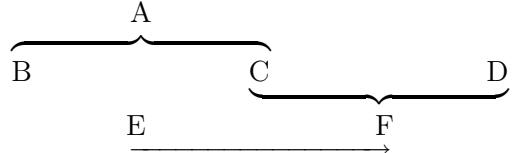
A\hrulefill B\hrulefill C | A_____B_____C

- `\dotfill` → Línea de puntos

A\dotfill B\dotfill C | A.....B.....C

- `\downbracefill` y `\upbracefill` → Llaves hacia abajo ó hacia arriba, respectivamente.
- `\leftarrowfill` y `\rightarrowfill` → Flechas a izquierda y derecha, respectivamente.

```
\parindent=0pt \parfillskip=0pt
\mbox{} \hspace{\stretch{1}} A%
\hspace{\stretch{3}} \mbox{} \[-3pt]
\mbox{} \downbracefill \mbox{} %
\hspace{\stretch{2.3}} \mbox{} \\\
B \hspace{\stretch{1}} C%
\hspace{\stretch{1}} D \[-7pt]
\mbox{} \hspace{\stretch{2.3}} %
\mbox{} \upbracefill \mbox{} \\\
\mbox{} \hspace{\stretch{1}} E%
\hspace{\stretch{2}} F%
\hspace{\stretch{1}} \[-5pt]
\mbox{} \hspace{\stretch{0.5}} %
\rightarrowfill%
\hspace{\stretch{0.5}} \mbox{} \\\
```



Finalmente, describiremos los comandos `\rlap{Objeto}` y `\llap{Objeto}`; respectivamente, colocan `Objeto` en una caja de anchura 0pt (por lo que el cursor no se mueve), con el objeto saliendo hacia la derecha ó izquierda de la caja. Por ejemplo:

Tachamos la palabra `izquierda`%
`\llap{\rule[2.5pt]{48pt}{0.4pt}}`
y seguimos escribiendo.\par
`\hfil \rlap{uno}\llap{dos}\vrule` \par
`\hfil \llap{dos}\rlap{uno}\vrule`

Tachamos la palabra `izquierda` y seguimos escribiendo.
dostuno
dostuno

Ejercicio: Diseñar un comando, dependiente de un argumento, que tache un fragmento de texto.

pepe y juan

4. Programación con \TeX

4.1. Otro modo de definir comandos

Anteriormente hemos visto cómo definir nuevos comandos mediante la utilización de los comandos tipo `\newcommand` de \LaTeX . Existe un modo alternativo, empleando

comandos de más bajo nivel de \TeX (de hecho, éste es el modo estándar de construir “macros” \LaTeX). Para ello existe el comando \def , con la siguiente sintaxis:

```
\def\NuevoComando#1...#9{Definición}
```

donde \NuevoComando es el nombre del nuevo comando, $#1\dots#9$ los argumentos de que depende (hasta 9), y entre llaves su definición. Por ejemplo, definamos:

```
\def\ecuacion#1#2{\ensuremath{\#1_1^2+\#1_2^2+\cdots+\#1_{#2}^2=1}}
```

tras lo cual, $\text{\ecuacion}{z}{5}$ produce $z_1^2 + z_2^2 + \dots + z_5^2 = 1$. Nótese el uso del comando $\text{\ensuremath{Fórmula}}$, que tiene como resultado asegurar que la expresión \Fórmula se ejecuta dentro del modo matemático (por lo cual, no es necesario abrir y cerrar signos $\$$ antes y después del comando \ecuacion).

El comando \def , a diferencia del \newcommand , permite elegir los delimitadores de los argumentos (que con \newcommand siempre deben ser llaves, ó corchetes para los argumentos optativos). Además, es incluso posible prescindir de las llaves al escribir los argumentos del comando; por ejemplo, en el caso anterior podríamos haber escrito \ecuacion z5 con el mismo resultado que $\text{\ecuacion}{z}{5}$: \TeX lee secuencialmente los argumentos tras el nombre del comando. Para elegir delimitadores especiales entre los argumentos, simplemente los incluimos entre $\#1$, $\#2\dots\#n$. Por ejemplo, redefinimos \ecuacion como:

```
\def\ecuacion#1:#2:{\ensuremath{\#1_1^2+\#1_2^2+\cdots+\#1_{#2}^2=1}}
```

lo cual indica que el primer argumento debe terminar con “;” y el segundo con “:”

ahora debemos escribir \ecuacion z;5 para obtener $z_1^2 + z_2^2 + \dots + z_5^2 = 1$

4.2. Definiciones globales

Todos los comandos creados con \newcommand , \providetcommand ó \def , (ó redefinidos con \renewcommand) son *locales*; es decir, si están definidos dentro de un grupo, su acción estará restringida a ese grupo. En el ejemplo siguiente se ve cómo la redefinición del comando \prueba dentro del entorno \itemize carece de efecto fuera de él:

<pre>\def\prueba{Prueba 1} \begin{itemize} \def\prueba{Prueba 2} \item \prueba \end{itemize} \prueba</pre>	<ul style="list-style-type: none"> ■ Prueba 2 <p>Prueba 1</p>
--	--

Si queremos definir un comando global (con efecto fuera del grupo donde es definido) se puede utilizar cualquiera de estas dos alternativas:

```
\global\def\NuevoComando#1...#9{Definición}
\gdef\NuevoComando#1...#9{Definición}
```

es decir, o bien anteponemos el comando \global a la definición, o bien usamos el comando \gdef . El comando \global también puede usarse para hacer globales otro tipo de asignaciones de tipo local, como por ejemplo las modificaciones de longitudes (\setlength y \addtolength)

Ejercicio: Reemplazar en el ejemplo anterior la definición de \prueba dentro del entorno `itemize` por una de carácter global, y observar el cambio en el resultado.

4.3. Definiciones recursivas: el comando `\edef`

Cuando se define un comando, no se ejecutan los comandos que puedan participar en la definición. Por ejemplo, si tan sólo definimos `\def\aa{\b}`, siendo `\b` un comando inexistente, la compilación no produce errores. Sólo cuando intentemos *utilizar* el nuevo comando `\aa` obtendremos un error. En ocasiones, puede interesar que al realizar la definición de un nuevo comando, se ejecuten las instrucciones que participan en la definición. Imaginemos, por ejemplo, que queremos redefinir un comando `\aa` de forma recursiva, es decir, de forma que la nueva definición de `\aa` contenga al mismo "`\aa`" en su definición:

```
\def\aa{hola} \def\aa{\aa \aa} \aa
```

las instrucciones anteriores provocan un error (comprobar). La forma de resolver el problema es redefinir el comando `\aa` con el comando `\edef`, totalmente análogo a `\def`, pero que hace que al redefinir `\aa` se expanda ó ejecute antes el comando `\aa` dentro de la definición:

```
\def\aa{hola} \edef\aa{\aa \aa} \aa
```

produce ahora: holaholahola

Si queremos que el comando `\edef` tenga efecto global, se debe utilizar el comando `\xdef`, que equivale a `\global\edef`.

4.4. El comando `\let`

Imaginemos que definimos un comando en función de otros comandos, y los comandos en los que se basa cambian. Este cambio se trasladará entonces al nuevo comando:

<pre>\noindent\def\uno{1}Uno: \uno \\ \def\dos{\uno\uno} Dos: \dos \\ \def\uno{uno} Uno: \uno \ Dos: \dos</pre>	Uno: 1 Dos: 11 Uno: uno Dos: unouno
---	---

En ocasiones, puede necesitarse definir un comando que sea independiente de los cambios que se produzcan en los comandos sobre los que está definido. Con esta utilidad está construido el comando `\let`, que “saca una copia” de un comando para que funcione siempre de la misma manera, con independencia de redefiniciones posteriores de comandos. Se utiliza con la sintaxis:

```
\let\NuevoComando=\ComandoExistente
```

que puede usarse también en caso de comandos con argumentos (cuidando de que el comando antiguo y su “copia” tengan el mismo número de argumentos). El ejemplo siguiente ilustra el funcionamiento de `\let`:

<pre>\noindent\def\uno{1}Uno: \uno \\ \def\dos{\uno\uno} Dos: \dos \\ \let\UNO=\uno \def\DOS{\UNO\UNO} \def\uno{uno} Uno: \uno \\ Dos: \dos \ DOS: \DOS</pre>	Uno: 1 Dos: 11 Uno: uno Dos: unouno DOS: 11
---	--

4.5. Manipulación de contadores y longitudes a través de TEX

Veamos ahora cómo se trabaja con contadores y longitudes desde el punto de vista de TEX. Se pueden realizar operaciones con tres tipos de magnitudes:

Contadores Corresponden a registros tipo `count`, y se definen con el comando:

`\newcount\NuevoContador;` el registro puede almacenar números enteros entre -214783647 y +214783647

Longitudes rígidas Corresponden a registros tipo `dimen`, y se definen con el comando:

`\newdimen\NuevaLongitud`

Longitudes elásticas Existen dos tipos de registro:

- `skip`: se definen con `\newskip\NuevaLongitud`
- `muskip`: análogo de longitud elástica, que se utiliza sólo en el modo matemático; se definen con `\newmuskip\NuevaLongitud`

Existen comandos para realizar las cuatro operaciones aritméticas básicas (suma, resta, multiplicación y división) con todos los registros anteriores (ambas sintaxis, con `advance` ó `advance by`, etc..., son equivalentes):

- `\advance\NombreRegistro \pm Número \advance\NombreRegistro by \pm Número` donde Número debe ser una longitud, si tratamos con registros de tipo longitud, ó un número entero, si trabajamos con un contador.
- `\multiply\NombreRegistro \pm Número \multiply\NombreRegistro by \pm Número` donde Número debe ser *siempre* un entero.
- `\divide\NombreRegistro \pm Número \divide\NombreRegistro by \pm Número` Número también debe de ser un entero. En el caso de un contador, se almacenará la parte entera de la división; en el caso de longitudes, éstas se transforman primero a unidades sp (la más pequeña de TEX; 1 sp = 65536 pt) y el resultado se redondea a un múltiplo entero de ésta unidad.

Veamos unos ejemplos:

```
\newskip\LongElastica
\LongElastica=%
10pt plus 1fill minus 2fill%
\par \the\LongElastica
\advance\LongElastica by %
5pt plus 3fill minus 1fill%
\par \the\LongElastica
\multiply\LongElastica by 3%
\par \the\LongElastica
\divide\LongElastica by 2
\par \the\LongElastica
```

10.0pt plus 1.0fill minus 2.0fill
15.0pt plus 4.0fill minus 3.0fill
45.0pt plus 12.0fill minus 9.0fill
22.5pt plus 6.0fill minus 4.5fill

En el caso de longitudes *rígidas*, existe un forma alternativa de multiplicarlas por un factor: `\Longitud1=Numero\Longitud2` (donde `\Longitud1` y `\Longitud2` pueden ser la misma). Este procedimiento de multiplicación tiene la ventaja de que pueden utilizarse

factores no enteros (0.5, 1.25, etc...). En el caso de que `\Longitud1` sea elástica, la acción anterior la transforma automáticamente en una rígida; por ejemplo, tras:

```
\LongElastica=10pt plus 1fill minus 2fill  
\LongElastica=2.5\LongElastica  
\the\LongElastica da como resultado: 25.0pt
```

El siguiente ejemplo ilustra cómo manejar contadores, definiendo un nuevo comando `\hora` que calcula la hora, a partir del contador `\time`, que almacena el número de minutos después de la medianoche (probar como ejercicio que el comando funciona correctamente):

```
\def\hora{\newcount\horas \newcount\minutos  
% (Definimos dos nuevos contadores)  
\horas=\time \global\divide\horas by 60  
% (la parte entera de la division produce la hora)  
\minutos=\horas \multiply\minutos by 60  
\advance\minutos by -\time  
\global\multiply\minutos by -1  
% (multiplicamos las horas por 60, restamos \time,  
% y cambiamos de signo para obtener los minutos)  
\the\horas:\ifnum\minutos<10 0\fi\the\minutos}  
% (se imprime horas:minutos, con un cero extra si minutos < 10)
```

4.6. Manejo de cajas en T_EX

Cuando T_EX compone un documento, trabaja manejando diversos objetos como si fuesen cajas con tres diferentes dimensiones (altura, anchura y profundidad), medidas con respecto a un punto de referencia. Por ejemplo, las líneas se componen alineando las cajas asociadas a cada carácter, según la línea base. Entonces, cada línea se convierte a su vez en una caja, que se alinea (ahora verticalmente), y así sucesivamente hasta que se construye la página.

Existen tres modos fundamentales de trabajo de T_EX, a la hora de componer cajas:

- **Modo horizontal:** T_EX agrupa cajas alinéandolas horizontalmente unas junto a otras, a lo largo de la línea base, creando una nueva caja de anchura igual a la suma de anchuras, y de altura y profundidad iguales a la mayor de las alturas y profundidades de las cajas, respectivamente. Existen dos sub-modos diferentes dentro del modo horizontal:
 - Ordinario: Es el característico cuando se construyen párrafos; se alinean caracteres horizontalmente, y después se va cortando para formar líneas de la misma anchura. T_EX estira ó contrae los espacios para optimizar el resultado final
 - Restringido: En este modo, sólo se alinean las cajas horizontalmente, sin posibilidad de dividir la caja resultante en cajas más pequeñas. Dentro de este modo, no se entienden los comandos asociados a saltos de línea, párrafo, etc...
- **Modo vertical:** Se agrupan las cajas verticalmente unas sobre otras (manteniendo los puntos de referencia en la misma vertical), creando una caja con anchura igual

a la mayor de las anchuras de las subcajas, y con altura total (suma de altura y profundidad) igual a la suma de alturas y profundidades de las subcajas. Al igual que para el modo horizontal, existen dos sub-modos:

- Ordinario: Es el modo por defecto, en el cual \TeX va recogiendo todas las cajas creadas en los modos horizontal y matemático, para empaquetarlas verticalmente.
- Interno: Se limita a apilar verticalmente cajas, creando una caja indivisible (por ejemplo, cuando se construyen las columnas de una tabla ó matriz)
- **Modo matemático**: Se abre para escribir símbolos ó fórmulas matemáticas, existe en dos variantes, *ordinario* (ó tipo párrafo) y *resaltado* (para fórmulas centradas y resaltadas), las cuales ya se han descrito en el capítulo correspondiente

En los sucesivo, discutiremos algunos comandos de \TeX útiles para crear y colocar cajas (que por supuesto, pueden ser utilizados dentro de cualquier documento \LaTeX ; al fin y al cabo, \LaTeX , como ya se ha repetido, no es más que un conjunto de macros construidas a partir de \TeX , que es el lenguaje de bajo nivel que realmente compila el documento fuente).

4.6.1. Cajas horizontales; \hbox

El comando $\text{\hbox}\{Material\}$ crea cajas indivisibles, en las que el contenido se escribe de izquierda a derecha. El argumento *Material* es procesado *en modo horizontal restringido*, y puede estar compuesto de varias cajas. En realidad, el comando \mbox de \LaTeX no es más que \hbox :

```
\def\mbox#1{\leavevmode\hbox{#1}}
```

(el comando \leavevmode se asegura de salir del modo vertical, si estuviésemos dentro de él).

Cada caja creada mediante \hbox tiene una anchura natural dependiente de la anchura del *Material* incluido en ella. Al igual que ocurría con el comando \makebox , es posible cambiar dicha anchura a nuestro gusto:

- $\text{\hbox to Ancho}\{Material\}$ Crea una caja de anchura *Ancho* y coloca en ella el material de izquierda a derecha. Si la anchura del material es menor que *Ancho*, se estirarán los espacios elásticos para ocupar todo el espacio disponible, mientras que si es mayor, el material sobresaldrá de la caja (con lo que se sobreescibirá en texto que venga a continuación); véase el siguiente ejemplo (donde se añade \fbox a fin de remarcar las cajas):

```
\parindent Opt Normal:  
\fbox{\hbox{caja ejemplo}}  
(texto) \\ Estiramos:  
\fbox{\hbox to 3cm{caja ejemplo}}  
(texto) \\ Contraemos: \fbox{\hbox  
to 1cm{caja ejemplo}} (texto) \\
```

Normal: caja ejemplo (texto)
Estiramos: caja exemplo (texto)
Contraemos: caja ejemp lo (texto)

- $\text{\hbox spread Ancho}\{Material\}$ Es análogo al anterior, con la diferencia de que aumenta (ó disminuye si el valor es negativo) la anchura natural de la caja en la cantidad *Ancho*:

```
\parindent 0pt
Estiramos: \fbox{\hbox spread
5mm{caja ejemplo}} (texto) \\
Contraemos: \fbox{\hbox spread
-5mm{caja ejemplo}} (texto) \\
```

Estiramos:

caja	ejemplo
------	---------

 (texto)
 Contraemos:

caja	ejempl
------	--------

(texto)

Con lo ya visto, podemos ahora entender la definición de los comandos `\leftline`, `\centerline`, `\rlap`, etc..., que muestran la potencia de combinar las manipulaciones de cajas y longitudes:

```
\def\leftline#1{\hbox to \hsize{#1\hss}}
\def\rightline#1{\hbox to \hsize{\hss#1}}
\def\centerline#1{\hbox to \hsize{\hss#1\hss}}
\def\rlap#1{\hbox to 0pt{#1\hss}}
\def\llap#1{\hbox to 0pt{\hss#1}}
```

donde recordemos que el comando `\hss` equivale a `\hspace{0pt plus 1fil minus 1fil}`; la longitud `\hsize` almacena la anchura del texto: normalmente equivale a `\textwidth`, aunque puede modificarse a voluntad.

Ejercicio:

Ponemos dos palabras en una caja

Test	Test
------	------

 de forma que haya un centímetro de separación con los extremos, y dos centímetros de separación entre las palabras.

4.6.2. Cajas verticales; `\vbox`

Las cajas verticales se construyen con el comando `\vbox{Material}`; este comando inicia el modo vertical interno, aunque es posible que T_EX ya esté en ése modo antes de invocar el comando. El comportamiento del comando depende de si la caja vertical contiene texto en el nivel más alto, o si contiene el comando `\vrule` (explicado más adelante); en ambos casos la anchura será la de una línea de texto (`\hsize`). Veamos algunos ejemplos:

```
\parindent 0pt \fbox{\vbox{Texto de prueba
\hbox{Una caja} \hbox{Otra caja}}} \fbox{\vbox{\hbox{Una
caja} Texto de prueba \hbox{Otra
caja}}} \fbox{\vbox{\hbox{Una caja}
\hbox{Otra caja} \hbox{Otra caja mas}}}
\fbox{\vbox{\hsize 4cm Texto de prueba
\hbox{Una caja} \hbox{Otra caja}}} \par
% Nótese la diferencia entre
% modos horizontal y vertical:
\fbox{\vbox{Texto de prueba}} \par
\fbox{\vbox{\hbox{Texto de prueba}}}
```

Texto de prueba	Una caja	Otra caja
Una caja		
Texto de prueba	Otra caja	
Una caja		
Otra caja	Texto de prueba	
Otra caja mas	Una caja Otra caja	
Texto de prueba		
Texto de prueba		

Al igual que con las cajas horizontales, se puede fijar de antemano la altura de una caja vertical con:

`\vbox to Alto{Material}` ó `\vbox spread Alto{Material}`

cuyo significado es idéntico a lo ya visto para `\hbox` (cambiando Ancho por Alto).

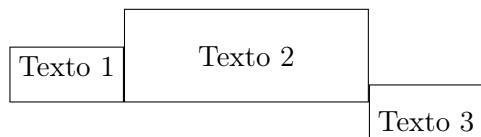
Es interesante remarcar la flexibilidad y potencia del comando `\vbox` de `TeX`; a diferencia de lo que ocurre con `\hbox`, no existen realmente comandos en `LATeX` con tanta capacidad; por ejemplo, `\parbox` ó el entorno `minipage` permiten fijar la altura de la caja, pero además requieren fijar su anchura, lo cual puede ser un inconveniente.

El comando `\vbox to Alto{Material}` alinea la línea base de la caja global con la línea base de *la última* caja (es decir, la inferior) incluida en la caja vertical. Existen también los comandos:

`\vtop to Ancho{Material}` y `\vcenter to Ancho{Material}`

que alinean, respectivamente, la parte superior y central de la caja total con la línea base (ATENCIÓN: `\vcenter` sólo se puede emplear dentro del modo matemático).

```
\parindent 0pt
\fbox{\vbox to 5mm{\hbox{Texto 1}}}
\fbox{\vbox to 10mm{\hsize 3cm%
\vfil\centerline{Texto 2}\vfil}}
\fbox{\vtop to 5mm%
{\vfil\hbox{Texto 3}}}
```



4.6.3. Moviendo cajas

Dependiendo del modo (horizontal ó vertical) en el que nos encontremos, disponemos de diversos comandos para desplazar cajas. *En el modo horizontal*, podemos desplazar cajas verticalmente con:

`\raise Desplazamiento` ó `\lower Desplazamiento`

donde Desplazamiento es cualquier longitud. De hecho, éstos dos comandos son esencialmente el mismo, ya que `\raise D = \lower -D`. Al usar éstos comandos, la línea base queda inalterada, aunque la altura y profundidad pueden cambiar. La nueva altura y profundidad de la caja se calculan dependiendo de los desplazamientos. Véase el siguiente ejemplo, donde se remarca la línea base con el comando `\hrule` (descrito a continuación):

```
Texto de prueba ; Texto de prueba ; Texto de prueba ; Texto de prueba \\
\fbox{\hbox{\hbox to 0pt{\vbox{\hrule width 6cm}} \hbox{pepe}
\lower3mm\hbox{pepe} \hbox{pepe}}} \ andres \ jaime \hspace{5mm}
\fbox{\hbox{\hbox to 0pt{\vbox{\hrule width 6cm}} \hbox{pepe}
\raise5mm\hbox{pepe} \hbox{pepe}}} \ andres \ jaime \\
Texto de prueba ; Texto de prueba ; Texto de prueba ; Texto de prueba
```

Texto de prueba ; Texto de prueba ; Texto de prueba ; Texto de prueba

 Texto de prueba ; Texto de prueba ; Texto de prueba ; Texto de prueba

El comando `\kern Longitud` se utiliza con carácter general para desplazar cajas una cantidad `Longitud` (que puede ser negativa). La dirección del desplazamiento, horizontal

ó vertical, depende que en que modo esté T_EX trabajando; en el modo horizontal (en una caja \hbox) el desplazamiento es horizontal, mientras que en el modo vertical (en una caja \vbox) el desplazamiento es vertical. Veamos como ejemplo el código T_EX para obtener el logotipo “T_EX”:

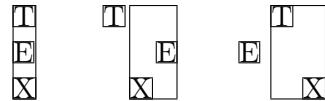
```
\hbox{T\kern-.1666em\lower.5ex\hbox{E}\kern-.125ex X}
```

Para mover horizontalmente las componentes de una caja vertical \vbox se utilizan los comandos:

```
\moveleft Desplazamiento y \moveright Desplazamiento
```

Es interesante hacer notar que la anchura de la caja tras los desplazamientos sólo se modifica con desplazamientos a la derecha, y no a la izquierda: la anchura se calcula comenzando en el punto de referencia y extendiéndose hacia la derecha hasta la parte derecha de la componente más alejada; por ejemplo (nótese cómo ahora se utiliza \frame para remarcar las cajas, en vez de \fbox, que dejaba un pequeño espacio \fboxsep alrededor):

```
\frame{\vbox{\hbox{\frame{T}}%  
 \hbox{\frame{E}\hbox{\frame{X}}}}}  
 \hspace{1cm}\frame{\vbox{\moveleft10pt%  
 \hbox{\frame{T}}\moveright10pt\hbox%  
 {\frame{E}}\hbox{\frame{X}}}}%  
 \hspace{1cm}  
 \frame{\vbox{\hbox{\frame{T}}%  
 \moveleft12pt\hbox{\frame{E}}%  
 \moveright12pt\hbox{\frame{X}}}}
```



4.6.4. Modificando, creando y reutilizando cajas

Hemos visto ya el modo de guardar y reutilizar cajas en L^AT_EX; veremos ahora el modo, más general, de manipularlas a través de T_EX. Podemos declarar una caja nueva con el comando: \newbox\NombreCaja, tras lo cual se almacena una caja en la variable \NombreCaja con:

```
\setbox\NombreCaja=Caja
```

A diferencia del comando \sbox, que sólo maneja cajas horizontales, con el comando \setbox Caja puede ser tanto horizontal como vertical.

Otra forma de declarar cajas es hacer uso de los registros (256) de los que T_EX dispone para guardar cajas. Están numerados de 0 a 255, estando el número 255 reservado para la caja de la página. Podríamos entonces, en vez de declarar primero NombreCaja con \newbox, crear directamente cajas numeradas con:

```
\setbox1=Caja1 \setbox2=Caja2 etc...
```

(tras lo cual, para todos los comandos descritos a continuación, se debería reemplazar \NombreCaja por 1, 2, ...)

Para insertar dentro de un documento los contenidos de una caja, se emplean los comandos:

```
\box\NombreCaja Tras ser usado, borra el contenido de la caja
```

```
\copy\NombreCaja Usa el contenido de la caja sin borrarlo
```

Por ejemplo: \setbox1=\hbox{A} \fbox{\box1} \fbox{\box1} produce , mientras

que `\setbox1=\hbox{A} \fbox{\copy1} \fbox{\copy1}` produce 

Para una caja `\NombreCaja` dada, las siguientes *longitudes* almacenan, respectivamente, los valores de anchura, altura y profundidad de la caja:

`\wd\NombreCaja` `\ht\NombreCaja` `\dp\NombreCaja`

Véase el siguiente ejemplo; definimos: `\newbox\NuevaCaja` y asignamos `\setbox\NuevaCaja=\hbox{A B C}` tras lo cual, `\the\wd\NuevaCaja` produce 30.99307pt (la anchura de la caja). Podemos “estirar” la caja con: `\wd\NuevaCaja=2\wd\NuevaCaja`, tras lo cual `\frame{\copy\NuevaCaja}` produce

Los siguientes comandos son análogos a `\box` y `\copy`, pero, en vez de simplemente escribir el contenido de la caja, las “desmembran” en sus subcomponentes en el momento de ser usadas. Hay versiones horizontal y vertical, así como versiones “`\box`” y “`\copy`”, que respectivamente vacían ó no la caja tras ser usada:

<code>\unhbox\NombreCaja</code>	<code>\unvbox\NombreCaja</code>
<code>\unhcopy\NombreCaja</code>	<code>\unvcopy\NombreCaja</code>

El siguiente ejemplo ilustra la diferencia entre simplemente copiar una caja, y desmembrarla con `\unhbox`:

```
\setbox1=\hbox{A B}
\setbox2=\hbox to 2.0\wd1{\unhcopy1}
\frame{\copy2}
\setbox3=\hbox{A B}
\setbox4=\hbox to 2.0\wd3{\copy3}
\frame{\copy4}
```



en el primer caso, tras desmembrar la caja, al construir una caja de anchura doble a la primitiva los elementos se reparten tratando de llenar toda la caja; en el segundo, al estar la caja intacta, los elementos se mantienen a la derecha de la caja `\box4`, que contiene a `\box3`

4.6.5. Rayas horizontales y verticales

En TEX se pueden utilizar dos tipos de rayas ó cajas negras; las horizontales, `\hrule`, y las verticales, `\vrule`. Para cada una de ellas se pueden especificar tres dimensiones: anchura, altura y profundidad:

`\hrule height Altura width Anchura depth Profundidad`
`\vrule height Altura width Anchura depth Profundidad`

puede omitirse cualquiera de estos tres parámetros, en cuyo caso TEX asignará valores por defecto:

- Altura 0.4 pt y profundidad 0 pt, si la raya es horizontal (`\hrule`)
- Anchura 0.4 pt, si la raya es vertical (`\vrule`)
- El resto de dimensiones se obtiene extendiendo la raya indefinidamente hasta completar el tamaño de la caja que la contiene

La diferencia esencial entre `\hrule` y `\vrule` reside en que `\hrule` es material *vertical*, por lo que sólo puede ser utilizado entre párrafos ó dentro de una caja vertical `\vbox`, mientras que `\vrule` es material *horizontal*, por lo que sólo puede utilizarse dentro de un párrafo ó de una caja horizontal `\hbox`.

Ejemplos:

```
\hbox{Ejemplo \vrule width 2pt\vbox to
25pt{linea \par vertical}} \vspace{3mm}
\vbox{\hbox to 4cm{Otro ejemplo
\kern 1mm\hrule height 1pt\kern 1mm
\hbox to 3cm{linea horizontal}}}
```

Ejemplo	linea
vertical	
Otro	ejemplo
linea	horizontal

```
\hbox{\vbox{\hbox to 25mm{\hfil%
\vbox{Texto 1}\hfil}\kern2pt\hrule}%
\vrule \lower7.5mm\vbox to 15mm{\hrule%
\kern-11pt\hbox to 25mm{\hfil%
\vbox{Texto 2}\hfil}\vfil\hbox to
25mm{\hfil\hbox{Texto 3}\hfil}%
\kern2pt\hrule}}
```

4.7. Repetición de objetos

Veremos ahora otro modo de repetir objetos, ligeramente diferente del comando `\multiput` ya visto. En vez de proporcionar el número de objetos a repetir, puede interesarnos llenar un cierto espacio, de longitud fija ó variable, con copias de un objeto. Mediante el comando `\leaders` se pueden obtener copias de un objeto en tal forma. Para ello, debemos especificar el objeto a copiar y el espacio que debe ser completado con copias de tal objeto. La sintaxis del comando es la siguiente:

`\leaders Objeto \hskip Longitud`

donde hay que tener en cuenta que *Objeto* debe ser una *caja*, y *Longitud* puede ser cualquier longitud (incluyendo elásticas). Se puede reemplazar `\hskip 1fil` por simplemente `\hfil`, ó `\hskip 1fill` por `\hfill`. Por ejemplo:

```
\noindent\null\leaders\hrule\hfill
\null\\[2mm]
\null\leaders\hbox{/\\textbackslash}%
\hskip.4\hsize\null\\[2mm] \hbox to
4cm{\leaders\hbox{\frame{\hbox to
10pt{\vbox to 10pt{}}}}\hfill}
```

Es importante tener en cuenta que es necesario “marcar” los puntos entre los que actúa el comando `\leaders`, sobre todo si estamos utilizando longitudes elásticas. Es por eso que se utiliza en el ejemplo anterior `\null` (otras posibilidades equivalentes serían `\mbox{}` ó `\kern0pt`).

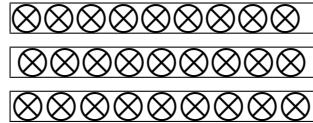
En el caso de que el objeto a repetir no sea un múltiplo entero del tamaño de la caja que contiene a las copias del objeto, aparecerá cierta asincronía. Para solucionar esto, se dispone de otras dos variantes de `\leaders` para repetir un objeto:

`\cleaders Objeto \hskip Longitud`

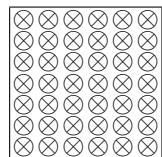
`\xleaders Objeto \hskip Longitud`

mientras que `\leaders` aparta el espacio sobrante a la derecha, `\cleaders` reparte el espacio sobrante a ambos lados (centrando las copias del objeto en la caja); `\xleaders` distribuye el espacio sobrante entre cada copia del objeto, ajustando las copias del objeto a la caja; por ejemplo:

```
\frame{\hbox to 4cm{\leaders%
\hbox{$\bigotimes$}\hfill}}\par
\frame{\hbox to 4cm{\cleaders%
\hbox{$\bigotimes$}\hfill}}\par
\frame{\hbox to 4cm{\xleaders%
\hbox{$\bigotimes$}\hfill}}
```



De igual manera a lo ya visto, se pueden repetir objetos en la dirección vertical; para ello, simplemente se cambia el segundo argumento del comando `\leaders` de horizontal (`\hskip Longitud`) a vertical (`\vskip Longitud`). Como ejercicio, escribir el código T_EX para obtener:



(la clave está en repetir el objeto `\otimes` primero horizontalmente, y repetir la repetición verticalmente a continuación)

4.8. Sistematizando tareas

Veremos ahora una serie de comandos útiles para sistematizar tareas; imaginemos que cada vez que se inicia un párrafo, fórmula, caja, ..., queremos que se ejecuten una serie de comandos. Para ello, se tienen las siguientes posibilidades:

- `\everypar{Comandos}` Antes de empezar a dar formato a cada párrafo, se ejecutan Comandos. Veamos un ejemplo, en el que definimos un nuevo contador `parrafo`, y para cada párrafo, se le pone como título “Párrafo n” en negrita y centrado:

```
\parindent 0pt \parskip 10pt
\setcounter{parrafo}{0}
\everypar{\addtocounter{parrafo}{1}}
\centerline{\bfseries Párrafo %
\the parrafo}\v[1mm]
Esto es un primer párrafo de
ejemplo; en el código anterior... \par
...se puede ver cómo incrementamos
el valor del contador \texttt{parrafo}
al empezar cada párrafo... \par ...y lo
recuperamos, para imprimirlo, con el
comando \verb+\the parrafo+\par MUY
IMPORTANTE: Nótese cómo, en este
documento, el comando
\verb+\everypar{...}+ es local; al
estar incluido dentro de un
entorno multicols (utilizado para
escribir este ejemplo) no tiene
efecto al salir del entorno
```

Párrafo 1

Esto es un primer párrafo de ejemplo; en el código anterior...

Párrafo 2

...se puede ver cómo incrementamos el valor del contador `parrafo` al empezar cada párrafo...

Párrafo 3

...y lo recuperamos, para imprimirlo, con el comando `\the parrafo`

Párrafo 4

MUY IMPORTANTE: Nótese cómo, en este documento, el comando `\everypar{...}` es local; al estar incluido dentro de un entorno `multicols` (utilizado para escribir este ejemplo) no tiene efecto al salir del entorno

- `\everymath{Comandos}` Análogo a `\everypar`; se ejecutan los comandos cada vez que entremos en modo matemático *ordinario* (ó modo texto)
- `\everydisplay{Comandos}` En este caso, los comandos se ejecutan cada vez que se abre el modo matemático *resaltado*. Imaginemos que queremos que todas las fórmulas resaltadas se escriban en color rojo; para conseguirlo, simplemente se puede declarar: `\everydisplay{\color{red}}`
- `\everyhbox{Comandos}` y `\everyvbox{Comandos}` Ejecutan los comandos cada vez que comience una caja horizontal (`\hbox`) ó vertical (`\vbox`), respectivamente.

4.9. Condicionales y bucles

El compilador TeX posee amplias capacidades a la hora de programar diversas acciones. Además de poder manejar diversos registros (contadores, longitudes, cajas) con total flexibilidad, su potencia se ve reforzada al ser posible incluir bucles y condicionales dentro de un documento.

Un condicional es una estructura de control que elige entre diversas acciones en función del valor de una variable lógica; su forma general es:

`IF <Test> [Instrucciones A] ELSE [Instrucciones B] END IF`

lo cual significa que, de cumplirse la condición `<Test>`, se ejecutarán las instrucciones “A”, y de no cumplirse, las instrucciones “B”. En lenguaje TeX, el condicional se escribe:

`\if<Test> [Parte A] \else [Parte B] \fi`

aunque podemos prescindir de cualquiera de las partes (A ó B), y tener simplemente:

`\if<Test> [Parte A] \fi` ó `\if<Test> \else [Parte B] \fi`

(el último, correspondería a una versión “de negación” del condicional).

Los condicionales pueden anidarse sin problemas; cada `\fi` se asume que corresponde con el más reciente `\if`. A continuación describiremos algunos de los 17 condicionales que están definidos en TeX, correspondientes a diversos formatos de la condición `<Test>`:

- `\ifnum Número1 Relación Número2`

Se utiliza para comparar números enteros, con `Relación` igual a `<`, `>` ó `=`. Como ejemplo, definimos un contador `\cuatrodigit`, que imprime números en formato de cuatro dígitos, con independencia de su tamaño:

```
\def\cuatrodigit#1{%
\ifnum #1<1000 0\fi
\ifnum #1<100 0\fi
\ifnum #1<10 0\fi #1}

```

tras lo cual,
`\cuatrodigit{8} - \cuatrodigit{18} -`
`\cuatrodigit{198} - \cuatrodigit{1238}`
produce:
0008 - 0018 - 0198 - 1238

- `\ifodd Número`

sirve para comprobar si un número entero es impar. En el caso de que queramos analizar el valor de un determinado contador, recordemos que debemos sustituir `Número` por `\value{NombreContador}`, si estamos trabajando con un contador definido en LATEX. Por contra, si el contador ha sido definido en TeX (con `\newcount`), podemos recuperar el valor numérico con `\the\NombreContador` ó `\number\NombreContador`

Ejemplo: Compilando Esta página es \iffodd{page} impar \else par\fi obtendremos “Esta página es impar”, si es impar, ó “Esta página es par” si es par.

- **\ifdim Dimensión1 Relación Dimensión2**

se utiliza para comparar dos longitudes. Como ejemplo, vamos a construir un comando que crea una caja enmarcada con un texto en tamaño \huge (que será el primer argumento del comando) y un texto de leyenda, que se colocará centrada si la longitud de la leyenda es menor que la del texto principal, ó en estilo párrafo si es mayor:

```
\newlength{\anchura}
\def\textoresaltado#1#2{%
\setbox1=\hbox{\fbox{\huge#1}}
\settowidth{\anchura}{#2}\vbox{\copy1%
\vspace{6pt}\ifdim\anchura<\wd1\hbox
to\wd1{\hss#2\hss}\else%
\hbox{\parbox{\wd1}{#2}}\fi}%
\textoresaltado{Juan y Ana}{quieren
un coche}\par\medskip
\textoresaltado{Juan y Ana}{necesitan
comprarse un coche nuevo porque el
antiguo se les ha quedado viejo}
```

Juan y Ana

quieren un coche

Juan y Ana

necesitan comprarse
un coche nuevo por-
que el antiguo se les
ha quedado viejo

- **\ifhmode**

- **\ifvmode**

- **\ifmmode**

sirven para comprobar, respectivamente, si estamos dentro del modo horizontal, vertical, ó matemático (en cada caso, no se distingue entre los diferentes sub-modos). Por ejemplo, el comando \ensuremath de LATEX está definido como:

```
\newcommand{\ensuremath}[1]{\ifmmode #1\else $#1$\fi}
```

- **\ifcase Número [Caso n=0] \or [Caso n=1] \or ...**
[Caso n=M] \else [Caso n>Otro Número] \fi

sirve para ejecutar diferentes acciones, de acuerdo a los valores que tome la variable Número (que puede, por ejemplo, ser un contador); si n=0 se ejecutarán las primeras instrucciones, si n=1 las, segundas, y así sucesivamente hasta M; opcionalmente podemos colocar más instrucciones después de \else, que se ejecutarán si Número es menor que 0 ó mayor que M. Véase el siguiente ejemplo, que traduce números naturales a notación hexadecimal:

```
\def\hexadec#1{\ifcase #1 %
0\or 1\or 2\or 3\or 4\or %
5\or 6\or 7\or 8\or 9\or %
A\or B\or C\or D\or E\or F\fi}
```

Comando	Resultado
\hexadec{7}	7
\hexadec{12}	C

- **\ifx Argumento1Argumento2**

compara dos argumentos entre sí, siendo verdadero si son iguales y falso si son distintos. Argumento1 y Argumento2 pueden ser caracteres, cajas, comandos... Es importante puntualizar que, a la hora de comparar cadenas de caracteres ó cajas, es necesario con anterioridad incluir tales contenidos en sendos comandos, que serán después comparados. Por ejemplo:

```

\columnbreak \vspace*{0.1cm}
\def\aa{Hola} \def\bb{Hola} \def\cc{hola}
\def\dd{H} \def\ee{\hbox{hola}}
\def\ff{\hbox{hola}} Comparación 1:
\ifx AA iguales \else distintos \fi
% (caracteres aislados son comparables)
Comparación 2:
\ifx \aa\bb iguales \else distintos \fi
Comparación 3:
\ifx \aa\cc iguales \else distintos \fi
Comparación 4:
\ifx H\dd iguales \else distintos \fi
Comparación 5:
\ifx \cc\ee iguales \else distintos \fi
Comparación 6:
\ifx \ee\ff iguales \else distintos \fi

```

Comparación 1: iguales
 Comparación 2: iguales
 Comparación 3: distintos
 Comparación 4: distintos
 Comparación 5: distintos
 Comparación 6: iguales

Pregunta de trivial: ¿Por qué en el primer caso hay un espacio extra?

\LaTeX también proporciona algunos condicionales predefinidos, que es bueno conocer:

- `\if@twoside`
- `\if@twocolumn`

son verdaderos si se está procesando el documento con las opciones `twoside` ó `twocolumn`, respectivamente, y falsos en caso contrario.

- `\@ifnextchar Carácter{ParteA}{ParteB}`

Se procesa `ParteA` en caso de que el siguiente carácter coincida con `Carácter`, y `ParteB` en caso contrario. Este condicional es muy utilizado en \LaTeX en los comandos que utilizan argumentos opcionales, caracterizados por ir entre corchetes. Veamos un ejemplo de cómo definir un comando con dos argumentos, uno de ellos optativo; queremos recuadrar un texto dado con una línea de grosor variable, 0.4 pt por defecto:

```

\def\mirecuadro[#1]#2{{\fboxrule#1\fbox{#2}}}
\makeatletter
\def\recuadro{@ifnextchar[\mirecuadro]{\mirecuadro[0.4pt]}}
\makeatother

```

Tras esto, `\recuadro{Prueba}` resultará en Prueba, mientras que si queremos cambiar el grosor de línea a 1pt, deberemos escribir: `\recuadro[1pt]{Prueba}` → Prueba. ¿Por qué los comandos `\makeatletter` y `\makeatother`? La razón está en que, por defecto, está prohibido utilizar el símbolo @ en los comandos dentro de un documento; éste comando se utiliza frecuentemente en las clases de documentos ó paquetes, por lo que se restringe su uso para evitar coincidencias casuales con comandos ya definidos. El comando `\makeatletter` levanta esta prohibición, mientras que el comando `\makeatother` la vuelve a recuperar.

Podemos utilizar lo aprendido en el ejemplo anterior para construir comandos más complicados. Por ejemplo, modifiquemos el comando `\recuadro` (renombrándolo a `\Recuadro`) para que ahora admita dos argumentos optativos, según la sintaxis:

```
\Recuadro[Grosor](Color){Texto}
```

siendo `Grosor` la anchura del recuadro (0.4pt por defecto), y `Color` su color (rojo por defecto). Utilizando recursivamente el condicional `\@ifnextchar` se obtiene el resultado deseado (comprobar):

```
\makeatletter
```

```
\def\Mybox(#1){\color{#1}\fbox{\color{black}#2}}
\def\Myboxaux[#1]{\fboxrule#1\@ifnextchar(%
{\Mybox}{\Mybox(red)}}
\def\Recuadro{@ifnextchar[{ \Myboxaux}{\Myboxaux[0.4pt]}}
\makeatother
```

4.9.1. Nuevos condicionales

Volviendo a T_EX, veremos ahora la forma de definir nuevos condicionales con el comando `\newif`, de sintaxis:

```
\newif\ifNombre
```

donde *Nombre* corresponderá al nombre del nuevo condicional. El comando `\newif` se encarga de definir tres nuevos comandos:

- `\iiftrue` Asigna a la variable lógica *Nombre* el valor verdadero
- `\iiffalse` Asigna a la variable lógica *Nombre* el valor falso
- `\ifNombre... \else... \fi` Nuevo condicional, que ejecuta una acción u otra según el valor que se le haya asignado anteriormente a la variable lógica *Nombre*

Como ejemplo, definamos un nuevo entorno `ocultar`, de forma que el texto dentro de tal entorno se muestre ó no en el documento final, dependiendo del valor de una variable lógica:

```
\newbox\boxocultar
\newif\ifocultar
\newenvironment{ocultar}
{\setbox\boxocultar\vbox\bgroup}
{\egroup\ifocultar\else\par\unvbox\boxocultar\fi}
```

tras esta definición, si se coloca el comando `\iiffalse`, todo el texto dentro de entornos `ocultar` que estén a continuación de este comando no se verá en el documento final; en cambio, sustituyéndolo por `\iiftrue`, se reestablecerá el texto dentro de tales entornos, por ejemplo:

```
\iiftrue \begin{ocultar}
texto de prueba que no se ve
\end{ocultar}
\iiffalse \begin{ocultar}
texto de prueba que si se ve
\end{ocultar}
```

texto de prueba que si se ve

La utilidad de éste entorno puede estar, por ejemplo, en la inclusión de notas y comentarios que puede convenirnos suprimir en el documento final; añadir un comando `\iiftrue` es más rápido que comentar líneas una por una. Merece la pena analizar un poco la definición del nuevo entorno:

- 1) `\newbox\boxocultar` → Define una nueva caja para almacenar el texto oculto
- 2) `\setbox\boxocultar\vbox\bgroup` → Abre una caja vertical y la almacena en `\boxocultar`; nótese el empleo del comando `\bgroup`: éste comando es análogo a “{”,

es decir, es un delimitador de grupo. La sutileza radica en que, de usar directamente {}, habría un conflicto con la sintaxis del comando `\newenvironment`

3) `{\egroup\ifocultar\else\par\unvbox\boxocultar\fi}` → Tras haber abierto la caja vertical, y haberse llenado con todo el texto dentro del entorno, se cierra con `\egroup` (análogo a “ } ”); recordemos que todo lo que iba entre el primer conjunto de llaves corresponde a las instrucciones L^AT_EX a ejecutar *al entrar en el entorno*, mientras que este segundo conjunto de instrucciones corresponde a lo que debe hacerse *al salir del entorno*. Tras eso, se comprueba con `ifocultar` si el texto debe ocultarse, en cuyo caso, no se hace nada, y, en caso contrario, se deshace e imprime la caja `\boxocultar`

4.9.2. Bucles

Se realizan bucles con el comando:

```
\loop ParteA \if... ParteB \repeat
```

donde `ParteA` y `ParteB` son conjuntos de comandos, y `\if` es cualquier condicional, sin la correspondiente partícula `\fi`. T_EX procesa primero `ParteA`; si la condición es verdadera, procesa `ParteB`, y repite el proceso comenzando de nuevo por `ParteA`; si no, inmediatamente se sale del bucle. Definamos como ejemplo un comando que imprima los primeros n números naturales:

```
\newcount\minum
\def\numeros{\ifnum#1< 1%
\else 1\minum=1\loop
\advance\minum by 1%
\ifnum\minum<#1,
\the\minum\repeat\fi}
\numeros{40}
```

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24,
25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
36, 37, 38, 39, 40

En L^AT_EX, están predefinidos bucles asociados a condicionales específicos, útiles para manejar números y longitudes:

- `\@whilenum{TestNum} \do {Acción}`
- `\@whiledim{TestLong} \do {Acción}`

En ellos, se evalúa la relación numérica `TestNum` ó `TestLong` (comparación de números ó longitudes, respectivamente); *mientras sea verdadera* se procesarán las instrucciones en `Acción`, terminando el bucle en el momento en que la relación sea falsa. El siguiente ejemplo calcula la sucesión de todos los números *pares* menores que uno dado:

```
\newcount\cuenta
\makeatletter \def\pares#1{%
\minum=2\@whilenum\minum<#1\do
{\the\minum, \advance\minum by 2}}
\makeatother
Los números pares menores que 95
son: \pares{95}
```

Los números pares menores que 95 son: 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94,

También es posible definir bucles en los que la condición de control sea un condicional `\ifNombre` definido en T_EX con `\newif`:

- `\@whilesw\ifNombre\fi{Acción}`

con lo que T_EX procesará los comandos de Acción hasta que el condicional `\ifNombre` sea falso.

Finalmente, se pueden también construir estructuras “for / next”, en las cuales se ejecuta una serie de acciones para cada uno de los elementos de una lista. Se utiliza la sintaxis:

```
\@for\Nombre:=\lista\do{Acción}
```

donde `\Nombre` es una variable (que no hace falta definirla previamente) que va almacenando los diferentes elementos de una lista (`\lista`), que debe ser previamente definida con `\def`; los elementos de la lista han de estar separados entre sí por comas. En el siguiente ejemplo, primeramente definimos a través de T_EX un comando `\longitud{Palabra}` para contar el número de letras de una palabra, que luego utilizamos para crear un tabla con las longitudes de una lista de palabras almacenadas en `\lista`:

```
\newcount\nna \def\longitud#1{\nna=0%
\expandafter\contar#1\end\number\nna}
\def\contar#1{%
\ifx#1\end\let\next=\relax
\else\advance\nna by1
\let\next=\contar\fi\next}
La longitud de la frase ''pepe tiene
un coche'' es de \longitud{pepe
tiene un coche} caracteres no blancos

\def\lista{Pepe,Juan,Andrés,Antonio}
\begin{tabular}{l}
Nombre y longitud \\
\hline
\makeatletter \@for\nombre:=\lista
\do{\hbox to 30mm{\nombre\hss}%
\longitud{\nombre} \\ } \makeatother
\end{tabular}
```

La longitud de la frase ”pepe tiene un coche” es de 16 caracteres no blancos

Nombre y longitud	
Pepe	4
Juan	4
Andrés	6
Antonio	7

4.9.3. Otros ejemplos

Invertir una palabra

```
\def\Invertir#1{%
\def\INV{}\INV\INVCAD#1\end\INV}%
\def\INVCAD#1{%
\ifx#1\end\let\next=\relax
\else\CONCAD#1%
\let\next=\INVCAD\fi\next}%
\def\CONCAD#1{\edef\INV{\#1\INV}}%
\Invertir{Espejo}Espejo\par
Curioso\Invertir{Curioso}
```

ojepsEEspejo
CuriosoosoiruC

Numeros primos

```
\newif\ifprime \newif\ifunknown %
\newcount\n \newcount\p %
\newcount\d \newcount\a %
\def\primes#1{2,{\bf ,}#1} % (#1 is at least 3)
  \n=#1 \advance\n by-2 % n more to go
  \p=5 % odd primes starting with p
  \loop\ifnum\n>0 \printifprime\advance\p by2 \repeat}
\def\printp{, % we will invoke \printp if p is prime
\ifnum\n=1 \fi
\number\p \advance\n by -1 }
\def\printifprime{\testprimality \ifprime\printp\fi}
\def\testprimality{{\d=3 \global\primetrue
\loop\trialdivision \ifunknown\advance\d by2 \repeat}}
\def\trialdivision{\a=\p \divide\a by\d
\ifnum\p>\d \unkowntrue\else\unknownfalse\fi
\multiply\p by\d
\ifnum\p=\a \global\primefalse\unknownfalse\fi}
```

Tras lo cual, \primes{200} calcula e imprime los 200 primeros números primos:

2, 3 , 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997, 1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049, 1051, 1061, 1063, 1069, 1087, 1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129, 1151, 1153, 1163, 1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223

Apuntes de L^AT_EX

Capítulo 9: Manejo de recursos de color

1. El paquete color; cambiando el color de las fuentes

En el primer capítulo vimos cómo el paquete `color` permite colorear una parte del texto, mediante los comandos:

- `\textcolor{NombreColor}{Texto}`
- `\color{NombreColor}`

y cómo mediante la opción `usenames` podemos utilizar una serie de colores básicos. Es importante destacar aquí (algo que obviamos en el capítulo 1) que la opción `usenames` sólo proporciona los 68 colores correctamente cuando se compila el documento a través de la opción **L^AT_EX + dvips + ps2pdf**, mientras que compilando con **PDFL^AT_EX** sólo se obtienen unos pocos colores básicos predefinidos, por ejemplo:

```
\textcolor{white}{Hola} \textcolor{black}{Hola} \textcolor{yellow}{Hola}  
\textcolor{green}{Hola} \textcolor{blue}{Hola} \textcolor{red}{Hola}  
\textcolor{cyan}{Hola}
```

Hola Hola Hola Hola Hola Hola

Si necesitamos colores extra, podemos obtenerlos a partir de la especificación de **modelos de color** que proporcionan los mismos comandos `\textcolor` y `\color`:

```
\textcolor[modelo]{codigo}{Texto a cambiar de color} \color[modelo]{codigo}
```

donde la etiqueta `modelo` designa el tipo de codificación del color y el `código` las coordenadas de ese color dentro de su tipo de codificación, eventualmente separadas por comas. Se tienen las siguientes tres posibilidades para el modelo:

rgb (Red Green Blue) Este es un sistema de composición de color a partir de la combinación de los colores rojo, verde y azul como colores primarios. La identificación de un color se hace por una terna de números comprendidos entre 0 y 1 que fijan la proporción de cada uno en la mezcla

cmyk (Cyan Magenta Yellow Black) Debemos especificar un conjunto de cuatro números comprendidos entre 0 y 1 representando la proporción de cada uno de los cuatro colores en la mezcla (método usado en impresoras)

gray escala de grises, un número único comprendido entre 0 y 1.

El comando \color posee las mismas características, con la conocida excepción de aplicarse a todo el material que siga, en vez de a una cadena corta de texto.

Ejemplos:

\textcolor[rgb]{1,0,0}{Rojo} \\	Rojo
\textcolor[rgb]{1,1,0}{Amarillo} \\	Amarillo
\textcolor[rgb]{0.2,0.5,0.7}{Azulado} \\	Azulado
\textcolor[cmyk]{0,1,0,0}{Magenta} \\	Magenta
\textcolor[cmyk]{1,0,1,0}{Verde} \\	Verde
\textcolor[gray]{0.3}{Gris Oscuro} \\	Gris Oscuro
\textcolor[gray]{0.8}{Gris Claro} \\	Gris Claro

Finalmente, un útil “shortcut” en los menús de iconos de WinEdt (marcado con el símbolo de una paletita) permite elegir entre unos cuantos colores predefinidos, ó modificarlos a voluntad variando parámetros rgb, insertando directamente en el documento la instrucción \textcolor[modelo]{codigo}{Texto} cuando remarcamos un trozo de texto.

2. Definiendo nuevos colores

En caso de que exista un color en particular (por ejemplo el azul claro determinado por el modelo RBG con parámetros 0.8,0.85,1.0)que utilizemos frecuentemente, podemos asignarle un *nombre* y definir el nuevo color mediante el comando:

\definecolor{nombre-color}{modelo color}{codigo color}

Por ejemplo: \definecolor{AzulClaro}{rgb}{0.8,0.85,1}, para luego utilizarlo mediante: \textcolor{AzulClaro}{Texto Azul Claro} que produciría: *Texto Azul Claro*

Tales definiciones conviene situarlas en el preámbulo del documento.

3. Páginas en color

Los colores de fondo de las páginas se pueden cambiar con los comandos

- \pagecolor{NombreColor}
- \pagecolor[modelo]{codigo}

Ambos comandos cambian el color del fondo de la página actual y las siguientes hasta que se cambie de nuevo el color de fondo, o bien se use \pagecolor{white} para regresar a páginas sin color de fondo. En el presente documento, se ha utilizado el comando:

\pagecolor[gray]{0.9}

para cambiar el color de fondo de ésta página, que es posteriormente cambiado de nuevo con \pagecolor{white} en la página siguiente.

4. Cajas coloreadas

Podemos obtener cajas coloreadas con los comandos:

- `\colorbox{NombreColor}{texto}`
- `\colorbox[modelo]{codigo}{texto}`
- `\fcolorbox{NombreColor contorno}{NombreColor relleno}{texto}`
- `\fcolorbox[modelo]{codigo contorno}{codigo relleno}{texto}`

donde `\colorbox` y `\fcolorbox` producen cajas sin marco y con marco, respectivamente.

Los siguientes ejemplos ilustran el uso de estos comandos, que permiten, al igual que `\fbox`, la modificación de grosores y separación de los marcos redefiniendo `\fboxrule` y `\fboxsep`:

```
\colorbox{blue}{este texto esta  
en una caja de color azul}\  
\colorbox[rgb]{0.5,0.5,1}{esta caja  
es una mezcla de rojo y azul al 50\%}\  
\fcolorbox{red}{yellow}{caja de fondo  
amarillo y contorno rojo}\  
\setlength{\fboxrule}{3 pt}  
\fcolorbox{red}{yellow}{caja de fondo  
amarillo y contorno rojo}\  
\setlength{\fboxsep}{5pt}  
\fcolorbox{red}{yellow}{caja de fondo  
amarillo y contorno rojo}
```

texto en una caja de color azul
caja mezcla de rojo y azul al 50 %
caja de fondo amarillo y contorno rojo
caja de fondo amarillo y contorno rojo
caja de fondo amarillo y contorno rojo

Ejercicio:

a) Dibujar la siguiente caja coloreada:

Texto en caja con doble marco

b) Escribir un comando `\doblecaja`, dependiente de tres argumentos, que enmarque un texto dado con el tipo de doble caja coloreada anterior, y que permita cambiar los colores del relleno de la caja interior y del texto recuadrado. Por ejemplo:

```
\doblecaja{black}{white}{Texto de prueba blanco en fondo negro}
```

Texto de prueba blanco en fondo negro

```
\doblecaja{green}{black}{Texto de prueba negro en fondo verde}
```

Texto de prueba negro en fondo verde

Apuntes de L^AT_EX

Capítulo 10: Conceptos avanzados sobre estructura de tablas

1. El paquete array

El paquete `array` introduce una serie de mejoras sobre el entorno `tabular` estándar. La tabla siguiente describe los nuevos elementos introducidos por el paquete:

Argumentos de los entornos estándar <code>tabular</code> y <code>array</code>	
<code>l, r, c, p{Ancho}</code>	Formatos de columna, que mantienen su significado básico. Para <code>p{Ancho}</code> , el texto se coloca justificado a la parte de arriba de la casilla.
<code> @{Objeto}</code>	Mantienen su significado, aunque se introduce una mejora para ' ': el espacio entre columnas se incrementa en el valor correspondiente a la anchura de la raya vertical.
Nuevos argumentos con el paquete <code>array</code>	
<code>m{Ancho}</code>	Nuevo especificador de columna, similar a <code>p{Ancho}</code> , con la diferencia de que las entradas aparecen centradas en sentido vertical.
<code>b{Ancho}</code>	Similar al anterior, pero las entradas se justifican a la parte inferior.
<code>!{Objeto}</code>	Nuevo separador de columnas, similar a <code>@{Objeto}</code> . La diferencia está en que no suprime el espacio normal entre columnas.
<code>>{comando}</code>	Se debe incluir antes de un especificador <code>l, r, c, p, m, b</code> . Tiene el efecto de aplicar el comando a cada una de las casillas de la columna.
<code><{comando}</code>	Análogo al anterior, sólo que se utiliza <i>después</i> del especificador de columna, y aplica el comando detrás de cada columna.

Para casillas tipo párrafo (`p{Ancho}`, `m{Ancho}` y `b{Ancho}`), es posible dividir el texto en varias líneas, y justificarlo a cualquiera de los lados utilizando los comandos:

- `\raggedright` —> El texto se ajusta sólo por la izquierda
- `\centering` —> Texto centrado
- `\raggedleft` —> El texto se ajusta sólo por la derecha

Dentro de cada casilla, cada línea se separa utilizando el comando estándar `\backslash`. Es importante tener en cuenta que, para la última casilla de una fila, se debe terminar con el comando `\tabularnewline`, que indica a L^AT_EX que hemos finalizado con esa fila de la tabla.

Ejemplo:

```
\begin{tabular}{|l|c|} \hline
\multicolumn{1}{|p{1.5cm}|}{\centering %
Planeta} & \multicolumn{1}{p{4cm}|}{\centering %
\\Distancia media al sol \\
(millones de km)} \tabularnewline \hline
Mercurio & 58.1 \\
Venus & 108.3 \\
Tierra & 150.0 \\ \hline
\end{tabular}
```

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.0

Cambiando los formatos “p{1.5cm}” y “p{4cm}” por “m{1.5cm}” y “m{4cm}” ó por “b{1.5cm}” y “b{4cm}” obtendríamos, respectivamente:

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.0

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.0

Vemos en los ejemplo anteriores que los puntos de los decimales no quedan bien alineados. Una posible solución sería colocar el signo “.” como separador de columna, de la siguiente forma:

```
\begin{tabular}{|l|r@{.}l|}
\hline
\multicolumn{1}{|m{1.5cm}|}{\centering %
Planeta} & \multicolumn{2}{m{4cm}|}{\centering %
\\Distancia media al sol \\
(millones de km)} \tabularnewline \hline
Mercurio & 58&1 \\
Venus & 108&3 \\
Tierra & 150&1 \\ \hline
\end{tabular}
```

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.1

Sin embargo, tal procedimiento no da buenos resultados para columnas muy anchas. La solución, para este caso, se encuentra en utilizar el paquete **dcolumn**. Este paquete introduce un nuevo separador, de sintaxis:

D{separador1}{separador2}{Num}

donde **separador1** es el decimal que utilizamos en el fichero .tex, y **separador2** es el decimal que LATEX colocará en la salida. **Num** denota el número máximo de decimales en la columna (el valor -1 equivaldría a cualquiera). Por ejemplo, obtendríamos el resultado deseado con:

```

\begin{center}
\begin{tabular}{|l|D{,}{.}{-1}|}
\hline
\multicolumn{1}{|m{1.5cm}|}{\centering %\\
Planeta} & \multicolumn{1}{m{4cm}|}{%\\
\centering Distancia media al sol \\
(millones de km)} \\ \hline
Mercurio & 58,1 \\
Venus & 108,3 \\
Tierra & 150,1 \\
\hline
\end{tabular}
\end{center}

```

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.1

Los argumentos `>{comando}` y `<{comando}` nos permiten hacer útiles manipulaciones por columnas. Por ejemplo, si queremos poner en negrita, añadir color, etc... una columna en concreto, podemos especificar:

```

\begin{center}
\begin{tabular}{|>{\bfseries l}|%
>{\color{red}c|}}
\hline
\multicolumn{1}{|m{1.5cm}|}{\centering %\\
Planeta} & \multicolumn{1}{m{4cm}|}{%\\
\centering Distancia media al sol \\
(millones de km)} \\ \hline
Mercurio & 58.1 \\
Venus & 108.3 \\
Tierra & 150.1 \\
\hline
\end{tabular}
\end{center}

```

Planeta	Distancia media al sol (millones de km)
Mercurio	58.1
Venus	108.3
Tierra	150.1

La utilidad del comando de cierre `<{comando}` puede verse en el siguiente ejemplo:

```

\begin{tabular}{|c|>{$\displaystyle c<{$}}|}
\hline
Función & \text{Definición} \\ \hline
tangente & \frac{\sen(x)}{\cos(x)} \\ \hline
cosecante & \frac{\cos(x)}{\sen(x)} \\ \hline
\end{tabular}

```

Función	Definición
tangente	$\frac{\sen(x)}{\cos(x)}$
cosecante	$\frac{\cos(x)}{\sen(x)}$

donde los contenidos de la segunda columna se escriben automáticamente en modo matemático tipo párrafo.

Si un determinado formato de columna se va a repetir en varias tablas (ó columnas), es conveniente utilizar el siguiente comando, proporcionado por el paquete array, que permite construir nuevos formatos:

```
\newcolumntype{Carácter}{Definición}
```

Así por ejemplo, la expresión `>$\displaystyle c<{$}` en la tabla anterior puede ser abreviada a 'X', utilizando previamente:

```
\newcolumntype{X}{>$\displaystyle c<{$}}
```

En tablas con líneas horizontales, las letras mayúsculas pueden quedar demasiado cerca de las líneas; para solventar ésto el paquete `array` proporciona la longitud `\extrarowheight`, que permite añadir una pequeña altura a cada línea mejorando el resultado, como puede verse en el siguiente ejemplo (donde además, se muestra la diferencia entre los separadores @ y !)

```
\newcolumntype{L}{>{\itshape}r}
\begin{tabular}{|l|L@{---}l|}
\hline
& Nombre & Name \\ \hline
1 & uno & one \\
2 & dos & two \\
3 & tres & three \\ \hline
\end{tabular}
\par \bigskip
\setlength{\extrarowheight}{2pt}
\begin{tabular}{|l|L!{---}l|}
\hline
& Nombre & Name \\ \hline
1 & uno & one \\
2 & dos & two \\
3 & tres & three \\ \hline
\end{tabular}
```

	<i>Nombre</i> —Name
1	<i>uno</i> —one
2	<i>dos</i> —two
3	<i>tres</i> —three

	<i>Nombre</i> — Name
1	<i>uno</i> — one
2	<i>dos</i> — two
3	<i>tres</i> — three

2. El paquete `multirow`

El paquete `multirow` nos permite construir tablas en las que algunas celdas ocupan varias filas dentro de un entorno `tabular`. Se utiliza la orden:

```
\multirow{nrow}{width}[vmove]{contenido}
```

donde: `nrow` es el número de filas a agrupar; `width` es el ancho de la columna; y `vmove` sirve para subir o bajar el texto (opcional). Esta orden funciona de forma similar a `\multicolumn`, pero para filas. Nótese que el formato `m{Ancho}` nos era de utilidad para centrar texto en el caso de columnas vecinas de diferente altura; `multirow` es adecuado para centrar texto con respecto a *varias* columnas vecinas. Por ejemplo:

```
\begin{tabular}{|l|r|r|}
\hline
\multicolumn{4}{c}{Planeta} \\
& \multicolumn{2}{c}{\centering Distancia al sol} \\
& \multicolumn{2}{c}{(millones de km)} \\
& \multicolumn{2}{c}{\centering Maxima} \\
& \multicolumn{2}{c}{\centering Minima} \\
Mercurio & 69.4 & 46.8 \\
Venus & 109.0 & 107.6 \\
Tierra & 152.6 & 147.4 \\
\hline
\end{tabular}
```

Planeta	Distancia al sol (millones de km)	
	Maxima distancia	Minima distancia
Mercurio	69.4	46.8
Venus	109.0	107.6
Tierra	152.6	147.4

Se aprecia que la entrada “Planeta” está justificada a la izquierda. Esto se debe a que el comando `\multirowsetup`, que determina el modo en el que aparece el texto de `\multirow`, está definido por defecto como `\raggedright`. Redefiniéndolo con `\renewcommand*\{\multirowsetup\}{\centering}` se consigue un texto centrado:

```
\renewcommand*\{\multirowsetup\}{\centering}
\begin{tabular}{|l|r|r|}
\hline \multirow{4}{1.8cm}{Planeta} & \multicolumn{2}{c}{Distancia al sol} \\ 
& (millones de km) & \\
& \multicolumn{2}{c}{Maxima distancia} \\ 
& & Minima distancia \\ 
& \multicolumn{2}{c}{Minima distancia} \\ 
& \tabularnewline \hline
Mercurio & 69.4 & 46.8\\ 
Venus & 109.0 & 107.6\\ 
Tierra & 152.6 & 147.4\\ 
\hline
\end{tabular}
```

Planeta	Distancia al sol (millones de km)	
	Maxima distancia	Minima distancia
Mercurio	69.4	46.8
Venus	109.0	107.6
Tierra	152.6	147.4

3. El paquete `hhline`

El paquete `hhline` define el comando `\hhline`, que produce rayas dobles ó simples, y añade capacidades para producir intersecciones bien construidas entre líneas horizontales y verticales. El comando se utiliza como:

```
\hhline{ColumnasEIntersecciones}
```

donde la especificación ColumnasEIntersecciones se hace utilizando los siguientes elementos:

- `=` Una raya horizontal doble del ancho de una columna.
- `-` Una raya horizontal simple del ancho de una columna.
- `~` Una columna sin raya horizontal.
- `||` Una raya vertical que corta a una horizontal (simple ó doble).
- `:|` Una raya vertical que es partida por una horizontal doble. Detrás ó delante debe haber =.
- `#` Dos rayas verticales que cortan a una horizontal doble.
- `t` La semiparte superior de una horizontal doble.
- `b` La semiparte inferior de una horizontal doble.
- `*` Podemos usar la abreviatura: `*{3}{==#}` (por ejemplo), que se expande como `====###` (ésto también es válido para el entorno tabular básico).

Los elementos anteriores se utilizan agrupados, para definir entrelazamientos con las rayas verticales. Por ejemplo:

- |t| → Esquina superior izquierda del cruce de dos rayas dobles (↖)
- :t| → Como el anterior, pero para la esquina superior derecha (↗)
- |b| → Análogo, para esquina inferior izquierda (↙)
- :b| → Análogo, para esquina inferior derecha (↘)
- |: → Una raya vertical seguida de otra vertical que se cruza con una horizontal doble
- :: → Una raya vertical, que se ha cruzado con una horizontal doble, seguida de otra vertical
- :: → Enlace sin cortes entre dos rayas dobles
- || → Raya vertical doble que no es atravesada por las rayas horizontales
- # → Corte de rayas dobles

El ejemplo siguiente ilustra las diferentes posibilidades de intersección:

```
\begin{tabular}{||cc||c|c||}
\hline{|t:==:t:==:t|}\\
a&b&c&d\\\
\hline{:==:|~|~||}\\
1&2&3&4\\\
\hline{#==#~|=#}\\
i&j&k&l\\\
\hline{||--||--||}\\
w&x&y&z\\\
\hline{!b:==:b:==:b!}\\
\end{tabular}
```

a	b	c	d
1	2	3	4
i	j	k	l
w	x	y	z

4. Tablas con color: el paquete `colortbl`

El objetivo del paquete `colortbl` es dar color al fondo de las tablas y a las rayas de separación. Este paquete basa su funcionamiento en los paquetes `color` y `array`, que se cargar automáticamente al cargar `colortbl` (podemos, por tanto, incluir en `colortbl` las opciones de controlador propias de `color`).

Para dar color a filas y columnas, podemos utilizar los comandos:

- `\columncolor[Modelo]{Color}[SepIzq][SepDer]` (columnas)
- `\rowcolor[Modelo]{Color}[SepIzq][SepDer]` (filas)

Debemos hacer aquí un pequeño inciso para explicar la sintaxis extendida del comando `\color`. Podemos utilizar simplemente el comando `\color{NombreColor}`, que permite utilizar hasta 68 colores predefinidos (con la opción `usenames` del paquete `color`), ó construir nuestros propios colores a medida mediante el comando `\color[Modelo]{Especificación-de-color}`. Podemos elegir, para `Modelo`, entre las opciones:

- **rgb** —> Especificación-de-color debe ser una terna de números entre 0 y 1, cada uno de los cuales especifica la cantidad relativa de rojo, verde y azul en la mezcla
- **cmyk** —> Análogo, pero utilizando 4 números que representan valores de cyan, magenta, amarillo y negro
- **gray** —> Un valor entre 0 y 1, especificando la proximidad al blanco ó negro del tono de gris

Todas éstas posibilidades son extensibles a los comandos `\columncolor` y `\rowcolor`, por tanto. Los argumentos (optativos) [SepIzq] y [SepDer] indican las longitudes (a izquierda y derecha, respectivamente) que el fondo de color debe exceder del espacio ocupado por el texto de la celda de la tabla. Por defecto, su valor es `\tabcolsep`, lo cual quiere decir que si no se incluyen, el fondo de color ocupará toda la celda.

Cada una de las dos versiones, para fila ó columna, debe utilizarse de forma diferente:

- `\columncolor` se debe colocar como argumento del especificador de columna `>{...}` del paquete `array` ó en la definición de un nuevo tipo de columna a través de `\newcolumntype`. También podemos incluirlo dentro de un comando `\multicolumn` (para llenar una sola celda, por ejemplo), pero siempre utilizando `>{...}` en el argumento de `\multicolumn`.
- `\rowcolor` debe aparecer SIEMPRE al comienzo de un fila. Debe tenerse en cuenta de que si entrecruzamos declaraciones de color en filas y columnas, la declaración `\rowcolor` prevalece, ya que es la última en ser asignada.

Finalmente, el color de las líneas de separación se controla mediante los comandos:

- `\arrayrulecolor[Modelo]{color}` que fija el color de las líneas
- `\doublerulesepcolor[Modelo]{color}` que fija el color del relleno entre líneas dobles

ADVERTENCIA FINAL: Sólo está asegurada la obtención del resultado correcto en el fichero final `.pdf` (los visores `.dvi` pueden dar resultados algo extraños).

Ejemplos:

```
\begin{tabular}{|>{\columncolor[gray]{0.9}}l|>{\color{white}\columncolor[gray]{0.6}}r|}%
>{\columncolor[gray]{0.7}[0pt]}l|>{\columncolor{yellow}[0.5\tabcolsep]}r|}%
\end{tabular} \hspace{1cm}
\begin{tabular}{|>{\color{yellow}}%}
\end{tabular}
```

alfa	beta
gamma	delta

alfa	beta
gamma	delta

```
\begin{tabular}{||c||c||}
\hhline{|t::t::t|} 
\rowcolor{red} alfa & beta \\
\hhline{|:::|=|} 
\rowcolor{green} gamma & delta \\
\hhline{|b::b::b|} 
\end{tabular}
```

alfa	beta
gamma	delta

5. Paquete **longtable**

Define el entorno **longtable**, que permite construir tablas que abarquen varias páginas.

Ejemplo:

```
\begin{longtable}{|c|c|c|} 
\hline
NIF & Nota & Calificación \\ \hline
45323459J & 7.2 & (NT) \\
etc...
71123261J & 9.5 & (MH) \\ \hline
\end{longtable}
```

NIF	Nota	Calificación
45323459J	7.2	(NT)
71283755Q	6.5	(AP)
82410104V	—	(NP)
61142427T	7.1	(NT)
22413133X	9.0	(SB)
21152619A	—	(NP)
41520126Q	8.0	(NT)
21151160Q	3.6	(SS)
41342139A	7.2	(NT)
19749402V	6.0	(AP)
72186899A	5.5	(AP)
09456969B	7.1	(NT)
51138427W	9.1	(SB)
11335720P	—	(NP)
71276261C	7.0	(NT)
21342465B	6.5	(AP)
12442545F	7.5	(NT)
91451213L	4.5	(SS)
19329402V	6.0	(AP)
72101499A	5.5	(AP)
09303969B	7.1	(NT)
52138427W	9.1	(SB)
11937720P	—	(NP)
71346261C	7.0	(NT)

21936765B	6.5	(AP)
12356545F	7.5	(NT)
91341213L	4.5	(SS)
19349402V	6.0	(AP)
72101099A	5.5	(AP)
09305969B	7.1	(NT)
51138427W	9.1	(SB)
11137720P	—	(NP)
71276261C	7.0	(NT)
21932465B	6.5	(AP)
12392545F	7.5	(NT)
91941213L	4.5	(SS)
71147749P	3.0	(SS)
71126520J	5.0	(AP)
73140778F	7.2	(NT)
71123261J	9.5	(MH)

Veamos ahora como colocar una cabecera común a todas las páginas, otra especial para el principio de la tabla, un pie de tabla general y otro especial para el fin de la tabla (además de leyendas); los textos, ó formatos de encabezamiento de tabla (`multicolumn`, columnas normales, etc...) han de ser colocados respectivamente antes de las marcas `\endfirsthead`, `\endhead`, `\endfoot` y `\endlastfoot`. Todo esto se ilustra en el siguiente ejemplo:

(Más detalles en la documentación del paquete)

```
\begin{longtable}{|c|c|c|}

\caption{Notas finales del primer ejercicio} \\
\hline \multicolumn{3}{|c|}{Notas del grupo C} \\
\hline NIF & Nota & Calificación \\ \hline
\endfirsthead
\caption{Continuación de la tabla} \\
\hline
NIF & Nota & Calificación \\ \hline
\endhead
\multicolumn{3}{c}{(sigue en la página siguiente)} \\
\endfoot
\multicolumn{3}{c}{(Fin de la tabla)} \\
\endlastfoot
45323459J & 7.2 & (NT) \\
etc...
71123261J & 9.5 & (MH) \\ \hline
\end{longtable}
```

Cuadro 2: Notas finales del primer ejercicio

Notas del grupo C		
NIF	Nota	Calificación
45323459J	7.2	(NT)
71283755Q	6.5	(AP)
82410104V	—	(NP)
61142427T	7.1	(NT)
22413133X	9.0	(SB)
21152619A	—	(NP)
41520126Q	8.0	(NT)
21151160Q	3.6	(SS)
41342139A	7.2	(NT)
19749402V	6.0	(AP)
72186899A	5.5	(AP)
09456969B	7.1	(NT)
51138427W	9.1	(SB)
11335720P	—	(NP)
71276261C	7.0	(NT)
21342465B	6.5	(AP)
12442545F	7.5	(NT)
91451213L	4.5	(SS)
19329402V	6.0	(AP)
72101499A	5.5	(AP)
09303969B	7.1	(NT)
52138427W	9.1	(SB)
11937720P	—	(NP)
71346261C	7.0	(NT)
21936765B	6.5	(AP)
12356545F	7.5	(NT)
91341213L	4.5	(SS)
19349402V	6.0	(AP)
72101099A	5.5	(AP)
09305969B	7.1	(NT)
51138427W	9.1	(SB)
11137720P	—	(NP)
71346261C	7.0	(NT)
21936765B	6.5	(AP)
19349402V	6.0	(AP)
72101099A	5.5	(AP)
09305969B	7.1	(NT)
51138427W	9.1	(SB)
11137720P	—	(NP)
71276261C	7.0	(NT)
21932465B	6.5	(AP)
12392545F	7.5	(NT)
91941213L	4.5	(SS)

(sigue en la página siguiente)

Cuadro 2: Continuación de la tabla

NIF	Nota	Calificación
71147749P	3.0	(SS)
71126520J	5.0	(AP)
73140778F	7.2	(NT)
71123261J	9.5	(MH)

(Fin de la tabla)

6. Paquete **tabularx**

Este paquete introduce el entorno `tabularx`, que permite construir tablas con una anchura total predeterminada; para ello se declara la anchura total como argumento del entorno, y se introduce un nuevo formato de columna “X”, equivalente al “`p{Ancho}`”, donde la anchura se calcula automáticamente de forma que la anchura final de la tabla sea la requerida; véanse los siguientes ejemplos:

```
\begin{tabularx}{8cm}{|XcXcXcX|}
```

```
\hline & test1 & & test2 & & test3 & \\
\hline & test4 & & test5 & & test6 & \\
\hline
\end{tabularx}
```

```
\begin{tabularx}{10cm}{|c|X|c|X|}
```

```
\hline
\multicolumn{2}{|c|}{Ejemplo de multicolumna} & TRES & CUATRO \\
\hline uno & La longitud de esta columna depende de la anchura total de la tabla & dos & Esta otra columna tambien variara de anchura, de la misma forma que lo hace la segunda \\
\footnote{Se pueden colocar notas al pie con \footnote{}}\\
\hline
\end{tabularx}
```

test1	test2	test3
test4	test5	test6

Ejemplo de multicolumna		TRES	CUATRO
uno	La longitud de esta columna depende de la anchura total de la tabla	dos	Esta otra columna tambien variara de anchura, de la misma forma que lo hace la segunda ¹

(Ensanchar la segunda tabla, y ver el efecto en los párrafos)

¹Se pueden colocar notas al pie con `\footnote{}`

7. Paquete slashbox

```
\noindent
\begin{tabular}{|c|*{5}{c|}}\hline
\backslash slashbox{Sala}{Fecha} & 31/5 & 1/6 & 2/6 &
3/6 & 4/6 \\ \hline
Aula A4 &&&&\hline
Aula III &&&&\hline
Seminario &&&&\hline
\end{tabular}
```

Fecha Sala	31/5	1/6	2/6	3/6	4/6
Aula A4					
Aula III					
Seminario					

8. Tabla con colores guía

El siguiente ejemplo ilustra las posibilidades de L^AT_EX para programar utilidades interesantes; para una tabla con numerosas columnas, es útil introducir un color de fondo en la mitad de las filas para hacer más fácil la lectura. Definimos un nuevo contador `rowparity`, que lleve la cuenta de la paridad de las filas de la tabla, y a continuación un comando `\X` que en vez de simplemente cambiar de línea efectúe las siguientes acciones:

- Sume 1 al contador `rowparity`
- Salte de línea con `\tabularnewline`
- Decida en función de la paridad de `rowparity` (condicional `ifodd`) si añadir el comando `\rowcolor[gray]{0.9}` ó no

```
\newcounter{rowparity}%
\newcommand{\X}{%
\addtocounter{rowparity}{1}%
\tabularnewline%
\ifodd\value{rowparity}%
\rowcolor[gray]{0.9}\fi}

{\setlength{\arrayrulewidth}{1pt}
\begin{tabular}{|c|ccc|}
\hline
Hola & Pepe & Juan & Jesus \\
\hline
}
```

Hola	Pepe	Juan	Jesus
Hola	Pepe	Juan	Jesus
Hola	Pepe	Juan	Jesus
Hola	Pepe	Juan	Jesus
Hola	Pepe	Juan	Jesus
Hola	Pepe	Juan	Jesus

Apuntes de LATEX

Capítulo 13: El paquete `titlesec`^{*}

1. Formato de unidades de estructura: el comando `\titleformat`

Mediante el paquete `titlesec` es posible cambiar de forma libre el aspecto de las unidades de estructura (capítulo, secciones, subsecciones, etc...) dentro de un documento. Para ello, se utiliza comando `\titleformat` (que deberíamos declarar en el preámbulo) con el siguiente formato:

```
\titleformat{\Comando de Estructura}[Tipo]{Formato}{Etiqueta}{Separación}
{Código anterior}[Código posterior]
```

Donde cada una de las opciones del comando (se debe tener cuidado con la sintaxis del comando, algunas van entre llaves y otras entre corchetes) tiene los siguientes significados:

- **Comando de Estructura:** El comando para la unidad de estructura cuyo formato deseamos cambiar; e.g., `\chapter`, `\section`, etc...
- **Tipo:** La forma básica para el encabezamiento de la unidad de estructura; se dispone de las siguientes opciones:
 - `hang` Formato similar a las clases estándar:

```
\titleformat{\section}[hang]
{\itshape}{\thesection.}{1cm}
{%
\section{Ejemplo tipo hang I}
Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto

\vspace{1cm}
\titleformat{\section}[hang]
{\itshape}{\footnotesize
SECCIÓN \thesection \ --- }{0pt}
{\upshape\bfseries}
\section{Ejemplo tipo hang II}
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto}
```

1. Ejemplo tipo `hang I`

Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto

SECCIÓN 2 — Ejemplo tipo `hang II`

Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto

*La documentación completa del paquete (en inglés) se encuentra en el archivo `titlesec.pdf` disponible en el apartado de apuntes.

- **display** Pone la etiqueta y el texto del título en líneas separadas; nótese como ahora el valor **Separación** es aplicado ahora en dirección vertical:

```
\newcommand{\caja}[1]
{\fbox{\large\bfseries #1}}
\titleformat{\section}[display]
{\filcenter\itshape}{Sección
\thesection.}{1mm}{\caja}
\section{Ejemplo tipo display}
Nótese el truco para enmarcar el
título en una caja; definimos
primero el comando caja, que
depende de una variable, y luego
lo usamos en el campo “Código
anterior”, que asigna a la
variable el texto del título de
la sección. Asimismo, ver cómo
el comando filcenter centra todo.
```

Sección 1.

Ejemplo tipo display

Nótese el truco para enmarcar el título en una caja; definimos primero el comando `caja`, que depende de una variable, y luego lo usamos en el campo “Código anterior”, que asigna a la variable el texto del título de la sección. Asimismo, ver cómo el comando `filcenter` centra todo.

- **runin** Integra el título de sección dentro de la primera línea del párrafo que comienza la sección:

```
\titleformat{\section}[runin]
{\scshape\bfseries}
{\color{blue} \S \thesection .}
{1ex}{\color{blue}\.\quad}
\section{Ejemplo runin}
Texto Texto Texto Texto Texto
```

§1. EJEMPLO RUNIN. Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto Texto Texto Texto Texto Texto
Texto

- **frame** Similar a `display` pero encuadrando el título:

```
\titleformat{\section}[frame]
{\normalfont\filcenter\small
\ SECCIÓN \thesection \ }
{7pt}{\Large\bfseries\filcenter}
\section{Ejemplo tipo frame}
En éste caso, separación (que vale
ahora 7pt) especifica la altura de
la caja que enmarca el título
\vspace{12mm}\newcommand{\cajacol}
[1]{\fcolorbox{black}{gris}
\parbox[c][8mm][c]{0.86\textwidth}
{\Large\bfseries \centerline{\#1}}}
\titleformat{\section}[frame]
{\normalfont\filright%
\footnotesize\fcolorbox{black}{%
\green}{SECCIÓN \thesection \ }}
{9pt}{\filcenter\cajacol}
\section{Ejemplo tipo frame}
Otro ejemplo similar en el que se
cambia la etiqueta a la izquierda con filright (en
vez de filcenter), y la hemos enmarcado
con fcolorbox. Lo mismo hemos hecho con
el título.
Lo mismo hemos hecho con el título.
```

SECCIÓN 1
Ejemplo tipo frame

En éste caso, separación (que vale ahora 7pt) especifica la altura de la caja que enmarca el título

SECCIÓN 2

Ejemplo tipo frame

Otro ejemplo similar en el que se cambia la etiqueta a la izquierda con `filright` (en vez de `filcenter`), y la hemos enmarcado con `fcolorbox`. Lo mismo hemos hecho con el título.

- **leftmargin** y **rightmargin** Colocan el título en los márgenes derecho e izquierdo respectivamente:
 - **block** Formato general que trata al conjunto etiqueta + título como un bloque. Preferible a **hang** para títulos centrados
 - **drop** y **wrap** Se encaja el título en el primer párrafo, ocupando dos líneas. Drop usa una longitud fija para el título y wrap es capaz de partirla en dos líneas.
- **Formato:** Comandos y declaraciones que se aplican tanto a la etiqueta como al texto (ver ejemplos anteriores)
- **Etiqueta:** Longitud de separación entre etiqueta y título de la sección; dependiendo del formato, puede ser una distancia vertical ó horizontal
- **Código anterior:** Código ejecutado inmediatamente antes de la escritura del título; podemos jugar, como se ha visto en ejemplos anteriores, incluyendo un comando dependiente de un argumento; en el argumento (que no especificamos) se pasa el título de la sección.
- **Código posterior:** Código opcional a ejecutar inmediatamente después de la escritura del título, que puede ser ejecutado en modo vertical ó horizontal dependiendo del formato del título.

Es importante mencionar que en las opciones del comando `\titleformat` podemos utilizar los comandos `\filcenter`, `\filright` y `\filleft` para justificar tanto el texto del título como la etiqueta (ó los dos globalmente) al centro, derecha ó izquierda, respectivamente (ver ejemplos anteriores y el ejemplo siguiente)

```
\newcommand{\cajados}[1]{\fbox{%
{\large\bfseries \thesection. #1}}
\titleformat{\section}{block}
{\normalfont\filright}{1em}{\cajados}
\section{Título tipo block}
Ejemplo en donde se utiliza de nuevo el
truco de pasar el título de la sección
como argumento de un comando en la
opción \texttt{código anterior}
\bigskip
\titleformat{\section}{block}{\normalfont
\filleft}{1em}{\cajados}
\section{Título tipo block}
Lo mismo con justificación a la derecha
\bigskip
\titleformat{\section}{display}
{}{\filcenter\bfseries Sección
\thesection.}{0pt}{\titlerule[1pt]
\itshape\fillast}[\titlerule[1pt]]
\section{Título tipo display extendido
a varias líneas, donde se hace uso de la
opción fillast. También se usan comandos
titlerule para las líneas horizontales}
Texto de la sección. Texto de la sección...
```

1. Título tipo block

Ejemplo en donde se utiliza de nuevo el truco de pasar el título de la sección como argumento de un comando en la opción **código anterior**

2. Título tipo block

Lo mismo con justificación a la derecha

Sección 3.

Título tipo display extendido a varias líneas, donde se hace uso de la opción fillast. También se usan comandos titlerule para las líneas horizontales

Texto de la sección. Texto de la sección...

Para dibujar líneas horizontales se dispone del comando `\titlerule[grosor]`; éste comando dispone de la versión con asterisco `\titlerule*[{Objeto}]`, que permite dibujar copias repetidas de un objeto dado. El comando `\titleline[justificación]{Material}` permite introducir material horizontal en argumentos de `\titleformat` que esperan material vertical. La variante `\titleline*[justificación]{Material}` permite introducir el material en una caja de anchura `\titlewidth` (variable de longitud que almacena la anchura del título). Es esencial, a la hora de utilizar ésta variante, cargar el paquete `titlesec` con la opción `calcwidth` en el preámbulo: `\usepackage[calcwidth]{titlesec}`. Véase en el siguiente ejemplo una aplicación práctica de éstos comandos; tómese nota especialmente del uso del parámetro optativo `[justificación]` (con los valores habituales r,l,c) para centrar las líneas:

```
\titleformat{\section}[display]
{\filcenter\normalfont\bfseries\sffamily}
{Sección \huge\thesection}{0pt}
{\titleline*[c]\titlerule[1pt]}
\vspace{2pt}\titleline*[c]
\titlerule*{\tiny\textrm{bullet}}
\vspace{2pt}}[\titleline*[c]
\titlerule*{\tiny\textrm{bullet}}\vspace{2pt}
\titleline*[c]\titlerule[1pt]]
\section{Título tipo display adornado}
Texto de la sección...
```

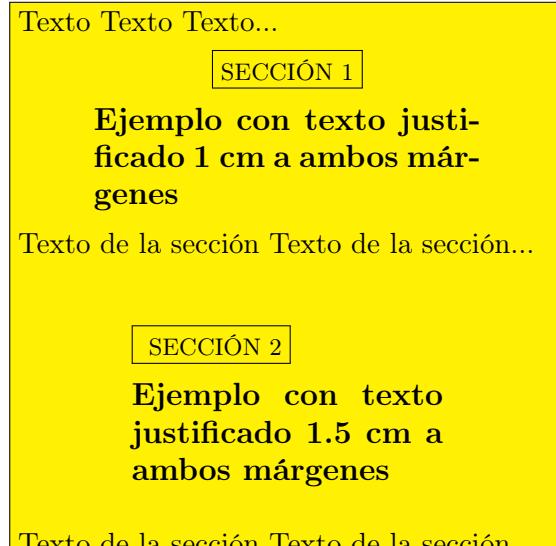


Para especificar una indentación arbitraria en los distintos elementos del título, se utiliza el comando:

```
\titlespacing{\Comando de estructura}{Indentación Izq.}
{Espacio anterior}{Espacio posterior}{Indentación Der.}
```

Donde las opciones de indentación a derecha ó izquierda se pueden utilizar para cambiar la anchura y colocación del título, y las opciones de espacios anterior y posterior especifican los espacios a dejar antes y después del título, respectivamente. Éstas longitudes, de tipo elástico, se especifican a través de la declaración `*f` (siendo `f` un factor decimal), lo cual equivale a `f` unidades “ex” con una cierta elasticidad. Ver los siguientes ejemplos:

```
\titleformat{\section}[display]
{\normalfont}{\filcenter\fbox{\footnotesize SECCIÓN \thesection }}{5pt}{\large\bfseries}
\titlespacing{\section}{1cm}{*1}{*1}{1cm}
Texto Texto Texto...
\section{Ejemplo con texto justificado
1 cm a ambos márgenes}
Texto de la sección Texto de la sección...
\bigskip
\titleformat{\section}[display]{\normalfont}
{\footnotesize\fbox{ SECCIÓN \thesection }}{5pt}{\large\bfseries}
\titlespacing{\section}{1.5cm}{*3}{*3}%
[1.5cm]
\section{Ejemplo con texto justificado
1.5 cm a ambos márgenes}
Texto de la sección Texto de la sección...
```



```
\newcommand{\cajon}[1]{\fbox{%
\parbox{4cm}{\large\bfseries #1}}}
\titleformat{\section}[display]
{\normalfont}{\footnotesize\fbox{%
SECCIÓN \thesection }}{5pt}{\cajon}
\titlespacing{\section}{1cm}{*2}{*2}
\section{Ejemplo con texto
a 1 cm del margen izquierdo}
Notar cómo utilizamos un parbox dentro
de una fbox para encuadrar el título y
especificar su anchura
```

SECCIÓN 1

Ejemplo con texto
a 1 cm del margen
izquierdo

Notar cómo utilizamos un parbox dentro de una fbox para encuadrar el título y especificar su anchura

Otra capacidad interesante del paquete es la especificación condicional de formatos de título para páginas a derecha ó izquierda, utilizando las variables `name` (con valor `\section`, `\subsection`, etc...) y `page` (con valores `even` ó `odd`) en la opción comando de estructura. Para que ésto funcione, es importante que el manuscrito esté formateado con la opción `twoside`

Por ejemplo:

```
\newcommand{\cajon}[1]{\fbox{\parbox{4cm}{\large\bfseries #1}}}
\titleformat{name=\section,page=odd}[display]
{\filleft\normalfont}{\footnotesize\fbox{%
SECCIÓN \thesection }}{5pt}{\cajon}
\titleformat{name=\section,page=even}[display]
{\filright\normalfont}{\footnotesize\fbox{%
SECCIÓN \thesection }}{5pt}{\cajon}
\section{Sección donde alternativamente se justifican
las cajas a los lados derecho e izquierdo}
Texto...
```

colocará los títulos a un lado ó a otro dependiendo de la página

2. Cambiando los encabezamientos de página

Para cambiar el estilo de los encabezamientos y/ó pies de página, debemos en primer lugar declarar en el preámbulo el nombre de uno (o varios) nuevos estilos con el comando:

```
\newpagestyle{Nombre de Estilo}[Estilo Global]{Comandos}
```

Después, podemos utilizarlos en el lugar ó lugares que nos convenga dentro del documento con la intrucción:

```
\pagestyle{Nombre de Estilo}
```

En el parámetro optativo `Estilo Global` podemos incluir comandos generales (tamaño de tipo de letra, negrita, itálica, etc... que afectarán a todas las partes de la cabecera ó pie de página; en el parámetro `Comandos`, se introducen las especificaciones concretas de formateado de encabezamientos. Disponemos de los siguientes comandos:

`\headrule` —> Dibuja una línea horizontal bajo el encabezamiento de página.

`\setheadrule{grosor}` —> Versión análoga donde podemos cambiar el grosor de línea.

`\footrule` —> Dibuja una línea horizontal sobre el pie de página.

`\setfootrule{grosor}` —> Versión análoga donde podemos cambiar el grosor de línea.

Los comandos:

```
\sethead[even-left][even-center][even-right]{odd-left}{odd-center}{odd-right}
\setfoot[even-left][even-center][even-right]{odd-left}{odd-center}{odd-right}
```

se utilizan para introducir los contenidos de las cabeceras ó pies de página, pudiendo especificarse lo que se desea colocar a derecha, izquierda ó en el centro, para páginas pares e impares (si se tiene activada la opción de documento `twoside`).

Como ejemplo, observe el formato del presente documento, obtenido mediante las declaraciones:

```
\newpagestyle{estiloA}[\large\sffamily]{\headrule
\sethead{El paquete titlesec}{\thesection. \ \sectiontitle}{Apuntes de \LaTeX}
\footrule\setfoot{}{\usepage}{}}

\pagestyle{estiloA}
```

donde el comando `\sectiontitle` permite escribir el título de la sección. Es importante observar el caso del título de la sección 1, demasiado largo; para escribir una versión abreviada, se utiliza el comando:

```
\sectionmark{Título abreviado}
```

justo tras el comando `\section{Título completo}`, donde `Título abreviado` será lo que aparezca en el encabezamiento de página.

Podemos especificar libremente la *estructura* (el contenido viene dado por los comandos `\sethead` y `\setfoot`) de los encabezamientos ó pies a través de la redefinición de los comandos `\makeheadrule` y `\makefootrule`, que en su forma estándar vienen definidos por:

```
\renewcommand{\makeheadrule}{\rule[-.3\baselineskip]{\linewidth}{0.4pt}}
```

(conanáloga definición para `\makefootrule`)

Por ejemplo, pruébese el siguiente código en un documento nuevo:

```
\documentclass[a4paper,11pt]{article}

\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[spanish]{babel}
\usepackage{color}
\usepackage[calcwidth]{titlesec}

\newpagestyle{estiloA}[\large]{\headrule
\sethead{Título del Trabajo}{\thesection. \ \sectiontitle}{\thepage}

\pagestyle{estiloA}

\renewcommand{\makeheadrule}{%
\makebox[0pt][l]{\rule[.9\baselineskip]{\linewidth}{0.8pt}}%
\rule[-.4\baselineskip]{\linewidth}{0.8pt}}}

\begin{document}
```

```
\section{Primera sección}
Bla bla bla...
\newpage
\section{Segunda sección}
Bla bla bla...
\newpage
\section{Tercera sección}
Bla bla bla...
\end{document}
```

(es importante tener en cuenta que se debe redefinir `\makeheadrule` DESPUÉS del comando `\pagestyle{Estilo}`).

Ejercicios:

Partiendo de la plantilla CochesYSistemas.tex, formatear los tres ejemplos colgados en el apartado “Ejemplos” de la página web de la asignatura.

3. Tablas de contenidos con titletoc

Continuará...

Presentaciones en \LaTeX con Beamer

Asignatura de LaTeX: Capítulo 13

Luis M. Molina

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX
- Además de poder introducir fórmulas con comodidad, tenemos a nuestra disposición amplias posibilidades para la animación de diapositivas

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX
- Además de poder introducir fórmulas con comodidad, tenemos a nuestra disposición amplias posibilidades para la animación de diapositivas

- $$\int_0^\infty \frac{1}{x^2} = 1$$

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX
- Además de poder introducir fórmulas con comodidad, tenemos a nuestra disposición amplias posibilidades para la animación de diapositivas

- $$\int_0^\infty \frac{1}{x^2} = 1$$

- $$V(x) = A \int_0^\infty \frac{dr}{r} +$$

Dipolo

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX
- Además de poder introducir fórmulas con comodidad, tenemos a nuestra disposición amplias posibilidades para la animación de diapositivas

- $$\int_0^\infty \frac{1}{x^2} = 1$$

- $$V(x) = A \int_0^\infty \frac{dr}{r} + B \int_0^\infty \frac{dr}{r^2}$$

Dipolo Coulomb

Generalidades

¿Qué es Beamer?

- Beamer es una clase de documento especialmente diseñada para presentaciones que utilicen recursos \LaTeX
- Requiere necesariamente la compilación a través de PDF \LaTeX
- Además de poder introducir fórmulas con comodidad, tenemos a nuestra disposición amplias posibilidades para la animación de diapositivas

- $$\int_0^\infty \frac{1}{x^2} = 1$$

- $$V(x) = A \int_0^\infty \frac{dr}{r} + B \int_0^\infty \frac{dr}{r^2} + C \int_0^\infty \left(\frac{1}{r^6} - \frac{1}{r^{12}} \right) dx$$

Dipolo Coulomb Van der Waals

Estructura básica del preámbulo del documento:

```
\documentclass[12pt]{beamer}

\usepackage{Paquete 1}
\usepackage{Paquete 2}
...
\mode{\usetheme{Madrid}}

\title{Ejemplo de Presentación en Beamer (I)}
\author{Asignatura de LaTeX}

\begin{document}

\end{document}
```

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`
- Para cada nueva transparencia, escribimos:
`\begin{frame}[Opciones]`
`\frametitle{Título del frame}`
Material del frame
`\end{frame}`

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`
- Para cada nueva transparencia, escribimos:
`\begin{frame}[Opciones]`
`\frametitle{Título del frame}`
Material del frame
`\end{frame}`
- En el argumento optativo [Opciones] podemos utilizar:

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`
- Para cada nueva transparencia, escribimos:
`\begin{frame}[Opciones]`
`\frametitle{Título del frame}`
Material del frame
`\end{frame}`
- En el argumento optativo [Opciones] podemos utilizar:
 - t, c, b Alineación vertical del contenido del frame

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`
- Para cada nueva transparencia, escribimos:
`\begin{frame}[Opciones]`
`\frametitle{Título del frame}`
Material del frame
`\end{frame}`
- En el argumento optativo [Opciones] podemos utilizar:
 - t, c, b Alineación vertical del contenido del frame
 - plain Elimina encabezamientos, pies y barras laterales.
Útil para frames que incluyan gráficos de grandes dimensiones

Transparencia de título; entornos frame

- Comenzamos colocando el título (tras `\begin{document}`) con:
`\frame{\titlepage}`
- Para cada nueva transparencia, escribimos:
`\begin{frame}[Opciones]`
`\frametitle{Título del frame}`
Material del frame
`\end{frame}`
- En el argumento optativo [Opciones] podemos utilizar:
 - `t`, `c`, `b` Alineación vertical del contenido del frame
 - `plain` Elimina encabezamientos, pies y barras laterales.
Útil para frames que incluyan gráficos de grandes dimensiones
 - `squeeze` Comprime todo lo posible los espacios verticales

Manejo de colores

Paquete color

Para utilizar colores, se pueden emplear los comandos del paquete color (cargado automáticamente por Beamer):

```
\definecolor{Azul}{rgb}{0.1,0.1,0.6}  
\textcolor{Azul}{Prueba de texto Azul}
```

→ Prueba de texto Azul

Manejo de colores

Paquete color

Para utilizar colores, se pueden emplear los comandos del paquete color (cargado automáticamente por Beamer):

```
\definecolor{Azul}{rgb}{0.1,0.1,0.6}  
\textcolor{Azul}{Prueba de texto Azul}
```

→ Prueba de texto Azul

Paquete xcolor

Existe también otra forma de especificar colores a través del paquete xcolor (que carga también beamer):

```
\colorlet{NaranjaA}{yellow!50!red}  
\colorlet{NaranjaB}{yellow!25!red!50!white}  
\textcolor{NaranjaA}{Texto NaranjaA}  
\textcolor{NaranjaB}{Texto NaranjaB}
```

→ Texto NaranjaA Texto NaranjaB

El entorno block

- Mediante el entorno `block`, podemos dibujar recuadros (optionalmente con título) donde incluir nuestros materiales.
Probemos:

El entorno block

- Mediante el entorno `block`, podemos dibujar recuadros (optionalmente con título) donde incluir nuestros materiales.
Probemos:

```
\begin{block}{}  
\begin{itemize}  
  \item Un item  
  \item Otro item  
  \item Otro item más  
\end{itemize}  
\end{block}
```

El entorno block

- Mediante el entorno `block`, podemos dibujar recuadros (optionalmente con título) donde incluir nuestros materiales.
Probemos:

```
\begin{block}{}  
\begin{itemize}  
  \item Un item  
  \item Otro item  
  \item Otro item más  
\end{itemize}  
\end{block}
```

- Cuyo resultado es (ver transparencia siguiente):

El entorno block

- Un item
- Otro item
- Otro item más

El entorno block

- Rellenando el argumento vacío del entorno block, le podemos poner título:

El entorno block

- Rellenando el argumento vacío del entorno block, le podemos poner título:

```
\begin{block}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{block}
```

El entorno block

- Rellenando el argumento vacío del entorno block, le podemos poner título:

```
\begin{block}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{block}
```

- Cuyo resultado es:

Título del bloque

- Un item
- Otro item
- Otro item más

Otros bloques

- Además de `block`, disponemos de `alertblock`:

Otros bloques

- Además de block, disponemos de alertblock:

```
\begin{alertblock}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{alertblock}
```

Otros bloques

- Además de block, disponemos de alertblock:

```
\begin{alertblock}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{alertblock}
```

- Con el resultado:

Título del bloque

- Un item
- Otro item
- Otro item más

Otros bloques

- También puede usarse exampleblock:

Otros bloques

- También puede usarse exampleblock:

```
\begin{exampleblock}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{exampleblock}
```

Otros bloques

- También puede usarse exampleblock:

```
\begin{exampleblock}{Título del bloque}
\begin{itemize} \item Un item \item Otro item
\item Otro item más \end{itemize}
\end{exampleblock}
```

- Con el resultado:

Título del bloque

- Un item
- Otro item
- Otro item más

Cajas coloreadas

- Con el entorno beamercolorbox podemos meter nuestro material en una caja coloreada con múltiples opciones.

Cajas coloreadas

- Con el entorno beamercolorbox podemos meter nuestro material en una caja coloreada con múltiples opciones.
- Definamos en el preámbulo el color “postit” como:
`\setbeamercolor{postit}{bg=yellow!50!white}`

Cajas coloreadas

- Con el entorno beamercolorbox podemos meter nuestro material en una caja coloreada con múltiples opciones.
- Definamos en el preámbulo el color “postit” como:
`\setbeamercolor{postit}{bg=yellow!50!white}`
- Probemos entonces:

Cajas coloreadas

- Con el entorno beamercolorbox podemos meter nuestro material en una caja coloreada con múltiples opciones.
- Definamos en el preámbulo el color “postit” como:
`\setbeamercolor{postit}{bg=yellow!50!white}`
- Probemos entonces:

Cajas coloreadas

- Con el entorno beamercolorbox podemos meter nuestro material en una caja coloreada con múltiples opciones.
- Definamos en el preámbulo el color “postit” como:
`\setbeamercolor{postit}{bg=yellow!50!white}`
- Probemos entonces:

```
\begin{beamercolorbox}{postit}  
Texto diverso  
\end{beamercolorbox}
```

- Que produce:

Texto diverso

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)
- `left right center` (Alineación del material en la caja)

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)
- `left right center` (Alineación del material en la caja)
- `sep=Distancia` (Separación entre texto y marco de la caja)

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)
- `left right center` (Alineación del material en la caja)
- `sep=Distancia` (Separación entre texto y marco de la caja)
- `rounded=true (false)` (Dibuja las esquinas redondeadas)

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)
- `left right center` (Alineación del material en la caja)
- `sep=Distancia` (Separación entre texto y marco de la caja)
- `rounded=true (false)` (Dibuja las esquinas redondeadas)
- `shadow=true (false)` (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Opciones del entorno beamercolorbox

- wd={Anchura} dp={Profundidad} ht={Altura}
(Dimensiones de la caja)
- left right center (Alineación del material en la caja)
- sep=Distancia (Separación entre texto y marco de la caja)
- rounded=true (false) (Dibuja las esquinas redondeadas)
- shadow=true (false) (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Ejemplos:

Opciones del entorno beamercolorbox

- wd={Anchura} dp={Profundidad} ht={Altura}
(Dimensiones de la caja)
- left right center (Alineación del material en la caja)
- sep=Distancia (Separación entre texto y marco de la caja)
- rounded=true (false) (Dibuja las esquinas redondeadas)
- shadow=true (false) (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Ejemplos:

```
\begin{beamercolorbox}
[sep=3mm]{postit}
Texto diverso Texto diverso
\end{beamercolorbox}
```

Texto diverso Texto
diverso

Opciones del entorno beamercolorbox

- `wd={Anchura} dp={Profundidad} ht={Altura}`
(Dimensiones de la caja)
- `left right center` (Alineación del material en la caja)
- `sep=Distancia` (Separación entre texto y marco de la caja)
- `rounded=true (false)` (Dibuja las esquinas redondeadas)
- `shadow=true (false)` (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Ejemplos:

```
\begin{beamercolorbox}
[wd={3cm},sep=1mm]{postit}
Texto diverso Texto diverso
\end{beamercolorbox}
```

Texto diverso
Texto diverso

Opciones del entorno beamercolorbox

- wd={Anchura} dp={Profundidad} ht={Altura}
(Dimensiones de la caja)
- left right center (Alineación del material en la caja)
- sep=Distancia (Separación entre texto y marco de la caja)
- rounded=true (false) (Dibuja las esquinas redondeadas)
- shadow=true (false) (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Ejemplos:

```
\begin{beamercolorbox}
[wd={4cm},sep=1mm,center,
rounded=true]{postit}
Texto diverso
\end{beamercolorbox}
```

Texto diverso

Opciones del entorno beamercolorbox

- wd={Anchura} dp={Profundidad} ht={Altura}
(Dimensiones de la caja)
- left right center (Alineación del material en la caja)
- sep=Distancia (Separación entre texto y marco de la caja)
- rounded=true (false) (Dibuja las esquinas redondeadas)
- shadow=true (false) (Coloca una sombra –
Ésta opción sólo tiene efecto si está activada la opción rounded)

Ejemplos:

```
\begin{beamercolorbox}[wd={4cm},  
sep=1mm,center,rounded=true,  
shadow=true]{postit}  
Texto diverso  
\end{beamercolorbox}
```

Texto diverso

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)
- El comando más básico es \pause, que coloca una pausa:

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)
- El comando más básico es \pause, que coloca una pausa:

Ejemplo:

```
\begin{itemize}
  \item Primer elemento \pause
  \item Segundo elemento \pause
  \item Tercer elemento
\end{itemize}
```

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)
- El comando más básico es \pause, que coloca una pausa:

Ejemplo:

```
\begin{itemize}
\item Primer elemento \pause
\item Segundo elemento \pause
\item Tercer elemento
\end{itemize}
```

- Primer elemento

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)
- El comando más básico es \pause, que coloca una pausa:

Ejemplo:

```
\begin{itemize}
\item Primer elemento \pause
\item Segundo elemento \pause
\item Tercer elemento
\end{itemize}
```

- Primer elemento
- Segundo elemento

Animaciones: Generalidades

- Dentro de cada frame, podemos introducir efectos de animación (también llamados *overlays*)
- El comando más básico es \pause, que coloca una pausa:

Ejemplo:

```
\begin{itemize}
\item Primer elemento \pause
\item Segundo elemento \pause
\item Tercer elemento
\end{itemize}
```

- Primer elemento
- Segundo elemento
- Tercer elemento

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es `<Rango>`, donde Rango especifica en qué pasos de la animación tiene efecto el comando:

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es `<Rango>`, donde Rango especifica en qué pasos de la animación tiene efecto el comando:
 - 1- → Del primer paso en adelante

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es <Rango>, donde Rango especifica en qué pasos de la animación tiene efecto el comando:
 - 1- → Del primer paso en adelante
 - -3 → Hasta el paso número 3

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es `<Rango>`, donde Rango especifica en qué pasos de la animación tiene efecto el comando:
 - `1-` → Del primer paso en adelante
 - `-3` → Hasta el paso número 3
 - `2-5` → Del paso 2 al 5 (ambos inclusive)

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es `<Rango>`, donde Rango especifica en qué pasos de la animación tiene efecto el comando:
 - `1-` → Del primer paso en adelante
 - `-3` → Hasta el paso número 3
 - `2-5` → Del paso 2 al 5 (ambos inclusive)
 - `1-3,5` → Del paso 1 al 3, así como el 5

Especificaciones de animación

Sintaxis

- Para todos los comandos (ver más adelante) que admiten especificaciones de animación, la sintaxis de las mismas es `<Rango>`, donde Rango especifica en qué pasos de la animación tiene efecto el comando:
 - 1- → Del primer paso en adelante
 - -3 → Hasta el paso número 3
 - 2-5 → Del paso 2 al 5 (ambos inclusive)
 - 1-3,5 → Del paso 1 al 3, así como el 5
- La sintaxis general de los comandos que admiten efectos de animación es `\Comando<Rango>\{Contenido}`

Comandos de animación: \onslide

El comando `\onslide<Rango>{Material}` muestra Material en los pasos especificados en `<Rango>`. Debe tenerse en cuenta que, cuando NO se muestre el material, éste seguirá ocupando espacio.

Ejemplo:

```
\onslide<1->{Éste texto se muestra desde el primer paso} \\  
\onslide<2>{Éste en el paso 2} \\  
\onslide<2-3>{Y éste en los pasos 2 y 3} \\  
\onslide<4->{Ésto se muestra a partir del cuarto} \\  
Ésto se muestra en todos los pasos
```

Éste texto se muestra desde el primer paso

Ésto se muestra en todos los pasos

Comandos de animación: \onslide

El comando `\onslide<Rango>{Material}` muestra Material en los pasos especificados en `<Rango>`. Debe tenerse en cuenta que, cuando NO se muestre el material, éste seguirá ocupando espacio.

Ejemplo:

```
\onslide<1->{Éste texto se muestra desde el primer paso} \\  
\onslide<2>{Éste en el paso 2} \\  
\onslide<2-3>{Y éste en los pasos 2 y 3} \\  
\onslide<4->{Ésto se muestra a partir del cuarto} \\  
Ésto se muestra en todos los pasos
```

Éste texto se muestra desde el primer paso

Éste en el paso 2

Y éste en los pasos 2 y 3

Ésto se muestra en todos los pasos

Comandos de animación: \onslide

El comando `\onslide<Rango>{Material}` muestra Material en los pasos especificados en `<Rango>`. Debe tenerse en cuenta que, cuando NO se muestre el material, éste seguirá ocupando espacio.

Ejemplo:

```
\onslide<1->{Éste texto se muestra desde el primer paso} \\  
\onslide<2>{Éste en el paso 2} \\  
\onslide<2-3>{Y éste en los pasos 2 y 3} \\  
\onslide<4->{Ésto se muestra a partir del cuarto} \\  
Ésto se muestra en todos los pasos
```

Éste texto se muestra desde el primer paso

Y éste en los pasos 2 y 3

Ésto se muestra en todos los pasos

Comandos de animación: \onslide

El comando `\onslide<Rango>{Material}` muestra Material en los pasos especificados en `<Rango>`. Debe tenerse en cuenta que, cuando NO se muestre el material, éste seguirá ocupando espacio.

Ejemplo:

```
\onslide<1->{Éste texto se muestra desde el primer paso} \\  
\onslide<2>{Éste en el paso 2} \\  
\onslide<2-3>{Y éste en los pasos 2 y 3} \\  
\onslide<4->{Ésto se muestra a partir del cuarto} \\  
Ésto se muestra en todos los pasos
```

Éste texto se muestra desde el primer paso

Ésto se muestra a partir del cuarto

Ésto se muestra en todos los pasos

Comandos de animación: \only

El comando `\only<Rango>{Material}` actúa de forma similar a `\onslide`, con la importante diferencia que el texto no mostrado *no ocupa espacio*. Podemos utilizarlo por tanto para hacer substituciones de objetos de forma animada.

Ejemplo:

```
\onslide<1->{Este texto se muestra desde el primer paso} \\  
\only<2>{Este sólo en el paso 2}  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Esto en el cuarto} \\
```

Este texto se muestra desde el primer paso

Comandos de animación: \only

El comando `\only<Rango>{Material}` actúa de forma similar a `\onslide`, con la importante diferencia que el texto no mostrado *no ocupa espacio*. Podemos utilizarlo por tanto para hacer substituciones de objetos de forma animada.

Ejemplo:

```
\onslide<1->{Este texto se muestra desde el primer paso} \\  
\only<2>{Este sólo en el paso 2}  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Este en el cuarto} \\
```

Este texto se muestra desde el primer paso
Este sólo en el paso 2

Comandos de animación: \only

El comando `\only<Rango>{Material}` actúa de forma similar a `\onslide`, con la importante diferencia que el texto no mostrado *no ocupa espacio*. Podemos utilizarlo por tanto para hacer substituciones de objetos de forma animada.

Ejemplo:

```
\onslide<1->{Este texto se muestra desde el primer paso} \\  
\only<2>{Este sólo en el paso 2}  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Esto en el cuarto} \\
```

Este texto se muestra desde el primer paso
Este del 3 en adelante

Comandos de animación: \only

El comando `\only<Rango>{Material}` actúa de forma similar a `\onslide`, con la importante diferencia que el texto no mostrado *no ocupa espacio*. Podemos utilizarlo por tanto para hacer substituciones de objetos de forma animada.

Ejemplo:

```
\onslide<1->{Éste texto se muestra desde el primer paso} \\  
\only<2>{Éste sólo en el paso 2}  
\onslide<3->{Éste del 3 en adelante} \\  
\onslide<4>{Ésto en el cuarto} \\
```

Éste texto se muestra desde el primer paso
Éste del 3 en adelante
Ésto en el cuarto

Efectos de transparencia

\setbeamercovered

La instrucción `\setbeamercovered{Opcion}` permite variar el modo de visualización del texto no mostrado en las animaciones. Se suele colocar en el preámbulo, dentro del comando `\mode{}`

Tenemos las opciones:

- `invisible` El texto oculto desaparece totalmente (defecto)
- `transparent` El texto oculto aparece como “cuasi-transparente”. Podemos variar el porcentaje de opacidad especificando `transparent=X`, con $0 \leq X \leq 100$
- `dynamic` A medida que pasa el tiempo, el texto no descubierto va variando su grado de visibilidad

(Como ejemplo, en éste frame de la presentación se ha cambiado el valor de `\setbeamercovered` de `invisible` a `transparent`)

Efectos de transparencia

\setbeamercolor{background canvas}{white}

La instrucción `\setbeamercolor{background canvas}{white}` permite variar el modo de visualización del texto no mostrado en las animaciones. Se suele colocar en el preámbulo, dentro del comando `\mode{}`

Tenemos las opciones:

- `invisible` El texto oculto desaparece totalmente (defecto)
- `transparent` El texto oculto aparece como “cuasi-transparente”. Podemos variar el porcentaje de opacidad especificando `transparent=X`, con $0 \leq X \leq 100$
- `dynamic` A medida que pasa el tiempo, el texto no descubierto va variando su grado de visibilidad

(Como ejemplo, en éste frame de la presentación se ha cambiado el valor de `\setbeamercolor{background canvas}{white}` de `invisible` a `transparent`)

Efectos de transparencia

\setbeamercovered

La instrucción `\setbeamercovered{Opcion}` permite variar el modo de visualización del texto no mostrado en las animaciones. Se suele colocar en el preámbulo, dentro del comando `\mode{}`

Tenemos las opciones:

- `invisible` El texto oculto desaparece totalmente (defecto)
- `transparent` El texto oculto aparece como “cuasi-transparente”. Podemos variar el porcentaje de opacidad especificando `transparent=X`, con $0 \leq X \leq 100$
- `dynamic` A medida que pasa el tiempo, el texto no descubierto va variando su grado de visibilidad

(Como ejemplo, en éste frame de la presentación se ha cambiado el valor de `\setbeamercovered` de `invisible` a `transparent`)

Efectos de transparencia

\setbeamercovered

La instrucción `\setbeamercovered{Opcion}` permite variar el modo de visualización del texto no mostrado en las animaciones. Se suele colocar en el preámbulo, dentro del comando `\mode{}`

Tenemos las opciones:

- `invisible` El texto oculto desaparece totalmente (defecto)
- `transparent` El texto oculto aparece como “cuasi-transparente”. Podemos variar el porcentaje de opacidad especificando `transparent=X`, con $0 \leq X \leq 100$
- `dynamic` A medida que pasa el tiempo, el texto no descubierto va variando su grado de visibilidad

(Como ejemplo, en éste frame de la presentación se ha cambiado el valor de `\setbeamercovered` de `invisible` a `transparent`)

Comandos de animación: \visible

En el caso de que utilicemos la opción `transparent` de `\setbeamercolor{covered}`, puede haber situaciones puntuales en las que nos interese que el material sea totalmente invisible en los pasos donde no se muestre. Para ello disponemos el comando `\visible<Rango>{Material}`

Ejemplo:

```
\onslide<1>{Éste texto se muestra solo el primer paso} \\  
\visible<2-3>{Éste en los pasos 2 y 3 (desaparece en el resto)} \\  
\onslide<3->{Éste del 3 en adelante} \\  
\onslide<4>{Ésto en el cuarto} \\
```

Este texto se muestra solo el primer paso

Este del 3 en adelante

Ésto en el cuarto

Comandos de animación: \visible

En el caso de que utilicemos la opción `transparent` de `\setbeamercolor{background canvas}{white}`, puede haber situaciones puntuales en las que nos interese que el material sea totalmente invisible en los pasos donde no se muestre. Para ello disponemos el comando `\visible<Rango>{Material}`

Ejemplo:

```
\onslide<1>{Este texto se muestra solo el primer paso} \\  
\visible<2-3>{Este en los pasos 2 y 3 (desaparece en el resto)} \\  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Esto en el cuarto} \\
```

Este texto se muestra solo el primer paso

Este en los pasos 2 y 3 (desaparece en el resto)

Este del 3 en adelante

Esto en el cuarto

Comandos de animación: \visible

En el caso de que utilicemos la opción `transparent` de `\setbeamercolor{background canvas}{white}`, puede haber situaciones puntuales en las que nos interese que el material sea totalmente invisible en los pasos donde no se muestre. Para ello disponemos el comando `\visible<Rango>{Material}`

Ejemplo:

```
\onslide<1>{Este texto se muestra solo el primer paso} \\  
\visible<2-3>{Este en los pasos 2 y 3 (desaparece en el resto)} \\  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Esto en el cuarto} \\
```

Este texto se muestra solo el primer paso

Este en los pasos 2 y 3 (desaparece en el resto)

Este del 3 en adelante

Esto en el cuarto

Comandos de animación: \visible

En el caso de que utilicemos la opción `transparent` de `\setbeamercolor{background canvas}{white}`, puede haber situaciones puntuales en las que nos interese que el material sea totalmente invisible en los pasos donde no se muestre. Para ello disponemos el comando `\visible<Rango>{Material}`

Ejemplo:

```
\onslide<1>{Este texto se muestra solo el primer paso} \\  
\visible<2-3>{Este en los pasos 2 y 3 (desaparece en el resto)} \\  
\onslide<3->{Este del 3 en adelante} \\  
\onslide<4>{Esto en el cuarto} \\
```

Este texto se muestra solo el primer paso

Este del 3 en adelante

Esto en el cuarto

Efectos de animación incrementales (I)

En el caso de que en entornos `itemize` ó `enumerate` queramos mostrar los ítems de modo progresivo, podemos conseguirlo del siguiente modo:

```
\begin{itemize}
  \item<1-> Primer ítem
  \item<2-> Segundo ítem
  \item<3-> Tercer ítem
\end{itemize}
```

- Primer ítem

Efectos de animación incrementales (I)

En el caso de que en entornos `itemize` ó `enumerate` queramos mostrar los ítems de modo progresivo, podemos conseguirlo del siguiente modo:

```
\begin{itemize}
  \item<1-> Primer ítem
  \item<2-> Segundo ítem
  \item<3-> Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem

Efectos de animación incrementales (I)

En el caso de que en entornos `itemize` ó `enumerate` queramos mostrar los ítems de modo progresivo, podemos conseguirlo del siguiente modo:

```
\begin{itemize}
  \item<1-> Primer ítem
  \item<2-> Segundo ítem
  \item<3-> Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem
- Tercer ítem

Efectos de animación incrementales (I)

En el caso de que en entornos itemize ó enumerate queramos mostrar los ítems de modo progresivo, podemos conseguirlo del siguiente modo:

```
\begin{itemize}
  \item<1-> Primer ítem
  \item<2-> Segundo ítem
  \item<3-> Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem
- Tercer ítem

Ya que el comando \item admite especificaciones de animación

Efectos de animación incrementales (II)

Sin embargo, una forma más sencilla sería la siguiente:

```
\begin{itemize}[<+->]
\item Primer ítem
\item Segundo ítem
\item Tercer ítem
\end{itemize}
```

- Primer ítem

Efectos de animación incrementales (II)

Sin embargo, una forma más sencilla sería la siguiente:

```
\begin{itemize}[<+->]
\item Primer ítem
\item Segundo ítem
\item Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem

Efectos de animación incrementales (II)

Sin embargo, una forma más sencilla sería la siguiente:

```
\begin{itemize}[<+->]
\item Primer ítem
\item Segundo ítem
\item Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem
- Tercer ítem

Efectos de animación incrementales (II)

Sin embargo, una forma más sencilla sería la siguiente:

```
\begin{itemize}[<+->]  
  \item Primer ítem  
  \item Segundo ítem  
  \item Tercer ítem  
\end{itemize}
```

- Primer ítem
- Segundo ítem
- Tercer ítem

Que anima automáticamente todos los ítems del entorno itemize (lo mismo valdría para el enumerate)

Efectos de animación incrementales (III)

Si deseamos que, además, el ítem que se muestra en cada paso aparezca resaltado, podemos usar lo siguiente:

```
\begin{itemize}[<+-| alert@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Primer ítem

Efectos de animación incrementales (III)

Si deseamos que, además, el ítem que se muestra en cada paso aparezca resaltado, podemos usar lo siguiente:

```
\begin{itemize}[<+-| alert@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem

Efectos de animación incrementales (III)

Si deseamos que, además, el ítem que se muestra en cada paso aparezca resaltado, podemos usar lo siguiente:

```
\begin{itemize}[<+-| alert@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem
- **Tercer ítem**

Efectos de animación incrementales (III)

Si deseamos que, además, el ítem que se muestra en cada paso aparezca resaltado, podemos usar lo siguiente:

```
\begin{itemize}[<+-| alert@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Primer ítem
- Segundo ítem
- Tercer ítem

(podemos también usar el comando `\alert<Rango>{Material}` en cualquier situación para resaltar cualquier tipo de contenido)

Especificaciones de acción

La sintaxis <+-| alert@+> puede utilizarse con otros comandos:

- **uncover** (comando equivalente a `onslide`); opción por defecto
- **only** Coloca texto sin ocupar espacio
- **visible** Texto invisible si no es mostrado

Probemos por ejemplo:

```
\begin{itemize}[<+-| only@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Un ítem

Especificaciones de acción

La sintaxis `<+-| alert@+>` puede utilizarse con otros comandos:

- `uncover` (comando equivalente a `onslide`); opción por defecto
- `only` Coloca texto sin ocupar espacio
- `visible` Texto invisible si no es mostrado

Probemos por ejemplo:

```
\begin{itemize}[<+-| only@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- Otro ítem

Especificaciones de acción

La sintaxis <+-| alert@+> puede utilizarse con otros comandos:

- **uncover** (comando equivalente a `onslide`); opción por defecto
- **only** Coloca texto sin ocupar espacio
- **visible** Texto invisible si no es mostrado

Probemos por ejemplo:

```
\begin{itemize}[<+-| only@+>]
  \item Primer ítem
  \item Segundo ítem
  \item Tercer ítem
\end{itemize}
```

- El último ítem

Especificaciones de acción

La sintaxis `<+-| alert@+>` puede utilizarse con otros comandos:

- `uncover` (comando equivalente a `onslide`); opción por defecto
- `only` Coloca texto sin ocupar espacio
- `visible` Texto invisible si no es mostrado

Probemos por ejemplo:

```
\begin{itemize}[<+-| only@+>]
\item Primer ítem
\item Segundo ítem
\item Tercer ítem
\end{itemize}
```

(nótese que hay un pequeño desplazamiento indeseado tras el primer ítem debido a las peculiaridades de espaciado vertical del entorno `itemize`)

Comandos con efectos de animación

Los siguientes comandos \LaTeX admiten efectos de animación:

- $\text{\textbf{}}<\text{Rango}>\{\text{Texto}\}$
- $\text{\color{Color}}<\text{Rango}>\{\text{Color}\}$ (ó $\text{\textcolor{Color}{}}<\text{Rango}>\{\text{Color}\}\{\text{Texto}\}$)
- $\text{\includegraphics{archivo}}$

En el siguiente ejemplo se ilustra su uso:

```
\begin{itemize}\sepa[-6pt]
\item<1-> \textbf{Primer item}
\item<2-> \textcolor{blue}{Segundo item}
\item<3-> Tercer item \includegraphics[height=4mm]{Links.jpg}
\end{itemize}
```

- **Primer item**

Comandos con efectos de animación

Los siguientes comandos \LaTeX admiten efectos de animación:

- $\text{\textbf{}}<\text{Rango}>\{\text{Texto}\}$
- $\text{\color{Color}}<\text{Rango}>\{\text{Color}\}$ (ó $\text{\textcolor{Color}{}}<\text{Rango}>\{\text{Color}\}\{\text{Texto}\}$)
- $\text{\includegraphics{archivo}}$

En el siguiente ejemplo se ilustra su uso:

```
\begin{itemize}\sepa[-6pt]
\item<1-> \textbf{\item} Primer item
\item<2-> \textcolor{blue}{\item} Segundo item
\item<3-> Tercer item \includegraphics[4][height=4mm]{Links.jpg}
\end{itemize}
```

- Primer item
- Segundo item

Comandos con efectos de animación

Los siguientes comandos \LaTeX admiten efectos de animación:

- $\text{\textbf{}}<\text{Rango}>\{\text{Texto}\}$
- $\text{\color{Color}}<\text{Rango}>\{\text{Color}\}$ (ó $\text{\textcolor{Color}{}}<\text{Rango}>\{\text{Color}\}\{\text{Texto}\}$)
- $\text{\includegraphics{archivo}}$

En el siguiente ejemplo se ilustra su uso:

```
\begin{itemize}\sepa[-6pt]
\item<1-> \textbf{Primer item}
\item<2-> \textcolor{blue}{Segundo item}
\item<3-> Tercer item \includegraphics[height=4mm]{Links.jpg}
\end{itemize}
```

- Primer item
- Segundo item
- Tercer item

Comandos con efectos de animación

Los siguientes comandos \LaTeX admiten efectos de animación:

- $\text{\textbf{}}<\text{Rango}>\{\text{Texto}\}$
- $\text{\color{Color}}<\text{Rango}>\{\text{Color}\}$ (ó $\text{\textcolor{Color}{}}<\text{Rango}>\{\text{Color}\}\{\text{Texto}\}$)
- $\text{\includegraphics{archivo}}$

En el siguiente ejemplo se ilustra su uso:

```
\begin{itemize}\sepa[-6pt]
\item<1-> \textbf{\item} Primer item
\item<2-> \textcolor{blue}{\item} Segundo item
\item<3-> Tercer item \includegraphics[4][height=4mm]{Links.jpg}
\end{itemize}
```

- Primer item
- Segundo item
- Tercer item 

Varias columnas

Para dividir el frame en varias columnas, se emplea el entorno `columns` (cuidado con la “s” final). Dentro de éste entorno, podemos colocar cuantas columnas deseemos, de anchura variable, con el entorno `column` (sin “s”). La sintaxis de éste entorno es: `\begin{column}{Ancho} ... \end{column}`. Véase el siguiente ejemplo:

```
\begin{block}{Introducción} En ésta... \end{block}
\begin{columns} \begin{column}{5cm} \vspace*{1cm}
\begin{block}{Resultados} \small \begin{itemize}
\item<2-> |alert@2> Comentario a la Figura 1
\item<3-> |alert@3> Comentario a la Figura 2
\item<4-> |alert@4> Comentario a la Figura 3
\end{itemize} \end{block} \end{column}
\begin{column}{5cm} \begin{center}
\only<2>{\includegraphics[height=3.5cm]{knuth.jpg}}%
\only<3>{\includegraphics[height=3.5cm]{forges.jpg}}%
\only<4>{\includegraphics[height=3.5cm]{kill-bill.jpg}}%
\end{center} \end{column}
\end{columns}
```

Varias columnas (II)

Introducción

En ésta transparencia coordinamos efectos de animación en varias columnas. Nótese cómo la anchura de los entornos block se ajusta al ancho de columna

Resultados

Varias columnas (II)

Introducción

En ésta transparencia coordinamos efectos de animación en varias columnas. Nótese cómo la anchura de los entornos block se ajusta al ancho de columna

Resultados

- Comentario a la Figura 1



Varias columnas (II)

Introducción

En ésta transparencia coordinamos efectos de animación en varias columnas. Nótese cómo la anchura de los entornos block se ajusta al ancho de columna

Resultados

- Comentario a la Figura 1
- Comentario a la Figura 2



Varias columnas (II)

Introducción

En ésta transparencia coordinamos efectos de animación en varias columnas. Nótese cómo la anchura de los entornos block se ajusta al ancho de columna

Resultados

- Comentario a la Figura 1
- Comentario a la Figura 2
- Comentario a la Figura 3



Estilos de presentación (I)

Para variar la apariencia de la presentación, tenemos comandos (a incluir en el argumento de `\mode{...}`) que activan estilos predefinidos de estructura de frame, colores, tipos de letra, etc... La guía de usuario de beamer proporciona ejemplos del empleo de los diversos estilos en los capítulos 15 a 18.

`\usetheme{Nombre del Tema}`

- Temas sin barra de navegación:

`default`, `boxes`, `Bergen`, `Boadilla`, `Madrid` (usado en ésta presentación),
`Ann Arbor`, `CambridgeUS`, `Pittsburgh`, `Rochester`,

Estilos de presentación (I)

Para variar la apariencia de la presentación, tenemos comandos (a incluir en el argumento de `\mode{...}`) que activan estilos predefinidos de estructura de frame, colores, tipos de letra, etc... La guía de usuario de beamer proporciona ejemplos del empleo de los diversos estilos en los capítulos 15 a 18.

`\usetheme{Nombre del Tema}`

- Temas sin barra de navegación:
default, boxes, Bergen, Boadilla, Madrid (usado en ésta presentación),
Ann Arbor, CambridgeUS, Pittsburgh, Rochester,
- Temas con barra de navegación tipo árbol:
Antibes, JuanLesPins

Estilos de presentación (I)

Para variar la apariencia de la presentación, tenemos comandos (a incluir en el argumento de `\mode{...}`) que activan estilos predefinidos de estructura de frame, colores, tipos de letra, etc... La guía de usuario de beamer proporciona ejemplos del empleo de los diversos estilos en los capítulos 15 a 18.

`\usetheme{Nombre del Tema}`

- Temas sin barra de navegación:
`default`, `boxes`, `Bergen`, `Boadilla`, `Madrid` (usado en ésta presentación),
`Ann Arbor`, `CambridgeUS`, `Pittsburgh`, `Rochester`,
- Temas con barra de navegación tipo árbol:
`Antibes`, `JuanLesPins`
- Temas con tabla de contenidos lateral:
`Berkeley`, `PaloAlto`, `Goettingen`, `Marburg`, `Hannover`

Estilos de presentación (I)

Para variar la apariencia de la presentación, tenemos comandos (a incluir en el argumento de `\mode{...}`) que activan estilos predefinidos de estructura de frame, colores, tipos de letra, etc... La guía de usuario de beamer proporciona ejemplos del empleo de los diversos estilos en los capítulos 15 a 18.

`\usetheme{Nombre del Tema}`

- Temas sin barra de navegación:
default, boxes, Bergen, Boadilla, Madrid (usado en ésta presentación),
Ann Arbor, CambridgeUS, Pittsburgh, Rochester,
- Temas con barra de navegación tipo árbol:
Antibes, JuanLesPins
- Temas con tabla de contenidos lateral:
Berkeley, Palo Alto, Goettingen, Marburg, Hannover
- Temas con navegación en miniframe:
Berlin, Ilmenau, Dresden, Darmstadt, Frankfurt, Singapore, Szeged

Estilos de presentación (I)

Para variar la apariencia de la presentación, tenemos comandos (a incluir en el argumento de `\mode{...}`) que activan estilos predefinidos de estructura de frame, colores, tipos de letra, etc... La guía de usuario de beamer proporciona ejemplos del empleo de los diversos estilos en los capítulos 15 a 18.

`\usetheme{Nombre del Tema}`

- Temas sin barra de navegación:
default, boxes, Bergen, Boadilla, Madrid (usado en ésta presentación),
Ann Arbor, CambridgeUS, Pittsburgh, Rochester,
- Temas con barra de navegación tipo árbol:
Antibes, JuanLesPins
- Temas con tabla de contenidos lateral:
Berkeley, Palo Alto, Goettingen, Marburg, Hannover
- Temas con navegación en miniframe:
Berlin, Ilmenau, Dresden, Darmstadt, Frankfurt, Singapore, Szeged
- Temas con menús de sección y subsección:
Copenhagen, Luebeck, Malmoe, Warsaw

Estilos de presentación (II)

`\useinnertheme{Nombre del Tema}`

`\useinnertheme` se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- `default`

`\usecolortheme{Nombre del Tema}`

`\usecolortheme` permite elegir entre diversos diseños de color:

`\usefonttheme{Nombre del Tema}`

`\usefonttheme` selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- rectangles
- circles

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly
- etc...

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly
- etc...

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

- serif

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly
- etc...

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

- serif
- structurebold

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly
- etc...

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

- serif
- structurebold
- structureitalicserif

Estilos de presentación (II)

\useinnertheme{Nombre del Tema}

\useinnertheme se encarga de definir qué tipo de bloques, entornos itemize y entornos enumerate se utilizarán para mostrar información:

- default
- circles
- rectangles
- rounded
- inmargin

\usecolortheme{Nombre del Tema}

\usecolortheme permite elegir entre diversos diseños de color:

- default
- structure
- sidebar
- albatross
- beetle
- crane
- dove
- fly
- etc...

\usefonttheme{Nombre del Tema}

\usefonttheme selecciona la apariencia de los tipos de letra:

- serif
- structurebold
- structureitalicserif
- structuresmallcapsserif

Objetos multimedia: el paquete movie15

El paquete `movie15` proporciona la capacidad de incluir objetos multimedia (películas y objetos 3D) en documentos \LaTeX ; compilando con `PDF\LaTeX`, los mismos pueden ser posteriormente visualizados con versiones superiores a la 8.0 de Acrobat Reader.

Objetos multimedia: el paquete movie15

El paquete `movie15` proporciona la capacidad de incluir objetos multimedia (películas y objetos 3D) en documentos \LaTeX ; compilando con PDF \LaTeX , los mismos pueden ser posteriormente visualizados con versiones superiores a la 8.0 de Acrobat Reader.

Incluyendo éste paquete en una presentación beamer, pueden incluirse tales elementos en la presentación.

Objetos multimedia: el paquete movie15

El paquete `movie15` proporciona la capacidad de incluir objetos multimedia (películas y objetos 3D) en documentos \LaTeX ; compilando con PDF \LaTeX , los mismos pueden ser posteriormente visualizados con versiones superiores a la 8.0 de Acrobat Reader.

Incluyendo éste paquete en una presentación beamer, pueden incluirse tales elementos en la presentación.

En el apartado **ejemplos** de la página web de la asignatura, puede consultarse una presentación beamer con ejemplos de integración de tales elementos multimedia (además de presentar más ejemplos de las posibilidades de animación en beamer)

Apuntes de Latex

Capítulo 16: Ampliación de conceptos sobre escritura matemática

En éste capítulo se continúa con algunos conceptos importantes sobre escritura de fórmulas matemáticas que no fueron tratados en el capítulo 3. De interés para los matemáticos es la sección 5, donde se detallan las capacidades del paquete `amsthm`. Se recuerda de nuevo que, para disponer de todas las capacidades matemáticas relacionadas con fórmulas y símbolos en un documento, se deben cargar los paquetes `amsmath` y `amssymb`.

1. Tamaño de tipos de letra en fórmulas matemáticas

Ya mencionamos en el capítulo 3 que los tamaños de algunos símbolos en fórmulas y su distribución dependen del modo en el que nos encontramos, texto, ó párrafo. Es importante tener en cuenta que, **globalmente**, podemos escalar el tamaño de una expresión matemática utilizando los comando de cambio de tipo de letra (`\small`, `\large`, `\huge`, etc...) **frente a la misma**. Por ejemplo:

$ax + by + c = 0$	<code>\tiny \$ax + by + c = 0\$</code>
$ax + by + c = 0$	<code>\small \$ax + by + c = 0\$</code>
$ax + by + c = 0$	<code>\$ax + by + c = 0\$</code>
$ax + by + c = 0$	<code>\Large \$ax + by + c = 0\$</code>

sin embargo, éstos comandos no sirven para cambiar los tamaños de tipo de letra **dentro** de una fórmula:

$$ax + by + c = 0 \quad \$ \{ \tiny ax \} + by + \{ \Large c \} = 0 \$$$

obteniéndose además los siguientes “warnings”:

Command `\tiny invalid in math mode on input line...`
Command `\tiny invalid in math mode on input line...`

Para efectuar tales cambios, dentro del modo matemático existen hasta 4 modos predefinidos de tipo de letra (y símbolos), que se ordenan por tamaño decreciente como párrafo, texto, sub-índices y sub-sub-índices. Si un estilo predefinido no nos satisface, existe la posibilidad de cambiarlo, siendo irrelevante cómo hayamos abierto el modo

matemático (\$... \$ ó \$\$... \$\$). Para ello se utilizan los comandos:

\displaystyle \textstyle \scriptstyle \scriptscriptstyle

Por ejemplo, si una fracción en modo texto queda demasiado pequeña:

Se cumple $|x_n| < \frac{1}{2}$ para todo $n \geq p$

Se cumple $|x_n| < \frac{1}{2}$ para todo $n \geq p$

puede cambiarse con:

Se cumple $|x_n| < \displaystyle\frac{1}{2}$ para todo $n \geq p$

Se cumple $|x_n| < \frac{1}{2}$ para todo $n \geq p$

Así, distintos modos pueden combinarse libremente dentro de la misma fórmula, por ejemplo:

```
\[
f(a+h) = f(a) + \frac{f'(a)h}{1} + \textstyle \frac{f''(a)h^2}{2!} + \dots + \scriptstyle \frac{f^{(n)}(a)h^n}{n!} + \scriptscriptstyle o(h^n)
\]
```

$$f(a+h) = f(a) + \frac{f'(a)h}{1} + \frac{f''(a)h^2}{2!} + \dots + \frac{f^{(n)}(a)h^n}{n!} + o(h^n)$$

Es interesante destacar que, habitualmente, los tamaños de los subíndices y superíndices se ajustan utilizando los tamaños `scriptstyle` y `scriptscriptstyle`; hay que tener en cuenta que `scriptscriptstyle` es el tamaño *mínimo* disponible, por lo que, por ejemplo, en la ecuación anterior se observa cómo para el último término ($o(h^n)$) el exponente n tiene el mismo tamaño que la base h (cuando debería ser más pequeño, debido a que no existe un tamaño menor que `scriptscriptstyle`).

Otro ejemplo:

$$\$\$e^{\int x^2 dx} \\quad e^{\int x^2 dx} \$\$$$

$$e^{\int x^2 dx} \quad e^{\int x^2 dx}$$

2. Manejo avanzado de espacios en fórmulas

A la hora de determinar el espaciado entre diversos elementos de una fórmula, L^AT_EX clasifica tales elementos de la siguiente forma:

1. *Ordinarios* (Ord): Letras latinas (las cuales, en modo matemático, se escriben en itálica), griegas, números, símbolos \exists , \emptyset , ∞ , etc...
2. *Operadores de tamaño variable* (Op): Σ , Π , \int , etc...
3. *Operadores binarios* (Bin): $+$, \cup , \times , etc...
4. *Operadores de relación* (Rel): Por ejemplo, símbolos como $=$, $<$, \approx , etc...; flechas como \cup , \rightarrow , \iff , etc...; símbolos de inclusión \subset , \supset , $\not\subset$, etc...
5. *Delimitadores de apertura* (Ape): $($, $\{$, etc...
6. *Delimitadores de cierre* (Cie): $)$, $\}$, etc...
7. *Signos de puntuación* (Pun): $?$, $,$, (coma), etc...

La importancia de esta clasificación reside en que L^AT_EX inserta espacios entre símbolos de acuerdo a las clases a las que pertenecen. La tabla siguiente detalla el tamaño de los espacios; Los números 0, 1, 2 y 3 indican, respectivamente, que no se deja espacio, que se deja un espacio pequeño (`\thinspace`), un espacio medio (`\medspace`) ó un espacio grande (`\thickspace`). Los números entre paréntesis denotan casos en los que los espacios no se insertan si estamos en modo de escritura de subíndices ó superíndices. Las entradas con asterisco son casos que no pueden presentarse.

	Ord	Op	Bin	Rel	Ape	Cie	Pun
Ord	0	1	(2)	(3)	0	0	0
Op	1	1	*	(3)	0	0	0
Bin	(2)	(2)	*	*	(2)	*	*
Rel	(3)	(3)	*	0	(3)	0	0
Ape	0	0	*	0	0	0	0
Cie	0	1	(2)	(3)	0	0	0
Pun	(1)	(1)	*	(1)	(1)	(1)	(1)

La siguiente tabla recopila los distintos comandos de espaciado utilizables en modo matemático (los comenzados por “neg” equivalen a distancias negativas):

Comando	Abreviatura	Comando	Abreviatura	Espacio
<code>\thinspace</code>	<code>\,</code>	<code>\thinspace</code>	<code>\!</code>	 (1.82pt)
<code>\medspace</code>	<code>\:</code>	<code>\medspace</code>		 (2.43pt)
<code>\thickspace</code>	<code>\;</code>	<code>\negthickspace</code>		 (3.04pt)
<code>\quad</code>				 (10.95pt)
<code>\quad</code>				 (21.9pt)

Veamos un ejemplo:

Para dos números x e y , definimos una operación \circ como:

$$x \circ y = x + y - xy$$

la cual es asociativa.

produce:

Para dos números x e y , definimos una operación \circ como:

$$x \circ y = x + y - xy$$

la cual es asociativa.

En la lista de símbolos, vemos que `\circ` está clasificado como operador binario. Si intentamos lo mismo con el símbolo \square (`\Box`, clasificado como “Símbolos varios”) obtenemos lo siguiente:

Para dos números x e y , definimos una operación \square como:

$$x\square y = x + y - xy$$

la cual es asociativa.

Los espacios han desaparecido, debido a que \square no es considerado como operador binario. Sin embargo, L^AT_EX permite, en modo matemático, cambiar el comportamiento (y espacios predefinidos adyacentes) de cualquier símbolo. Para ello disponemos de los comandos:

- `\mathord` → Símbolo ordinario
- `\mathrel` → Operador de relación
- `\mathbin` → Operador binario

que respectivamente permiten asignar comportamientos de símbolos ordinarios, de relación, ó binarios. Así por ejemplo, en el caso anterior, tecleando `\mathbin` delante de `\Box` (\$\$ x\mathbin{\Box} y=x+y-xy \$\$) obtendríamos:

$$x \square y = x + y - xy$$

3. Fórmulas matemáticas resaltadas

3.1. Etiquetando y referenciando ecuaciones

Recordemos (ver capítulo 3) que el entorno `equation` permite etiquetar ecuaciones:

$$\int x^2 dx = 2x \quad (1)$$

$$\int x^3 dx = 3x^2 \quad (2)$$

```
\begin{equation}
\int x^2 dx = 2x
\label{intcuad}
\end{equation}
\begin{equation}
\int x^3 dx = 3x^2
\label{intcub}
\end{equation}
```

Donde se observa que las ecuaciones se van numerando en orden. Para ello, L^AT_EX emplea el contador **equation**. Éste contador, en el caso de documentos tipo book tiene la representación (1.1), (1.2), etc..., denotando el número de capítulo y el número de ecuación y para documentos article simplemente (1), (2), etc...

Podemos, mediante el empleo de comandos **\label{Etiqueta}** (ver ejemplo anterior) etiquetar las ecuaciones para referenciarlas más adelante en el texto a través del comando **\eqref{Etiqueta}**; por ejemplo:

Aquí referenciamos a la ecuación **\eqref{intcuad}** y aquí a la **\eqref{intcub}** produce:

Aquí referenciamos a la ecuación (1) y aquí a la (2)

Podemos resetear si nos interesa el valor de éste contador con:

\setcounter{equation}{0}

Si queremos que se reinicie al principio de cada sección (suponiendo una clase article), y que se incluya además en la representación el número de sección, se puede emplear:
\numberwithin{equation}{section}
 (comprobar con unos ejemplos sencillos en un article)

Otra herramienta interesante para numerar ecuaciones es el entorno **subequations**, con sintaxis:

```
\begin{subequations}
ParteDelDocumento
\end{subequations}
```

Todas las ecuaciones dentro de éste entorno, serán numeradas como (6a), (6b), etc... La última ecuación anterior al entorno y la inmediatamente siguiente serán numeradas como (5) y (7), respectivamente.

El formato de las etiquetas de ecuación puede cambiarse puntualmente con el comando **\tag{NuevaEtiqueta}**, que coloca el argumento NuevaEtiqueta entre paréntesis, ó con **\tag*{NuevaEtiqueta}**, análogo pero que suprime los paréntesis. También puede utilizarse **\notag**, que suprime la etiqueta, y que es análogo a **\nonumber**. La ubicación vertical también es modificable (por ejemplo, nos puede convenir colocar la etiqueta a mitad de camino entre dos ecuaciones), lo cual se consigue con el comando **\raisetag{Longitud}**

3.2. Ecuaciones multilínea

En caso de que una ecuación sea demasiado larga para caber en una línea, se puede utilizar el entorno `multiline*`, por ejemplo:

$$(a + b + c + d + e)^2 = a^2 + b^2 + c^2 + d^2 + e^2 \\ + 2ab + 2ac + 2ad + 2ae + 2bc + 2bd + 2be + 2cd + 2ce + 2de$$

se produce con:

```
\begin{multiline*}
(a+b+c+d+e)^2=a^2+b^2+c^2+d^2+e^2\\
+2ab+2ac+2ad+2ae+2bc+2bd+2be+2cd+2ce+2de
\end{multiline*}
```

Para ecuaciones en más de dos líneas, los resultados no son muy satisfactorios. Por ejemplo: tecleando:

```
\begin{multiline*}
(a+b+c+d+e+f)^2=a^2+b^2+c^2+d^2+e^2+f^2\\
+2ab+2ac+2ad+2ae+2af\\
+2bc+2bd+2be+2bf\\
+2cd+2ce+2cf\\
+2de+2df\\
+2ef
\end{multiline*}
```

obtenemos:

$$(a + b + c + d + e + f)^2 = a^2 + b^2 + c^2 + d^2 + e^2 + f^2 \\ + 2ab + 2ac + 2ad + 2ae + 2af \\ + 2bc + 2bd + 2be + 2bf \\ + 2cd + 2ce + 2cf \\ + 2de + 2df \\ + 2ef$$

Para solucionar esto se puede utilizar el entorno `split`, que *no puede usarse independientemente*, esto es, debe incluirse dentro de alguna estructura tipo `equation`. Por ejemplo, podemos modificar el caso anterior tecleando:

```
\begin{equation*}
\begin{split}
```

```

(a+b+c+d+e+f)^2 &= a^2+b^2+c^2+d^2+e^2+f^2 \\
&\quad +2ab+2ac+2ad+2ae+2af \\
&\quad +2bc+2bd+2be+2bf \\
&\quad +2cd+2ce+2cf \\
&\quad +2de+2df \\
&\quad +2ef
\end{split}
\end{equation*}

```

con lo que se tiene:

$$\begin{aligned}
(a+b+c+d+e+f)^2 &= a^2+b^2+c^2+d^2+e^2+f^2 \\
&\quad +2ab+2ac+2ad+2ae+2af \\
&\quad +2bc+2bd+2be+2bf \\
&\quad +2cd+2ce+2cf \\
&\quad +2de+2df \\
&\quad +2ef
\end{aligned}$$

Este entorno también es útil cuando la ecuación contiene múltiples igualdades; por ejemplo:

```

\begin{equation*}
\begin{split}
(a+b)^2 &= (a+b)(a+b) \\
&= a^2+ab+ba+b^2 \\
&= a^2+2ab+b^2
\end{split}
\end{equation*}

```

produce:

$$\begin{aligned}
(a+b)^2 &= (a+b)(a+b) \\
&= a^2+ab+ba+b^2 \\
&= a^2+2ab+b^2
\end{aligned}$$

3.3. Grupos de ecuaciones

Un grupo de ecuaciones puede ser escrito utilizando el entorno `gather`:

```

\begin{gather*}
(a,b)+(c,d)=(a+c,b+d) \\
(a,b)(c,d)=(ac-bd,ad+bc)
\end{gather*}

```

produce como salida:

$$(a, b) + (c, d) = (a + c, b + d)$$

$$(a, b)(c, d) = (ac - bd, ad + bc)$$

Cuando un grupo de ecuaciones deben formar una sola unidad, el modo lógicamente correcto de escribirlas es incluyendo cierta alineación. Para ello podemos utilizar el entorno `align*`, como se ve a continuación:

Suponemos que x , y , z satisfacen las ecuaciones:

```
\begin{align*}
x+y-z &= 1 \\
x-y+z &= 1
\end{align*}
```

obteniendo:

Suponemos que x, y, z satisfacen las ecuaciones:

$$\begin{aligned}
x + y - z &= 1 \\
x - y + z &= 1
\end{aligned}$$

Dentro del entorno `align*` podemos añadir un texto intermedio, sin romper la alineación, con el comando `\intertext{Texto}`. Por ejemplo:

Suponemos que x , y , z satisfacen las ecuaciones:

```
\begin{align*}
x+y-z &= 1 \\
x-y+z &= 1 \\
\intertext{y por hipótesis}
x+y+z &= 1
\end{align*}
```

Suponemos que x, y, z satisfacen las ecuaciones:

$$\begin{aligned}
x + y - z &= 1 \\
x - y + z &= 1
\end{aligned}$$

y por hipótesis

$$x + y + z = 1$$

En éste entorno, también pueden alinearse varias ecuaciones en varias columnas; todo lo que se necesita es añadir separadores “&” extra:

Comparamos los siguientes conjuntos de ecuaciones:

```
\begin{align*}
\cos^2 x + \sen^2 x &= 1 & \cosh^2 x - \senh^2 x &= 1 \\
\cos^2 x - \sen^2 x &= \cos 2x & \cosh^2 x + \senh^2 x &= \cosh 2x
\end{align*}
```

Comparamos los siguientes conjuntos de ecuaciones:

$$\begin{array}{ll} \cos^2 x + \sen^2 x = 1 & \cosh^2 x - \senh^2 x = 1 \\ \cos^2 x - \sen^2 x = \cos 2x & \cosh^2 x + \senh^2 x = \cosh 2x \end{array}$$

Imaginemos que queremos modificar lo anterior en la siguiente forma:

Comparamos los siguientes conjuntos de ecuaciones:

$$\begin{array}{ll} \cos^2 x + \sen^2 x = 1 & \cosh^2 x - \senh^2 x = 1 \\ \cos^2 x - \sen^2 x = \cos 2x & \quad \text{y} \quad \cosh^2 x + \senh^2 x = \cosh 2x \end{array}$$

Esto no se puede escribir utilizando los entornos ya vistos, dado que cualquiera de ellos ocupa toda la anchura de texto; el paquete `amsmath` proporciona las variantes `gathered`, `aligned` y `alignedat`, que ocupan sólo la anchura real de los contenidos. Así, lo anterior se puede obtener a partir del código:

Comparamos los siguientes conjuntos de ecuaciones:

```
\begin{equation*}
\begin{aligned}
&\cos^2 x + \sen^2 x &= 1 \\
&\cos^2 x - \sen^2 x &= \cos 2x
\end{aligned}
\qquad \text{y} \qquad
\begin{aligned}
&\cosh^2 x - \senh^2 x &= 1 \\
&\cosh^2 x + \senh^2 x &= \cosh 2x
\end{aligned}
\end{equation*}
```

Una estructura común en matemáticas es definición de funciones a trozos, por ejemplo:

$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x \leq 0 \end{cases}$$

lo cual se obtiene utilizando el entorno `cases` de `amsmath` dentro de una ecuación:

```
\begin{equation*}
|x| =

```

```
\begin{cases}
x & \text{if } x \geq 0 \\
-x & \text{if } x \leq 0
\end{cases}
\end{cases}
```

La siguiente tabla ilustra con ejemplos sencillos las distintas posibilidades básicas vistos hasta ahora:

Comparación de diferentes entornos para ecuaciones

```
\begin{equation*}
a=b
\end{equation*}
```

```
\begin{equation}
a=b
\end{equation}
```

```
\begin{equation}\label{xx}
\begin{aligned}
&= b + c - d \\
&\quad + e - f \\
&= g + h \\
&= i
\end{aligned}
\end{equation}
```

```
\begin{multline}
a+b+c+d+e+f \\
+i+j+k+l+m+n
\end{multline}
```

```
\begin{gathered}
a_1=b_1+c_1 \\
a_2=b_2+c_2-d_2+e_2
\end{gathered}
```

```
\begin{aligned}
a_1 &= b_1 + c_1 \\
a_2 &= b_2 + c_2 - d_2 + e_2
\end{aligned}
```

$$a = b$$

$$(3)$$

$$\begin{aligned}
a &= b + c - d \\
&\quad + e - f \\
&= g + h \\
&= i
\end{aligned}$$

$$(4)$$

$$\begin{aligned}
&a + b + c + d + e + f \\
&\quad + i + j + k + l + m + n
\end{aligned}$$

$$(5)$$

$$a_1 = b_1 + c_1$$

$$(6)$$

$$a_2 = b_2 + c_2 - d_2 + e_2$$

$$(7)$$

$$a_1 = b_1 + c_1$$

$$(8)$$

$$a_2 = b_2 + c_2 - d_2 + e_2$$

$$(9)$$

```
\begin{align}
a_{11}&=b_{11} & a_{12} = b_{12} & (10) \\
a_{12}&=b_{12} \\ 
a_{21}&=b_{21} & a_{21} = b_{21} & (11) \\
a_{22}&=b_{22}+c_{22} \\
\end{align}
```

```
\begin{flalign*}
a_{11}&=b_{11} & a_{12} = b_{12} & \\
a_{12}&=b_{12} \\ 
a_{21}&=b_{21} & a_{22} = b_{22} + c_{22} \\
a_{22}&=b_{22}+c_{22} \\
\end{flalign*}
```

Otra posibilidad más para alinear ecuaciones es el entorno `eqnarray`. Funciona de forma similar al entorno `align`, y su sintaxis es la siguiente:

```
\begin{eqnarray}
\text{FormulaIzquierda1} && \text{FormulaDerecha1} \\
\text{FormulaIzquierda1} && \text{FormulaDerecha2} \\
\ldots \\
\end{eqnarray}
```

Existe igualmente en dos versiones, sin asterisco ó con asterisco, lo cual implica que se numeran ó no todas las ecuaciones, respectivamente. Para no numerar una ecuación en particular, se debe utilizar el comando `\nonumber` delante del salto de línea `\backslash\backslash`.

4. Miscelánea

4.1. Fracciones generalizadas y continuas

El comando `\genfrac` se puede utilizar para producir fracciones personalizadas, con la sintaxis:

```
\genfrac{\Delim.Izqdo}{\Delim.Derecho}{\GrosorLínea}{\Tamaño}{\Numerador}{\Denominador}
```

Para `Tamaño`, se puede elegir entre los valores 0, 1, 2 y 3, que corresponden respectivamente a `\displaystyle`, `\textstyle`, `\scriptstyle` y `\scriptscriptstyle` (OJO! valores mayores implican entonces tamaños más pequeños). Veamos un ejemplo:

$$\left\{ \begin{matrix} ij \\ k \end{matrix} \right\} = g^{k_1} \left[\begin{matrix} ij \\ 1 \end{matrix} \right] + g^{k_2} \left[\begin{matrix} ij \\ 2 \end{matrix} \right]$$

```
\begin{equation*}
```

```
\genfrac{\{}{\}}{0pt}{}{ij}{k}=  
g^{k1}\genfrac{[}{]}{0pt}{}{ij}{1}  
+g^{k2}\genfrac{[}{]}{0pt}{}{ij}{2}  
\end{equation*}
```

Las fracciones continuas se obtienen a través del comando \cfrac:

$$\frac{4}{\pi} = 1 + \cfrac{1^2}{2 + \cfrac{3^2}{2 + \cfrac{5^2}{2 + \dots}}}$$

```
\begin{equation*}  
 \frac{4}{\pi}=1+\cfrac{1^2}{2+  
 \cfrac{3^2}{2+  
 \cfrac{5^2}{2+\dotsb}}}  
\end{equation*}
```

4.2. El comando \smash

El comando \smash[Argumento]{Objeto} anula la altura ó profundidad (si Argumento toma el valor t ó b, respectivamente) de Objeto, lo cual, por ejemplo, puede ser útil para hacer más consistente el aspecto de radicales adyacentes de distinto tamaño:

```
\[ x:=(1/\sqrt{x_{i_j}})\sqrt{1+y} \quad  
x:=(1/\sqrt{\smash[b]{x_{i_j}}})\sqrt{1+y} \]
```

$$x := (1/\sqrt{x_{i_j}}) \sqrt{1+y} \quad x := (1/\sqrt{x_{i_j}}) \sqrt{1+y}$$

5. Teoremas, lemas, corolarios; el paquete amsthm

(continuará...)

Apuntes de LATEX

Capítulo 17: Gestión de tipos

Abstract

En éste capítulo trataremos las posibilidades existentes para manejar tipos (ó caracteres) en LATEX. Existe una amplia variedad de tipos a nuestra disposición, cada uno de ellos almacenado en un paquete específico. Los tipos por defecto son los llamados “Computer Modern Fonts”, creados por D. E. Knuth empleando el programa METAFONT. Este programa permite construir los tipos utilizando un lenguaje gráfico especial. A continuación describiremos cómo utilizar otras familias de tipos a nuestra disposición.

1 Fuentes postscript

Además de las fuentes Computer Modern creadas por Knuth, podemos utilizar otros muchos diferentes tipos. Existen dos procedimientos para el cambio de fuentes; si queremos cambiar el tipo de letra para TODO EL DOCUMENTO, se debe cargar uno de los siguientes paquetes de la tabla en el preámbulo (por ejemplo, para éste documento se ha usado el paquete mathptmx, que proporciona el tipo de letra times):

Paquete	Código	roman	sansserif	typewriter	formulas
mathpazo	ppl	Palatino			≈ Palatino
mathptmx	ptm	Times			≈ Times
helvet	phv		Helvetica		
avant	pag		Avant Garde		
courier	pcr			Courier	
bookman	pbk	Bookman			
newcent	pnc	New Century			
charter	bch	Charter			
chancery	pzc	Zapf Chancery			

Si por el contrario deseamos cambios puntuales de tipo de letra, debemos emplear los códigos de tres letras (ptm, phv, etc...) en combinación con los siguientes comandos de bajo nivel que especifican las propiedades de la fuente:

- \fontencoding{Codificación} Tipo de codificación de la fuente: OT1, T1, OT2... (no es necesario especificarla en general, si ya lo hemos hecho en el preámbulo)
- \fontfamily{Familia} La familia se especifica a través de los códigos de tres letras mencionados en la tabla, que identifican a la fuente
- \fontseries{Serie} Se pueden emplear los valores m (medio ó normal), b (negrita), bx (negrita extendida), sb (seminegrita) y c (condensada). Puede que no todos ellos estén disponibles, éso depende de la familia en particular
- \fontshape{Perfil} n (normal ó recto), it (italíco), sl (inclinado), sc (versalita)

- `\fontsize{Tamaño}{Interlínea}` Tamaño es una longitud rígida, mientras que Interlínea es otra longitud que puede admitir valores elásticos; Para un resultado correcto es aconsejable que la interlínea sea aproximadamente un 20% mayor que el tamaño del tipo de letra. Debe hacerse notar que, en ciertos casos (las fuentes Computer Modern, por ejemplo), se pueden haber fijado valores admisibles del tamaño, con lo que L^AT_EX sustituirá el valor elegido para Tamaño por el más cercano entre los admisibles. En tales situaciones, se debe optar por emplear los comandos `\resizebox` ó `\scalebox`
- `\selectfont` Tras haber configurado los parámetros anteriores (ó alguno de ellos), se debe declarar este comando para hacer efectivos los cambios. Si queremos que tales cambios sean locales (por ejemplo para cambiar el tipo de un determinado párrafo) deberían encerrarse todos los comandos y el texto al que afectan entre llaves (a fin de delimitar un grupo)

por ejemplo:

Ejemplo:

```
\newdimen\tamanyo
\newdimen\interlinea
\def\letra#1#2{%
\tamanyo=#1%
\interlinea=1.2\tamanyo%
\fontfamily{pbk}
\fontsize{\the\tamanyo}{%
\the\interlinea}\selectfont#2}
\letra{1pt}{Hola} \letra{5pt}{Hola}
\letra{10pt}{Hola}
\letra{20pt}{Hola} \\
\letra{1cm}{Hola} \\
\letra{2cm}{Hola}
```

_Hola Hola **Hola**

Hola
Hola

2 Los paquetes `pifont` y `marvosym`

Los paquetes `pifont` y `marvosym` proporcionan herramientas para manejar con más comodidad las fuentes `Symbol`, `ZapfDingbats` y `MarvoSym`. Por ejemplo, con el paquete `pifont`, podemos acceder a las fuentes `Symbol` y `ZapfDingbats` mediante el comando

`\Pisymbol{Fuente}{Número}`

donde `Fuente` admite como opciones `psy` ó `pzd`, y `Número` tiene el mismo significado que para el comando `\symbol`.

`\Pifill{Fuente}{Número}`

`\Piline{Fuente}{Número}`

rellenan un espacio extensible, en el primer caso, y una línea entera, en el segundo, con el símbolo escogido.

Tenemos también entornos para construir listas análogas a `itemize` y `enumerate`, en las cuales cada ítem viene precedido de un símbolo obtenido de las fuentes `Symbol` ó `ZapfDingbats`:

`\begin{Pilist}{Fuente}{Número}` `\item xxx ...` `\end{Pilist}`

`\begin{Piautolist}{Fuente}{Número}` `\item xxx ...` `\end{Piautolist}`

donde para el primero, el símbolo se mantiene constante (como en `itemize`), y en el segundo, va cambiando a partir de un valor inicial descrito por `Número`

Como lo más frecuente es utilizar la fuente ZapfDingbats, existen versiones simplificadas de los anteriores comandos y entornos:

```
\ding{Número}          \dingfill{Número}          \dingline{Número}  
\begin{dinglist}{Número} ...  \begin{dingautolist}{Número} ...
```

Ejemplo:

El paquete babel permite gestionar, entre otros, los idiomas:

```
\begin{dingautolist}{'266}  
\item Español  
\item Catalán  
\item Gallego  
\end{dingautolist}
```

El paquete babel permite gestionar, entre otros, los idiomas:

- ① Español
 - ② Catalán
 - ③ Gallego
-

Finalmente, el paquete marvosym implementa un comando básico `\mvchr{Número}`, de significado análogo a `\char`. Además, proporciona una serie de comandos, en inglés, que nombran los diversos símbolos de la tabla de la fuente (más información en la documentación del paquete). Por ejemplo:

`\Letter` ☎ `\Mobilefone` ☎ `\Faxmachine` ☎ etc...

Apuntes de LATEX

Capítulo 18: Miscelánea de paquetes

1. Letras capitales

1.1. El paquete `lettrine`

El paquete `lettrine` permite de forma muy sencilla el incluir letras capitales al comienzo de un párrafo; su sintaxis básica es:

```
\lettrine[opciones]{Letra}{Texto Resaltado}
```

y debe emplearse al comienzo de un párrafo. El argumento `Letra` indica la letra capital, mientras que el texto incluido en el argumento `Texto Resaltado` es impreso en letras mayúsculas (tipo `small caps`).

Es importante descartar que, para que el paquete funcione correctamente, se debe utilizar la opción `\usepackage[T1]{fontenc}`, así como emplear uno de los siguientes tipos de letra:

```
\usepackage{palatino} \usepackage{type1ec}  
\usepackage{lmodern} \usepackage{type1cm}
```

Los siguientes ejemplos ilustran el efecto obtenido, así como algunas de las diferentes opciones que se pueden emplear:

```
\lettrine{E}{n} un lugar de la mancha...
```

EN un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

```
\lettrine[lines=1]{E}{n} un lugar de la mancha...
```

EN un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

```
\lettrine[lines=3]{\textit{E}}{n un lugar de la Mancha}...
```

EN UN LUGAR DE LA MANCHA, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

```
\lettrine[lhang=1, nindent=0pt, lines=3]{E}{n}...
```

EN un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

```
\lettrine[lines=3, lhang=0.33, loversize=0.25]{E}{n}
```

EN un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

```
\lettrine[lines=4, slope=0.6em, findent=-1em, nindent=0.6em]{A}{un lugar de la mancha}...
```

AUN LUGAR DE LA MANCHA de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

```
\lettrine[lines=4, slope=-0.5em, lhang=0.5, nindent=0pt]{V}{ivía}...
```

VIVÍA un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

El siguiente ejemplo ilustra el modo (con la opción “image=true”) de incluir una imagen en un archivo gráfico externo:

```
\lettrine[image=true, lines=3]{letraW.eps}{\enceslao}...
```

 ENCESLAO fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos. fué el nombre de uno de los reyes godos.

1.2. El paquete **yfonts**

Cargando el paquete **yfonts** en el preámbulo, podemos añadir letras capitales de tipo gótico, mediante el comando `\yinipar{Letra}` al comienzo de un párrafo. Por ejemplo:

\yinipar{U} na olla de algo más...

 na olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

\yinipar{E} n un lugar de la mancha...

 n un lugar de la mancha de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda.

2. Marcas de fondo y wallpapers

El paquete `draftcopy` permite colocar una palabra (draft por defecto) como fondo de una ó más páginas. El método de utilización es cargar el paquete (consultar documentación para algunas de sus opciones) y, en el preámbulo, especificar algunos de los siguientes comandos:

- `\draftcopySetGrey{Tono}` Nivel de gris de la marca de fondo (entre 0.0 y 1.0)
- `\draftcopyFirstPage{Número}` Primera página a marcar
- `\draftcopyLastPage{Número}` Última página a marcar
- `\draftcopyName{Texto}{Número}` Texto de la marca de fondo (DRAFT por defecto) y escala de la fuente (215 por defecto)
- (consultar documentación del paquete para más comandos)

Es importante tener en cuenta que éste paquete sólo funciona correctamente compilando con `LATEX + dvips + ps2pdf`. Si necesitamos compilar con `pdflatex`, existe una modificación (no incluida en TexLive, y que necesita instalación manual) llamada `pdf-draftcopy`.

Como ejemplo, para obtener el presente documento, que marca con la palabra “borrador” la página 2, se incluyeron las intrucciones:

```
\usepackage[light]{draftcopy}
\draftcopyName{Borrador}{180}
\draftcopyFirstPage{2}
\draftcopyLastPage{2}
```

en el preámbulo.

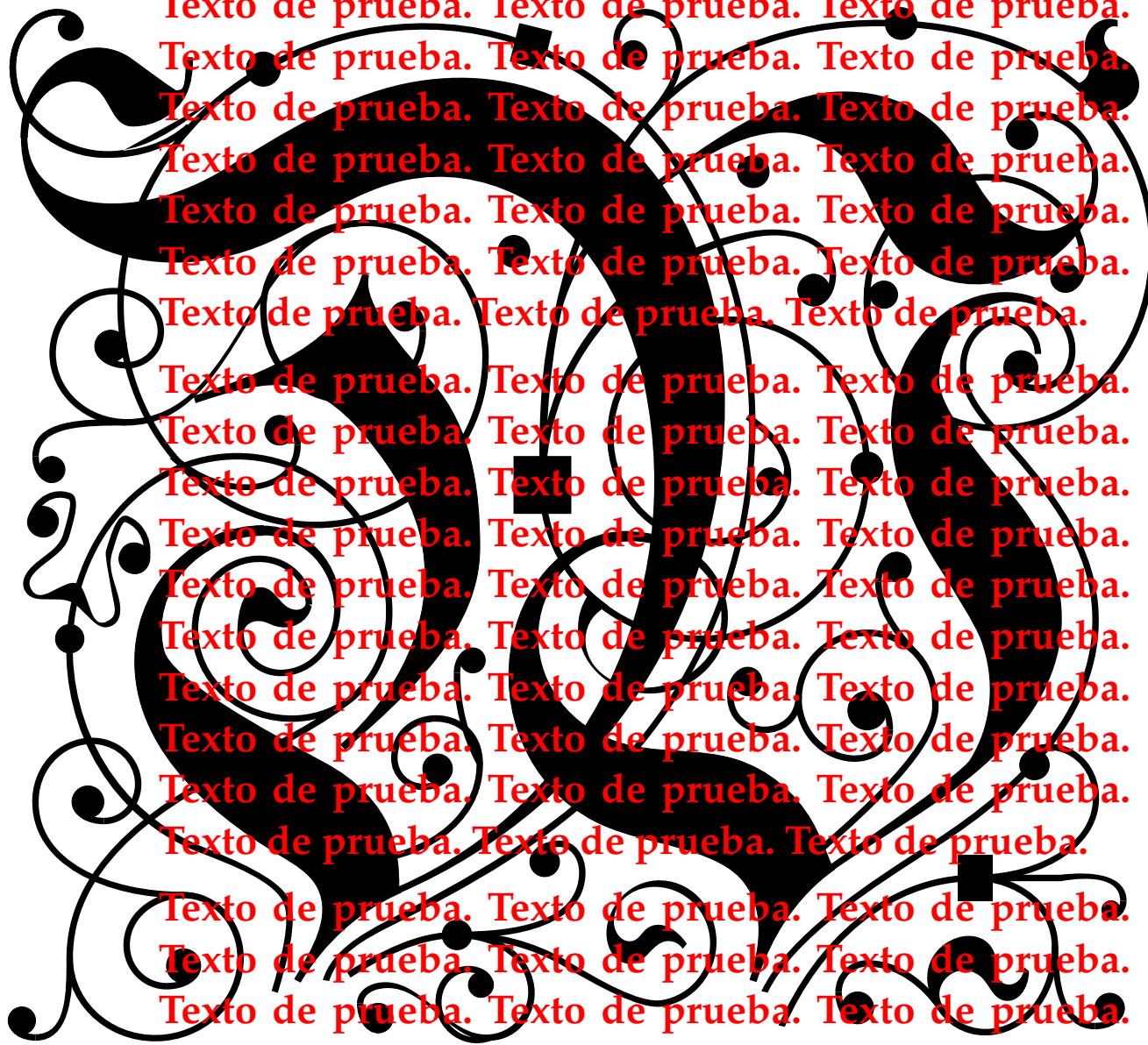
Otra posibilidad es colocar una imagen de fondo utilizando el paquete `wallpaper`. Los siguientes comandos permiten colocar global ó selectivamente una ó varias copias de una imagen como papel de fondo:

- `\CenterWallPaper{Escala}{Fichero.eps}` Coloca la imagen en `Fichero.eps` como fondo, con el factor de escala (relativo a las dimensiones de la página) especificado
- `\ThisCenterWallPaper{Escala}{Fichero.eps}` Lo mismo, pero sólo para la página actual.
- `\TileWallPaper{Anchura}{Altura}{Fichero.eps}` Empapela todas las páginas con copias de Anchura y Altura dadas
- `\ThisTileWallPaper{Anchura}{Altura}{Fichero.eps}` Lo mismo para sólo la página actual
- `\TileSquareWallPaper{NumeroDeCopias}{Fichero.eps}` Análogo a `\TileWallPaper`, pero con copias cuadradas, y especificando el número de copias por página

Las siguientes dos páginas ilustran el empleo de éstas imágenes de fondo, siendo el resultado de utilizar el código:

```
\newpage
\color{red} \bfseries \LARGE
\ThisCenterWallPaper{0.85}{letraw.eps}
Texto de Prueba...

\newpage
\ThisTileSquarePaper{7}{letraw.eps}
Texto de Prueba...
```



Apuntes de Latex

Capítulo 19: Bibliografía avanzada

SECCIÓN 1

Generalidades: Esquema estándar

Recordemos (ver tema 6) que para incluir citas bibliográficas en un documento, se emplea el comando `\cite{etiqueta}`. Posteriormente, mediante el entorno `thebibliography`, procedemos a citar las distintas referencias colocadas a lo largo del documento:

```
\begin{thebibliography}{Numero}
\bibitem[NuevaMarca]{etiqueta1} Información bibliográfica
\bibitem[NuevaMarca]{etiqueta2} Información bibliográfica
...
\bibitem[NuevaMarca]{etiquetaN} Información bibliográfica
\end{thebibliography}
```

donde el argumento `Numero` indica el numero aproximado de citas bibliográficas en el documento (se utiliza para calcular la indentación de los ítems de la lista de referencias). Tal procedimiento lleva a una lista de citas ordenada numéricamente por el orden que ocupan dentro de la lista *y no por el orden en el que fueron citadas a lo largo del documento*. Ésto implica entonces la necesidad de llevar a cabo tal ordenación manualmente. El argumento operativo `NuevaMarca` puede utilizarse para cambiar la marca estándar ([1], [2], etc..) por un texto cualquiera.

Para cambiar el título estándar de la sección de referencias bibliográficas (`References` en L^AT_EX inglés, y `Referencias` si se emplea la opción `spanish` de `babel`) se debe renombrar la variable `\refname` del modo siguiente:

```
\renewcommand{\refname}{Bibliografía}
```

(si deseamos `Bibliografía` en el título, por ejemplo). En caso de utilizar `babel`, es muy importante tener en cuenta que tal comando se debe ejecutar *después* de `\begin{document}`. Se debe tener también en cuenta que, en el caso de utilizar la clase `book`, el comando `\refname` pasa a convertirse en `\bibname`.

A diferencia de cualquier otro tipo de sección, la de bibliografía no es incluida en la tabla de contenidos que se obtiene a través del comando `tableofcontents`. Si necesitamos incluirla, se puede colocar el siguiente código en el preámbulo del documento:

```
\let\OLDthebibliography=\thebibliography
\def\thebibliography#1{\OLDthebibliography{#1}%
\addcontentsline{toc}{section}{\refname}}
```

El comando `\let` (ver capítulo 8) se encarga de guardar una “copia de seguridad” del comando `\thebibliography`, bajo la denominación `\OLDthebibliography`; seguidamente, redefinimos el comando `\thebibliography` (que, al igual que su copia “`\OLDthebibliography`”, depende de un argumento), indicando que debe ejecutarse primeramente `\OLDthebibliography` (es decir, todo lo que hace el comando `\thebibliography` original) y seguidamente la instrucción `\addcontentsline{toc}{section}{\refname}`, que se encarga de añadir a la tabla de contenidos una nueva *sección* (de ahí el argumento `section`) de nombre `\refname`. Para el caso de un documento de clase `book`, deberíamos reemplazar `section` por `chapter`, y `\refname` por `\bibname`, ésto es: `\addcontentsline{toc}{chapter}{\bibname}`.

El comando `\cite` admite un argumento optativo, que podemos utilizar para añadir comentarios extra durante la cita de una referencia bibliográfica. Así por ejemplo, si empleamos:

Texto diverso `\cite[pág.\ 24--44]{knuth}`

(donde `knuth` es la etiqueta correspondiente a la referencia número 1), obtenemos:

Texto diverso [1, pág. 24–44]

lo cual constituye un método útil, a la hora de citar un libro, de especificar una parte del mismo; otra aplicación interesante de éste procedimiento sería utilizar el argumento opcional para citar capítulos.

En la lista de referencias bibliográficas al final del documento, puede interesarnos cambiar el formato estándar de las etiquetas (con números entre corchetes) por otro formato. Para ello, debemos redefinir el comando `\@biblabel`. Debido a la presencia del símbolo `@`, la redefinición del comando en el preámbulo del documento debe intercalarse entre los comandos `\makeatletter` y `\makeatother`. Por ejemplo, tras:

```
\makeatletter
\renewcommand\@biblabel[1]{#1. \ }
\makeatother
```

la lista de referencias quedaría como:

1. Referencia número 1
 2. Referencia número 2
 3. Referencia número 3
- etc...

Debe hacerse notar que el comando `\@biblabel` depende de un argumento, que consiste en el número de orden de la etiqueta.

SECCIÓN 2

El paquete cite

El paquete `cite` mejora las capacidades de L^AT_EX estándar a la hora de colocar las citas a lo largo del documento. En caso de que en un lugar se haga referencia a más de 3 citas seguidas,

obtendríamos algo como: [1,2,3,4,...]. El uso del paquete `cite` permite que las listas de citas se agrupen automáticamente, del modo: [1-4,...]. Además, el paquete amplía enormemente las posibilidades de manejo del formato de las citas. Mediante los comandos:

```
\citeleft      \citeright
```

podemos cambiar el material (por defecto corchetes) a la izquierda y derecha de las listas de citas, respectivamente. Por ejemplo, para poner las citas entre paréntesis utilizaríamos:

```
\renewcommand{\citeleft}{(}
\renewcommand{\citeright}{)}
```

Debe tenerse en cuenta que en `\citeleft` pueden incluirse comandos dependientes de un argumento, que es considerado por defecto la lista de citas (es decir, 1-4, ó 2,3, etc...). Podemos utilizar entonces el comando `\fbox` del siguiente modo para obtener un curioso efecto de citas enmarcadas:

```
\renewcommand{\citeleft}{\fboxsep=2pt\fbox}
\renewcommand{\citeright}{}
```

que daría como resultado: 1–4,6

Otro comando útil introducido por el paquete `cite` es `\citeform` (que depende también de un argumento, las citas en cuestión), que se utiliza para cambiar el formato del número de cita. Por ejemplo, con:

```
\renewcommand{\citeform}[1]{\textcolor{blue}{\#1}}
```

obtenemos el texto de las citas en azul: [1–4,6]

Por último, se debe mencionar que cargando el paquete `cite` con el argumento optativo `superscript` (ésto es, `\usepackage[superscript]{cite}`), las citas se colocan en forma de superíndices a lo largo del documento, en la forma:

Texto diverso^{1–4,6}

SECCIÓN 3

Bases de datos bibliográficas y BibTeX

Una forma más organizada de trabajar, que nos ahorra el trabajo de mantener manualmente una ordenación de las citas, es construir una base de datos bibliográfica y emplear posteriormente el programa BibTeX para construir nuestra sección de referencias. Tal procedimiento posee, además, la ventaja de permitir elegir entre otros diversos esquemas de citas, como título corto, autor-fecha, etc...

La base de datos bibliográfica consiste en uno (ó varios) ficheros de extensión `.bib`, que contienen una serie de registros (ó fichas) con toda la información bibliográfica de la que disponemos. La sintaxis básica de un registro es:

```
@TipoDeRegistro{Etiqueta,CampoA={Contenido},CampoB={Contenido},etc...}
```

donde `TipoDeRegistro` denota qué clase de documento referenciamos, y `Etiqueta` va a ser la etiqueta que debemos emplear como argumento del comando `\cite` para citar la referencia en cualquier documento. Posteriormente, se deben especificar una serie de campos, algunos obligatorios, y otros optativos, con la información de título, autores, año de publicación, etc...

El carácter obligatorio ú optativo de diversos campos varía, en general, con el tipo de referencia bibliográfica. En la siguiente figura se puede ver un ejemplo del contenido de un fichero .bib de bibliografía típico:

```
@Article{abril2005a,
  author = {Juan Francisco Abril and Ruben Castelo and Ricardo Guigo},
  title = {Comparison of splice sites in mammals and chicken},
  journal = {Genome Research},
  year = 2005,
  volume = 15,
  pages = {111--119},
}

@Book{alberts1994a,
  author = {Bruce Alberts and David Bray and John Lewis and Michael Raff},
  title = {Molecular biology of the cell},
  publisher = {Garland publishing},
  year = 1994,
  edition = {Third},
  isbn = {0-8153-1620-8}
}

@Article{bucher1990a,
  author = {Peter Michael Bucher},
  title = {Weight matrix descriptions of four eukaryotic {RNA polymerase II} promoter elements derived from 502 unrelated promoter sequences},
  journal = {Journal of Molecular Biology},
  year = 1990,
  volume = 212,
  pages = {563-578},
}

@Article{beltran2003a,
  author = {Santiago Beltran and Enrique Blanco and Fernando Manuel Serras},
  title = {Transcriptional network controlled by the trithorax-group gene ash2 in Drosophila melanogaster},
  journal = {Proceedings of the National Academy of Sciences},
  year = {2003},
  volume = {100},
  pages = {3293--3298},
}
```

Figura 1: Contenido de una base de datos bibliográfica

A continuación se detallan los distintos tipos de registros disponibles para una base de datos, especificando los diversos campos disponibles, así como su posible carácter obligatorio ó opcional:

article Artículos de revistas periódicas

Requerido: author, title, journal, year.

Opcional: volume, number, pages, month, note.

book Libros de editorial conocida

Requerido: author or editor, title, publisher, year.

Opcional: volume or number, series, address, edition, month, note.

booklet Libros no publicados por editoriales

Requerido: title.

Opcional: author, howpublished, address, month, year, note.

inbook Parte de un libro (capítulos, secciones ó rango de páginas)

Requerido: author or editor, title, chapter and/or pages, publisher, year.

Opcional: volume or number, series, type, address, edition, month, note.

incollection Libro parte de una colección (con título propio)

Requerido: author, title, booktitle, publisher, year.

Opcional: editor, volume or number, series, type, chapter, pages, address, edition, month, note.

inproceedings Artículos de comunicaciones a congresos

Requerido: author, title, booktitle, year.

Opcional: editor, volume or number, series, pages, address, month, organization, publisher, note.

manual Documentación técnica

Opcional: author, organization, address, edition, month, year, note.

mastersthesis Tesis de máster

Requerido: author, title, school, year.

Opcional: type, address, month, note.

misc Material variado

Requerido: —

Opcional: author, title, howpublished, month, year, note.

phdthesis Tesis doctorales

Requerido: author, title, school, year.

Opcional: type, address, month, note.

proceedings Comunicaciones a congresos

Requerido: title, year.

Opcional: editor, volume or number, series, address, publisher, note, month, organization

techreport Informe publicado por alguna institución

Requerido: author, title, institution, year.

Opcional: type, number, address, month, note.

unpublished Documentos no publicados formalmente

Requerido: author, title, note.

Opcional: month, year.

Para construir una base de datos de referencias bibliográficas, es de suma utilidad emplear alguno de los muchos programas disponibles tanto en MS-Windows (**JabRef**, por ejemplo) como en Linux (**kbibtex**). Éstos programas proporcionan herramientas para ir completando de forma manual los diversos campos de cada referencia bibliográfica, haciendo innecesario el conocer la sintaxis concreta que L^AT_EX requiere para cada uno. Una vez introducidos los datos, además de poder buscar registros en la base de dato, ordenar por autor, título, etc..., podemos automáticamente exportar toda la información a un fichero de tipo **.bib**.

Una vez construido el fichero **.bib** (que debe estar colocado, obviamente, en el directorio donde compilaremos nuestro documento), se colocan a lo largo del documento los diversos comandos **\cite**; la sección de biliografía puede entonces construirse automáticamente colocando en el lugar adecuado (generalmente, al final) los comandos:

```
\bibliography{NombreFichero}      \bibliographystyle{style}
```

El primero de ellos tiene como argumento el nombre del fichero (sin incluir extensión) auxiliar de bibliografía, y se encarga de construir la lista de referencias bibliográficas. El segundo, especifica el estilo bibliográfico a emplear para construir las referencias. Existen cuatro estilos estándar disponibles en L^AT_EX: (los ejemplos correspondientes ilustran cómo, a partir de la base de datos de la Figura 1, se construye la sección de referencias:

unsrt Las entradas se numeran, y aparecen en la lista según el orden en el que son citadas a lo largo del documento (al igual que en el método estándar listas de bibliografía con **\bibitem**).

Referencias

- [1] Juan Francisco Abril, Ruben Castelo, and Ricardo Guigo. Comparison of splice sites in mammals and chicken. *Genome Research*, 15:111–119, 2005.
- [2] Bruce Alberts, David Bray, John Lewis, and Michael Raff.
- [3] Peter Michael Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212:563–578, 1990.
- [4] Santiago Beltrán, Enrique Blanco, and Fernando Manuel Serras. Transcriptional network controlled by the trithorax-group gene ash2 in drosophila melanogaster. *Proceedings of the National Academy of Sciences*, 100:3293–3298, 2003.

plain Análogo a **unsrt**, pero las entradas son ordenadas alfabéticamente. El orden es autor, después año, y por último título.

Referencias

- [1] Juan Francisco Abril, Ruben Castelo, and Ricardo Guigo. Comparison of splice sites in mammals and chicken. *Genome Research*, 15:111–119, 2005.
- [2] Bruce Alberts, David Bray, John Lewis, and Michael Raff.
- [3] Santiago Beltrán, Enrique Blanco, and Fernando Manuel Serras. Transcriptional network controlled by the trithorax-group gene ash2 in drosophila melanogaster. *Proceedings of the National Academy of Sciences*, 100:3293–3298, 2003.
- [4] Peter Michael Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212:563–578, 1990.

`abbrv` Como `plain`, pero con los nombres de autores, y revistas abreviados.

Referencias

- [1] J. F. Abril, R. Castelo, and R. Guigo. Comparison of splice sites in mammals and chicken. *Genome Research*, 15:111–119, 2005.
- [2] B. Alberts, D. Bray, J. Lewis, and M. Raff.
- [3] S. Beltrán, E. Blanco, and F. M. Serras. Transcriptional network controlled by the trithorax-group gene ash2 in drosophila melanogaster. *Proceedings of the National Academy of Sciences*, 100:3293–3298, 2003.
- [4] P. M. Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212:563–578, 1990.

`alpha` Las referencias se identifican con el nombre de autor y año de publicación. Se ordenan por etiqueta, autor, año y título.

Referencias

- [ABLR] Bruce Alberts, David Bray, John Lewis, and Michael Raff.
- [ACG05] Juan Francisco Abril, Ruben Castelo, and Ricardo Guigo. Comparison of splice sites in mammals and chicken. *Genome Research*, 15:111–119, 2005.
- [BBS03] Santiago Beltrán, Enrique Blanco, and Fernando Manuel Serras. Transcriptional network controlled by the trithorax-group gene ash2 in drosophila melanogaster. *Proceedings of the National Academy of Sciences*, 100:3293–3298, 2003.
- [Buc90] Peter Michael Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212:563–578, 1990.

La compilación del documento se efectúa entonces primero ejecutando `latex fichero.tex` (o PDFLatex), para posteriormente ejecutar `bibtex fichero` (lo cual genera la lista de bibliografía en un fichero auxiliar) y terminando con una segunda compilación en `latex` del documento.

fuente (que incluye en el mismo la lista de referencias). En el caso de trabajar en el entorno WinEdt, todas éstas acciones se hacen automáticamente al ejecutar “texify” ó “pdftexify”.

Al utilizar BibTeX, el comando `\nocite{Etiqueta}` puede emplearse para incluir en la lista de referencias citas bibliográficas que no hayan sido referenciadas en el texto principal del documento. En particular, para gestionar formatos y bases de datos, es muy útil utilizar `\nocite{*}`, que tiene como efecto imprimir la lista completa de referencias de la base de datos.

Una de las mayores ventajas del empleo de BibTeX reside en que, aparte de los cuatro estilos estándar, existen muchos otros disponibles en la distribución TeXLive, adaptados específicamente a diversos tipos de publicaciones. Por ejemplo, utilizando el estilo `achemso` obtenemos la lista de referencias en el formato correspondiente a las publicaciones de la American Chemical Society:

Referencias

- [1] Abril, J. F.; Castelo, R.; Guigo, R. *Genome Research* **2005**, *15*, 111–119.
- [2] Bucher, P. M. *Journal of Molecular Biology* **1990**, *212*, 563–578.
- [3] Beltrán, S.; Blanco, E.; Serras, F. M. *Proceedings of the National Academy of Sciences* **2003**, *100*, 3293–3298.

SECCIÓN 4

Varias listas de bibliografía con bibunits

Para incorporar varias listas de bibliografía en un documento largo (típicamente una tesis) podemos emplear el paquete `bibunits`. Una vez cargado el paquete, *en cada unidad de estructura* donde queramos añadir una bibliografía, debemos utilizar el entorno `bibunit`:

```
\begin{bibunit}[Estilo]
Texto diverso...
\cite{Etiqueta1,Etiqueta2}
\putbib[NombreBase]
\end{bibunit}
```

para delimitar la parte del documento que debe incluir la sub-bibliografía. El argumento `Estilo` especifica que estilo de bibliografía para BibTeX queremos utilizar, mientras que el comando `\putbib[NombreBase]` coloca la lista de referencias basándose en la base de datos `NombreBase`. El siguiente ejemplo ilustra el uso de éste paquete:

```
\documentclass{article}
\usepackage{bibunits}
\begin{document}
\section{Primera sección}
\begin{bibunit}[unsrt]
Una referencia \cite{abril2005a} \par
Otra referencia \cite{alberts1994a}.
\putbib[mybib]
\end{bibunit}
\section{Segunda sección}
\begin{bibunit}[abbrv]
Otra referencia más \cite{bucher1990a} \par
La última referencia \cite{beltran2003a}.
\putbib[mybib]
\end{bibunit}
\end{document}
```

1. Primera sección

Una referencia [1]
Otra referencia [2].

Referencias

- [1] Juan Francisco Abril, Ruben Castelo, and Ricardo Guigo. Comparison of splice sites in mammals and chicken. *Genome Research*, 15:111–119, 2005.
- [2] Bruce Alberts, David Bray, John Lewis, and Michael Raff.

2. Segunda sección

Otra referencia más [2]
La última referencia [1].

Referencias

- [1] S. Beltrán, E. Blanco, and F. M. Serras. Transcriptional network controlled by the trithorax-group gene ash2 in *drosophila melanogaster*. *Proceedings of the National Academy of Sciences*, 100:3293–3298, 2003.
- [2] P. M. Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *Journal of Molecular Biology*, 212:563–578, 1990.

Es de importancia vital aclarar que la primera compilación del documento fuente producirá una serie de archivos auxiliares bu1.aux, bu2.aux, etc... Tras esa primera compilación, debemos ejecutar **manualmente** bibtex (OJO! ésto no lo hace WinEdt automáticamente) sobre cada uno de éstos ficheros auxiliares. Posteriormente, la compilación final del documento fuente producirá el resultado deseado.