My TEX and LATEX-notes

cnmr.me

August 18, 2023

Contents

Introduction

This notes is for people who work with algebraic geometry and use both Plain TEX and LATEX.

Why I Write This Manul? It happens that I know there is a symble that I need but I can't remember the command. And that I want to check if some symble exists in some font that I can use directly. It also happens often that I learned some uesful command but can not remember it when I want to assume. That's why I write this cheet for myself to review.

I don't like the package `amsmath`, so I assume not this package and I will introduce some commands I defined to avoiding this package.

What's the difference between TEX and LATEX? Like [?], I will give each primitive command that I mentioned a prefix start in the index. There are many primitive commands that deal with boxes, and TEX is much pleasure to configure in the level of boxes. Using just `\hbox`, `\vbox`, `\vtop`, `\vcenter`, etc., many work can be done.

I will always give the different ways to do one thing by using TEX or LATEX.

In my opinion, one use LATEX for global document, such as Contents, Sections, Index and Bibliography. In the other hand, one use TEX for local structure, especially where the configuration is mainly vertical and horizontal. I don't like `amsmath` package since it report warning messages if one used the primitive generalized fraction commands, like `\over`, `\overwithdelims`, etc., because I like the TEX way.

I copied someone's notes, but I lost the reference.

Chapter 1

Math Mode

1.1 Math Mode: Fonts

1.1.1 Font Family

Blackboard Bold letters (`msbm`; no lowercase)

Usage: `\mathbb{R}`. Requires `amsfonts`.

$$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

One lowercase letter is available with a distinct name: $\Bbbk$ `\Bbbk`

Calligraphic letters (`cmsy`; no lowercase)

Usage: `\mathcal{M}`.

$$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

Non-CM calligraphic and script letters

(`rsfs`; no lowercase) Usage: `\usepackage{mathrsfs} \mathscr{B}`.

$$\mathscr{A B C D E F G H I J K L M N O P Q R S T U V W X Y Z}$$

(`eusm`; no lowercase) Usage: `\usepackage{euscript} \EuScript{E}`, or `\usepackage[mathcal]{euscript} \mathcal{E}`. or `\mathscr{E}` with option `mathscr`.

$$\mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

Fraktur letters (`eufm`)

Usage: `\mathfrak{S}`. Requires `amsfonts`.

$$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

$$\mathfrak{abcdefghijklmnopqrstuvwxyz}$$

1.2 Math Symbols

1.2.1 Latin letters, Arabic numerals and Greek letters

The Latin letters (Arabic numerals) are simple symbols, class 0. The default font for them in math formulas is italic (upright/roman).

When adding an accent to an i or j in math, dotless variants can be obtained with `\imath` and `\jmath`:

$\imath$ \imath	$\jmath$ \jmath	$\hat{\jmath}$ \hat{\jmath}

The Greek letters are also class 0. The default font for lowercase Greek letters in math formulas is italic while the capital Greek letters is upright/roman.

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

1.2.2 Other "basic" alphabetic symbols

These are also class 0.

$\aleph$ \aleph[a]	ℓ \ell	∂ \partial[a]	$\Game$ \Game
$\beth$ \beth	$\eth$ \eth	$\wp$ \wp	$\Im$ \Im
$\daleth$ \daleth	$\hbar$ \hbar	$\circledS$ \circledS	$\Re$ \Re
$\gimel$ \gimel	$\hslash$ \hslash	$\Bbbk$ \Bbbk	
$\complement$ \complement	$\mho$ \mho	$\Finv$ \Finv	

NOTE.Labels [a],[b] indicate `amssymb` package, font `msam` or `msbm`.

1.2.3 Miscellaneous simple symbols

These symbols are also of class 0 (ordinary) which means they do not have any built-in spacing.

$\#$ \#	$\diagup$ \diagup	$\neg$ \neg
$\&$ \&	$\diamondsuit$ \diamondsuit	$\nexists$ \nexists[a]
$\angle$ \angle[b]	$\emptyset$ \emptyset	$\prime$ \prime
$\backprime$ \backprime	$\exists$ \exists	$\sharp$ \sharp[b]
$\bigstar$ \bigstar[a]	$\flat$ \flat[b]	$\spadesuit$ \spadesuit
$\blacklozenge$ \blacklozenge	$\forall$ \forall	$\sphericalangle$ \sphericalangle[b]
$\blacksquare$ \blacksquare	$\heartsuit$ \heartsuit	$\square$ \square
$\blacktriangle$ \blacktriangle[a]	∞ \infty	$\surd$ \surd
$\blacktriangledown$ \blacktriangledown[a]	$\lozenge$ \lozenge	$\top$ \top
$\bot$ \bot	$\measuredangle$ \measuredangle[b]	$\triangle$ \triangle
$\clubsuit$ \clubsuit	∇ \nabla	$\triangledown$ \triangledown[a]
$\diagdown$ \diagdown	$\natural$ \natural[b]	$\varnothing$ \varnothing

Note 1. Labels [a,b] indicate `amssymb` package, font `msam` or `msbm`.

Note 2. Synonyms: $\neg$ `lnot`

1.2.4 Binary operator symbols

$*$	`*`	$\circleddash$	`\circleddash`^{a}	$\oplus$	`\oplus`
$+$	`+`	$\cup$	`\cup`	$\oslash$	`\oslash`
$-$	`-`	$\Cup$	`\Cup`^{a}	$\otimes$	`\otimes`
$\amalg$	`\amalg`	$\curlyvee$	`\curlyvee`^{a}	$\pm$	`\pm`
$\ast$	`\ast`	$\curlywedge$	`\curlywedge`^{a}	$\rightthreetimes$	`\rightthreetimes`^{a}
$\barwedge$	`\barwedge`^{a}	$\dagger$	`\dagger`	$\rtimes$	`\rtimes`^{b}
$\bigcirc$	`\bigcirc`	$\ddagger$	`\ddagger`	$\setminus$	`\setminus`
$\bigtriangledown$	`\bigtriangledown`	$\diamond$	`\diamond`	$\smallsetminus$	`\smallsetminus`^{b}
$\bigtriangleup$	`\bigtriangleup`	$\div$	`\div`	$\sqcap$	`\sqcap`
$\boxdot$	`\boxdot`^{a}	$\divideontimes$	`\divideontimes`^{b}	$\sqcup$	`\sqcup`
$\boxminus$	`\boxminus`^{a}	$\dotplus$	`\dotplus`^{a}	$\star$	`\star`
$\boxplus$	`\boxplus`^{a}	$\doublebarwedge$	`\doublebarwedge`^{a}	$\times$	`\times`
$\boxtimes$	`\boxtimes`^{a}	$\gtrdot$	`\gtrdot`^{b}	$\triangleleft$	`\triangleleft`
$\bullet$	`\bullet`	$\intercal$	`\intercal`^{a}	$\triangleright$	`\triangleright`
$\cap$	`\cap`	$\leftthreetimes$	`\leftthreetimes`^{a}	$\uplus$	`\uplus`
$\Cap$	`\Cap`^{a}	$\lessdot$	`\lessdot`^{b}	$\vee$	`\vee`
$\cdot$	`\cdot`	$\ltimes$	`\ltimes`^{b}	$\veebar$	`\veebar`^{a}
$\centerdot$	`\centerdot`^{a}	$\mp$	`\mp`	$\wedge$	`\wedge`
$\circ$	`\circ`	$\odot$	`\odot`	$\wr$	`\wr`
$\circledast$	`\circledast`^{a}	$\ominus$	`\ominus`		
$\circledcirc$	`\circledcirc`^{a}				

Note 1. Labels ^{a,b} indicate `amssymb` package, font `msam` or `msbm`.

Synonyms: $\wedge$ `land`, $\vee$ `lor`, $\Cup$ `doublecup`, $\Cap$ `doublecap`

1.2.5 Relation symbols: $< = > \succ \sim$ and variants

$<$	`<`	$\geqslant$	`\geqslant`^{a}	$\lnapprox$	`\lnapprox`^{b}	$\preccurlyeq$	`\preccurlyeq`^{a}
$=$	`=`	$\gg$	`\gg`	$\lneq$	`\lneq`^{b}	$\preceq$	`\preceq`
$>$	`>`	$\ggg$	`\ggg`^{a}	$\lneqq$	`\lneqq`^{b}	$\precnapprox$	`\precnapprox`^{b}
$\approx$	`\approx`	$\gnapprox$	`\gnapprox`^{b}	$\lnsim$	`\lnsim`^{b}	$\precneqq$	`\precneqq`^{b}
$\approxeq$	`\approxeq`^{b}	$\gneq$	`\gneq`^{b}	$\lvertneqq$	`\lvertneqq`^{b}	$\precnsim$	`\precnsim`^{b}
$\asymp$	`\asymp`	$\gneqq$	`\gneqq`^{b}	$\ncong$	`\ncong`^{b}	$\precsim$	`\precsim`^{a}
$\backsim$	`\backsim`^{a}	$\gnsim$	`\gnsim`^{b}	$\neq$	`\neq`	$\risingdotseq$	`\risingdotseq`^{a}
$\backsimeq$	`\backsimeq`^{a}	$\gtrapprox$	`\gtrapprox`^{a}	$\ngeq$	`\ngeq`^{b}	$\sim$	`\sim`
$\bumpeq$	`\bumpeq`^{a}	$\gtreqless$	`\gtreqless`^{a}	$\ngeqq$	`\ngeqq`^{b}	$\simeq$	`\simeq`
$\Bumpeq$	`\Bumpeq`^{a}	$\gtreqqless$	`\gtreqqless`^{a}	$\ngeqslant$	`\ngeqslant`^{b}	$\succ$	`\succ`
$\circeq$	`\circeq`^{a}	$\gtrless$	`\gtrless`^{a}	$\ngtr$	`\ngtr`^{b}	$\succapprox$	`\succapprox`^{b}
$\cong$	`\cong`	$\gtrsim$	`\gtrsim`^{a}	$\nleq$	`\nleq`^{b}	$\succcurlyeq$	`\succcurlyeq`^{a}
$\curlyeqprec$	`\curlyeqprec`^{a}	$\gvertneqq$	`\gvertneqq`^{b}	$\nleqq$	`\nleqq`^{b}	$\succeq$	`\succeq`
$\curlyeqsucc$	`\curlyeqsucc`^{a}	$\leq$	`\leq`	$\nleqslant$	`\nleqslant`^{b}	$\succnapprox$	`\succnapprox`^{b}
$\doteq$	`\doteq`	$\leqq$	`\leqq`^{a}	$\nless$	`\nless`^{b}	$\succneqq$	`\succneqq`^{b}
$\doteqdot$	`\doteqdot`^{a}	$\leqslant$	`\leqslant`^{a}	$\nprec$	`\nprec`^{b}	$\succnsim$	`\succnsim`^{b}
$\eqcirc$	`\eqcirc`^{a}	$\lessapprox$	`\lessapprox`^{a}	$\npreceq$	`\npreceq`^{b}	$\succsim$	`\succsim`^{a}
$\eqsim$	`\eqsim`^{b}	$\lesseqgtr$	`\lesseqgtr`^{a}	$\nsim$	`\nsim`^{b}	$\thickapprox$	`\thickapprox`^{b}
$\eqslantgtr$	`\eqslantgtr`^{a}	$\lesseqqgtr$	`\lesseqqgtr`^{a}	$\nsucc$	`\nsucc`^{b}	$\thicksim$	`\thicksim`^{b}
$\eqslantless$	`\eqslantless`^{a}	$\lessgtr$	`\lessgtr`^{a}	$\nsucceq$	`\nsucceq`^{b}	$\triangleq$	`\triangleq`^{a}
$\equiv$	`\equiv`	$\lesssim$	`\lesssim`^{a}	$\prec$	`\prec`		
$\fallingdotseq$	`\fallingdotseq`^{a}	$\ll$	`\ll`	$\precapprox$	`\precapprox`^{b}		
$\geq$	`\geq`	$\lll$	`\lll`^{a}				
$\geqq$	`\geqq`^{a}						

Note 1. Labels ^{a,b} indicate `amssymb` package, font `msam` or `msbm`.

Synonyms: $\neq$ `ne`, $\leq$ `le`, $\geq$ `ge`, $\doteqdot$ `Doteq`, $\lll$ `llless`, $\ggg$ `gggtr`

1.2.6 Relation symbols: arrows

$\circlearrowleft$	`\circlearrowleft`[a]	$\leftrightsquigarrow$	`\leftrightsquigarrow`[a]	$\nwarrow$	`\nwarrow`
$\circlearrowright$	`\circlearrowright`[a]	$\Lleftarrow$	`\Lleftarrow`[a]	$\rightarrow$	`\rightarrow`
$\curvearrowleft$	`\curvearrowleft`[b]	$\longleftarrow$	`\longleftarrow`	$\Rightarrow$	`\Rightarrow`
$\curvearrowright$	`\curvearrowright`[b]	$\Longleftarrow$	`\Longleftarrow`	$\rightarrowtail$	`\rightarrowtail`[a]
$\dashleftarrow$	`\dashleftarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\rightharpoondown$	`\rightharpoondown`
$\dashrightarrow$	`\dashrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\rightharpoonup$	`\rightharpoonup`
$\downdownarrows$	`\downdownarrows`[a]	$\longmapsto$	`\longmapsto`	$\rightleftarrows$	`\rightleftarrows`[a]
$\downharpoonleft$	`\downharpoonleft`[a]	$\longrightarrow$	`\longrightarrow`	$\rightleftharpoons$	`\rightleftharpoons`[a]
$\downharpoonright$	`\downharpoonright`[a]	$\Longrightarrow$	`\Longrightarrow`	$\rightrightarrows$	`\rightrightarrows`[a]
$\hookleftarrow$	`\hookleftarrow`	$\looparrowleft$	`\looparrowleft`[a]	$\rightsquigarrow$	`\rightsquigarrow`[a]
$\hookrightarrow$	`\hookrightarrow`	$\looparrowright$	`\looparrowright`[a]	$\Rrightarrow$	`\Rrightarrow`[a]
$\leftarrow$	`\leftarrow`	$\Lsh$	`\Lsh`[a]	$\Rsh$	`\Rsh`[a]
$\Leftarrow$	`\Leftarrow`	$\mapsto$	`\mapsto`	$\searrow$	`\searrow`
$\leftarrowtail$	`\leftarrowtail`[a]	$\multimap$	`\multimap`[a]	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\nLeftarrow$	`\nLeftarrow`[b]	$\twoheadleftarrow$	`\twoheadleftarrow`[a]
$\leftharpoonup$	`\leftharpoonup`	$\nLeftrightarrow$	`\nLeftrightarrow`[b]	$\twoheadrightarrow$	`\twoheadrightarrow`[a]
$\leftleftarrows$	`\leftleftarrows`[a]	$\nRightarrow$	`\nRightarrow`[b]	$\upharpoonleft$	`\upharpoonleft`[a]
$\leftrightarrow$	`\leftrightarrow`	$\nearrow$	`\nearrow`	$\upharpoonright$	`\upharpoonright`[a]
$\Leftrightarrow$	`\Leftrightarrow`	$\nleftarrow$	`\nleftarrow`[b]	$\upuparrows$	`\upuparrows`[a]
$\leftrightarrows$	`\leftrightarrows`[a]	$\nleftrightarrow$	`\nleftrightarrow`[b]		
$\leftrightharpoons$	`\leftrightharpoons`[a]	$\nrightarrow$	`\nrightarrow`[b]		

Note 1. Labels [a,b] indicate `amssymb` package, font `msam` or `msbm`.

Synonyms: $\leftarrow$ `gets`, $\rightarrow$ `to`, $\upharpoonright$ `restriction`

1.2.7 Relation symbols: miscellaneous

$\backepsilon$ \backepsilon[b]	$\ntriangleright$ \ntriangleright[b]	$\subsetneqq$ \subsetneqq[b]
$\because$ \because[a]	$\ntrianglerighteq$ \ntrianglerighteq[b]	$\supset$ \supset
$\between$ \between[a]	$\nvdash$ \nvdash[b]	$\Supset$ \Supset[a]
$\blacktriangleleft$ \blacktriangleleft[a]	$\nVdash$ \nVdash[b]	$\supseteq$ \supseteq
$\blacktriangleright$ \blacktriangleright[a]	$\nvDash$ \nvDash[b]	$\supseteqq$ \supseteqq[a]
$\bowtie$ \bowtie	$\nVDash$ \nVDash[b]	$\supsetneq$ \supsetneq[b]
$\dashv$ \dashv	$\parallel$ \parallel	$\supsetneqq$ \supsetneqq[b]
$\frown$ \frown	$\perp$ \perp	$\therefore$ \therefore[a]
$\in$ \in	$\pitchfork$ \pitchfork[a]	$\trianglelefteq$ \trianglelefteq[a]
$\mid$ \mid	$\propto$ \propto	$\trianglerighteq$ \trianglerighteq[a]
$\models$ \models	$\shortmid$ \shortmid[b]	$\varpropto$ \varpropto[a]
$\ni$ \ni	$\shortparallel$ \shortparallel[b]	$\varsubsetneq$ \varsubsetneq[b]
$\nmid$ \nmid[b]	$\smallfrown$ \smallfrown[a]	$\varsubsetneqq$ \varsubsetneqq[b]
$\notin$ \notin	$\smallsmile$ \smallsmile[a]	$\varsupsetneq$ \varsupsetneq[b]
$\nparallel$ \nparallel[b]	$\smile$ \smile	$\varsupsetneqq$ \varsupsetneqq[b]
$\nshortmid$ \nshortmid[b]	$\sqsubset$ \sqsubset[a]	$\vartriangle$ \vartriangle[a]
$\nshortparallel$ \nshortparallel[b]	$\sqsubseteq$ \sqsubseteq	$\vartriangleleft$ \vartriangleleft[a]
$\nsubseteq$ \nsubseteq[b]	$\sqsupset$ \sqsupset[a]	$\vartriangleright$ \vartriangleright[a]
$\nsubseteqq$ \nsubseteqq[b]	$\sqsupseteq$ \sqsupseteq	$\vdash$ \vdash
$\nsupseteq$ \nsupseteq[b]	$\subset$ \subset	$\Vdash$ \Vdash[a]
$\nsupseteqq$ \nsupseteqq[b]	$\Subset$ \Subset[a]	$\vDash$ \vDash[a]
$\ntriangleleft$ \ntriangleleft[b]	$\subseteq$ \subseteq	$\Vvdash$ \Vvdash[a]
$\ntrianglelefteq$ \ntrianglelefteq[b]	$\subseteqq$ \subseteqq[a]	
	$\subsetneq$ \subsetneq[b]	

Note 1. Labels [a,b] indicate `amssymb` package, font `msam` or `msbm`.

Synonyms: $\ni$ `owns`

Remark. $\propto$ proportional to

1.2.8 Cumulative (variable-size) operators

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

1.2.9 Pairing delimiters (extensible)

See Section 1.5.5 for more information.

$\Big(\Big)$ ( )	$\Big\langle \Big\rangle$ \langle \rangle
$\Big[\Big]$ [ ]	$\Big\lceil \Big\rceil$ \lceil \rceil
$\Big\{ \Big\}$ \lbrace \rbrace	$\Big\lfloor \Big\rfloor$ \lfloor \rfloor
$\Big\lvert \Big\rvert$ \lvert \rvert	$\Big\lgroup \Big\rgroup$ \lgroup \rgroup
$\Big\lVert \Big\rVert$ \lVert \rVert	$\Big\lmoustache \Big\rmoustache$ \lmoustache \rmoustache

1.2.10 Nonpairing extensible symbols

$\Big\vert$	`\vert`	$\Big/$	`/`	$\Big\arrowvert$	`\arrowvert`	$\Big\bracevert$	`\bracevert`
$\Big\Vert$	`\Vert`	$\Big\backslash$	`\backslash`	$\Big\Arrowvert$	`\Arrowvert`		

Note 1. Using `vert`, `|`, `Vert`, or `|` for paired delimiters is not recommended. Instead, use delimiters from the list in Section 1.2.9.

1.2.11 Extensible vertical arrows

$\Big\uparrow$	`\uparrow`	$\Big\downarrow$	`\downarrow`	$\Big\updownarrow$	`\updownarrow`
$\Big\Uparrow$	`\Uparrow`	$\Big\Downarrow$	`\Downarrow`	$\Big\Updownarrow$	`\Updownarrow`

1.2.12 Math accents

In horizontal mode, plain TEX defined commonly used accents:

Type	to get	
`` \`o ``	ò	(grave accent)
`\'o`	ó	(acute accent)
`\^o`	ô	(circumflex or "hat")
`\"o`	ö	(umlaut or dieresis)
`\~o`	õ	(tilde or "squiggle")
`\=o`	ō	(macron or "bar")
`\.o`	ȯ	(dot accent)
`\u o`	ŏ	(breve accent)
`\v o`	ǒ	(háček or "check")
`\H o`	ő	(long Hungarian umlaut)
`\t oo`	o͡o	(tie-after accent)
`\c o`	o̧	(cedilla accent)
`\d o`	ọ	(dot-under accent)
`\b o`	o̲	(bar-under accent)

In math mode, we use different command:

$\acute{x}$	`\acute{x}`	$\bar{x}$	`\bar{x}`	$\vec{x}$	`\vec{x}`
$\grave{x}$	`\grave{x}`	$\breve{x}$	`\breve{x}`	$\dot{x}$	`\dot{x}`
$\ddot{x}$	`\ddot{x}`	$\check{x}$	`\check{x}`	$\widetilde{xxx}$	`\widetilde{xxx}`
$\tilde{x}$	`\tilde{x}`	$\hat{x}$	`\hat{x}`	$\widehat{xxx}$	`\widehat{xxx}`

Use `\'\i` to obtain 'í'. Moreover you can type `` \`{\i} \.{\i} \^{\i} \"{\i} ``. Assuming LATEX, the coding `` \'i \`i \.i \^i \"i `` will achieve the same results.

1.2.13 Top and bottom embellishments

$\widetilde{xxx}$	`\widetilde{xxx}`	$\overleftarrow{xxx}$	`\overleftarrow{xxx}`
$\widehat{xxx}$	`\widehat{xxx}`	$\overrightarrow{xxx}$	`\overrightarrow{xxx}`
$\overline{xxx}$	`\overline{xxx}`	xxx	`\underleftarrow{xxx}`
$\underline{xxx}$	`\underline{xxx}`	xxx	`\underrightarrow{xxx}`
$\overbrace{xxx}$	`\overbrace{xxx}`	xxx	`\overleftrightarrow{xxx}`
$\underbrace{xxx}$	`\underbrace{xxx}`	xxx	`\underleftrightarrow{xxx}`

1.2.14 Extensible arrows

$B \xrightarrow[T]{n\pm i-1} C$	`B\xrightarrow[T]{n\pm i-1}C`	`amsmath`

$\overset{\sim}{\to}$	`\buildrel\sim\over\to`	Plain TEX
$\mathop{\rightrightarrows}^{p_1^*}_{p_2^*}$	`\mathop{\rightrightarrows}^{p_1^*}_{p_2^*}`	

With out use the `amsmath` package, one can define the command `\xrightarrow` as the following

```
\renewcommand\xrightarrow[2][]{%
  \newdimen\xarwd \setbox0=\hbox{$\ \scriptstyle#1$}\setbox1=\hbox{$\ \scriptstyle#2$}
  \ifdim\wd0>\wd1 \xarwd\wd0 \else \xarwd\wd1 \fi
  \mathrel{\mathop{\hbox to\xarwd{\rightarrowfill}}\limits^{#2}_{#1}}}
```

1.3 More Math Symbols that I used

Type `texdoc encguide` for tables of math fonts.

Use `bbm` to obtain 𝟙 as `\mathbbm1`.

The following code provide an alternative for the dashed arrow `\dasharrow`: $\dashrightarrow$ provided by `amsfonts`:

```
\newcommand*\DashedArrow[1][]{\mathbin{%
  \tikz[baseline=-.25ex,-latex,dashed,#1]\draw[#1](0pt,.5ex)--(1.3em,.5ex);}}%
\def\rationalmap{\DashedArrow[->,densely dashed]}
```

Some time it is convenient to rotate a symbol, it is easy with the help of the package `graphicx`. Such as ≀≀, I got it by `\rotatebox{90}{$\approx$}`.

Ceil and floor delimiters (amsfonts)

$\ulcorner$	`\ulcorner`	$\urcorner$	`\urcorner`
$\llcorner$	`\llcorner`	$\lrcorner$	`\lrcorner`

Small fractions

```
\font\tentcrm=tcrm10
\font\seventcrm=tcrm7
\font\fivetcrm=tcrm5
\textfont8=\tentcrm
\scriptfont8=\seventcrm
\scriptscriptfont8=\fivetcrm
\mathchardef\onehalf="08BD
\mathchardef\onequater="08BC
\mathchardef\threequater="08BE
$\onehalf\onequater\threequater A^\onehalf\fam8 0123456{1\over2}$
```

½¼¾$A^{½}$0123456$\frac{1}{2}$

1.4 Math Operators

1.4.1 Named operators

These operators are represented by a multiletter abbreviation.

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

To define additional named operators outside the above list, use the `\DeclareMathOperator` command;(`amsmath`) for example,

```
\DeclareMathOperator{\rank}{rank}
\DeclareMathOperator{\esssup}{ess\,sup}
```

The star form `\DeclareMathOperator*` creates an operator that takes limits in a displayed formula, such as sup or max.

For temporally use just type `{\rm rank}V` to obtain $\mathrm{rank}V$. Or `\mathop{\rm rank}V` to get $\mathop{\rm rank}V$ more nice.

1.4.2 `\mod` and its relatives

Commands `\mod`, `\bmod` (Plain TEX), `\pmod` (Plain TEX) `\pod` deal with the special spacing conventions of "mod" notation. `\mod` and `\pod` are variants of `\pmod` preferred by some authors; `\mod` omits the parentheses, whereas `\pod` omits the "mod" and retains the parentheses.

$$\gcd(n,m \bmod n) ;\quad x\equiv y\pmod b;\quad x\equiv y\mod c ;\quad x\equiv y\pod d \tag{1.1}$$

```
\gcd(n,m\bmod n) ;\quad x\equiv y\pmod b;\quad
x\equiv y\mod c ;\quad x\equiv y\pod d
```

1.5 Math Commands

1.5.1 Matrices

Epoche

Note. The plain TEX form `\matrix{...\cr...\cr}` and the related commands `\pmatrix`, `\cases` should be avoided in LATEX (and when the `amsmath` package is loaded they are disabled).

1.5.2 Math spacing commands

When the `amsmath` package is used, all of these math spacing commands can be used both in and out of math mode.

Abbrev.	Spelled out	Example	Abbrev.	Spelled out	Example
	no space	34		no space	34
`\,`	`thinspace`	$3\,4$	`!`	`negthinspace`	$3\!4$
`\:`	`medspace`	$3\:4$		`negmedspace`	$3\mskip-\medmuskip4$
`\;`	`thickspace`	$3\;4$		`negthickspace`	$3\mskip-\thickmuskip4$
	`quad`	$3\quad4$			
	`qquad`	$3\qquad4$			

Here without using `amsmath`, I claimed

```
\def\negmedspace{\mskip-\medmuskip}
\def\negthickspace{\mskip-\thickmuskip}
```

One math unit, or `mu`, is equal to 1/18 em. In the above, `\,` skips `3mu`, and `\: \; \!` skips `4mu 5mu -3mu` respectively.

There are also three commands that leave a space equal to the height and/or width of a given fragment of LATEX/ material:

Example	**Result**
`\phantom{XXX}`	space as wide and high as three X's
`\hphantom{XXX}`	space as wide as three X's; height 0
`\vphantom{X}`	space of width 0, height = height of X

1.5.3 Nonbreaking dashes

The command `nobreakdash` suppresses the possibility of a linebreak after the following hyphen or dash. For example, if you write 'pages 1–9' as `pages 1\nobreakdash--9` then a linebreak will never occur between the dash and the 9. You can also use `nobreakdash` to prevent undesirable hyphenations in combinations like `$p$-adic`. For frequent use, it's advisable to make abbreviations, e.g.,

```
\newcommand{\p}{$p$\nobreakdash}% for "\p adic" ("p-adic")
\newcommand{\Ndash}{\nobreakdash\textendash}% for "pages 1\Ndash 9"
%    For "\n dimensional" ("n-dimensional"):
\newcommand{\n}{$n$\nobreakdash-\hspace{0pt}}
```

The last example shows how to prohibit a linebreak after the hyphen but allow normal hyphenation in the following word. (Add a zero-width space after the hyphen.)

1.5.4 Fractions and related constructions

In TeXbook Appendix B, the following delimiters are define

```
    \def\choose{\atopwithdelims()}
    \def\brack{\atopwithdelims[]}
    \def\brace{\atopwithdelims\{\}}
```

TeX has three primitives for the "generalized fraction", that is

`\overwithdelims` $\langle \text{delim}_1\rangle\langle \text{delim}_2\rangle$
`\atopwithdelims` $\langle \text{delim}_1\rangle\langle \text{delim}_2\rangle$
`\abovewithdelims` $\langle \text{delim}_1\rangle\langle \text{delim}_2\rangle\langle \text{dimen}\rangle$

Thus one writes the Legendre symbol $\left(\frac{a}{b}\right)$ by `{a\legendre b}` after defining

```
\def\legendre{\overwithdelims()}
```

1.5.5 Delimiters

Delimiter size	no size specified	`\left` `\right`	`\bigl` `\bigr`	`\Bigl` `\Bigr`	`\biggl` `\biggr`	`\Biggl` `\Biggr`
Result	$(b)(\frac{c}{d})$	$(b)\left(\frac{c}{d}\right)$	$\bigl(b\bigr)\bigl(\frac{c}{d}\bigr)$	$\Bigl(b\Bigr)\Bigl(\frac{c}{d}\Bigr)$	$\biggl(b\biggr)\biggl(\frac{c}{d}\biggr)$	$\Biggl(b\Biggr)\Biggl(\frac{c}{d}\Biggr)$

The first kind of adjustment is done for cumulative operators with limits, such as summation signs. With `left` and `right` the delimiters usually turn out larger than necessary, and using the `Big` or `bigg` sizes instead gives better results:

$$\left[\sum_i a_i\left\lvert\sum_j x_{ij}\right\rvert^p\right]^{1/p} \quad \text{versus} \quad \biggl[\sum_i a_i\Bigl\lvert\sum_j x_{ij}\Bigr\rvert^p\biggr]^{1/p}$$

```
\biggl[\sum_i a_i\Bigl\lvert\sum_j x_{ij}\Bigr\rvert^p\biggr]^{1/p}
```

The second kind of situation is that you want to make some of the delimiters slightly larger to make the nesting easier to see.

$$((a_1b_1)-(a_2b_2))\,((a_2b_1)+(a_1b_2)) \quad \text{versus} \quad \bigl((a_1b_1)-(a_2b_2)\bigr)\bigl((a_2b_1)+(a_1b_2)\bigr)$$

```
\left((a_1 b_1) - (a_2 b_2)\right)
\left((a_2 b_1) + (a_1 b_2)\right)
```

versus

```
\bigl((a_1 b_1) - (a_2 b_2)\bigr)
\bigl((a_2 b_1) + (a_1 b_2)\bigr)
```

The third kind of situation is a slightly oversize object in running text, such as $\left|\frac{b'}{d'}\right|$ where the delimiters produced by `left` and `right` cause too much line spreading. In that case `\bigl` and `\bigr` can be used to produce delimiters that are larger than the base size but still able to fit within the normal line spacing: $\bigl|\frac{b'}{d'}\bigr|$.

1.5.6 The `\text` command

If we are in math mode, usually we use `\hbox` to switch back to text mode:

> Two divisors D_1, D_2 are said to be *numerically equivalent*, written $D_1 \mathop{\equiv}_{\rm num} D_2$, or $D_1 \mathop{\equiv}_{\text{num},\mathbf{Q}} D_2$ if
>
> $$(D_1.C) = (D_2.C) \text{ for all irreducible curves } C \subset X.$$

```
Two divisors $D_1,D_2$ are said to be {\it numerically equivalent},
written $D_1\mathop{\equiv}_{\rm num}D_2$,
or $D_1\mathop{\equiv}_{\text{num},\mathbf Q}D_2$ if \[
  (D_1.C)=(D_2.C) \hbox{ for all irreducible curves } C\subset X.
\]
```

Note that `\hbox` does not work well in sub or supscript, such as $D_1 \mathop{\equiv}_{\hbox{num}} D_2$ by writing `$D_1\mathop{\equiv}_{\hbox{num}}D_2$`. A command `\text` (defined in `amsmath`, see an approach in the following) is provided, like `\hbox`, but automatically produces subscript-size text if used in a subscript.

```
\def\text#1{\mathchoice{\hbox{#1}}{\hbox{#1}}{\hbox{\scriptsize#1}}{\hbox{\tiny#1}}}
```

1.5.7 Integrals and sums

There are three LaTeX commands that can be used to influence the placement of the limits: `\limits`, `\nolimits`, `\displaylimits`. Compare

$$\int_{|x-x_z(t)|<X_0} z^6(t)\phi(x) \qquad \text{and} \qquad \int\limits_{|x-x_z(t)|<X_0} z^6(t)\phi(x)$$

```
\int_{\abs{x-x_z(t)}<X_0} ...            \int\limits_{\abs{x-x_z(t)}<X_0} ...
```

Epoche

1.5.8 Multiline subscripts and superscripts

$\mathcal{AMS}$-TeX use the command `\substack`, but this is much convenient using TeX's `\atop` command:

```
\[ \sum_{0\le i\le m\atop 0<j<n} P(i,j)\]
```

$$\sum_{0\le i\le m\atop 0<j<n} P(i,j)$$

Of course, this can be done by `\halign`, which allows more lines:

```
\baselineskip=8pt\lineskip=2pt
\[
  \sum_{\vcenter{\ialign{$\scriptstyle\nonscript#$\cr
        0<i<m\cr
        0<j<n\cr
        i\le j\cr
    }}
  }P(i,j)
\]
```

$$\sum_{\substack{0<i<m\\0<j<n\\i\le j}} P(i,j)$$

1.5.9 Changing the size of elements in a formula

You need the `displaystyle` command:

$$\frac{{\displaystyle\sum\nolimits_{n>0} z^n}}{{\displaystyle\prod\nolimits_{1\leq k\leq n} (1-q^k)}}$$

```
\frac{{\displaystyle\sum\nolimits_{n> 0} z^n}}
  {{\displaystyle\prod\nolimits_{1\leq k\leq n} (1-q^k)}}
```

There are similar commands `textstyle`, `scriptstyle`, and `scriptscriptstyle`.

Note: These commands

Right: `{\displaystyle ...}` **Wrong:** `\displaystyle{...}`

1.5.10 More Math Commands

How to reference `align` environment

```
\begin{align}
  E &= mc^2 \nonumber \\
  E &= mc^2 \label{eq1}
\end{align}
```

1.6 Math Formulae

In math mode TEX is making math lists into the following types of things:

- an atom;
- horizontal material (a rule or discretionary or penalty or “whatsit”);
- vertical material (from `\mark` or `\insert` or `\vadjust`);
- a glob of glue (from `\hskip` or `\mskip` or `\nonscript`);
- a kern (from `\kern` or `\mkern`);
- a style change (from `\displaystyle`, `\textstyle`, etc.);
- a generalized fraction (from `\above`, `\over`, etc.);
- a boundary (from `\left` or `\right`);
- a four-way choice (from `\mathchoice`).

The most important items are called *atoms*, and they have three parts: a *nucleus*, a *superscript*, and a *subscript*. There are thirteen kinds of atoms:

Ord	is an ordinary atom like ‘x’;
Op	is a large operator atom like ‘$\sum$’;
Bin	
Rel	
Open	
Close	
Punct	
Inner	is an inner atom like ‘$\frac{1}{2}$’;
Over	
Under	

Acc	
Rad	
Vcent	is a vbox to be centered, produced by `\vcenter`.

When you say `\hbox`{...} in math mode, an Ord atom is placed on the current math list, with the hbox as its nucleus. You can experiment with `\showlists` to discover how things arranged. As soon as math mode ends, (i.e., when the closing '`$`' occurs), TEX dismantles the current math list and converts it into a horizontal list. You can see "before and after" representations of such math typesetting by ending a formula with '`\showlists$\showlists`'; the first `\showlists` will display the math list, and the second will show the (possibly complex) horizontal list that is manufactured from it.

1.7 More about Math

1.7.1 Math Fonts

Plain TEX use `cmr10` as `\textfont0` one use `\rm` to use it, and `cmmi10` for `\textfont1` as default, also a command `\mit` is provided. Others, `cmsy10` for `\textfont2` by using `\cal`, and `cmex10` for `\textfont3`.

The symbols in a math formula fall into seven different classes.

Class number	**Mnemonic**	**Description (part of speach)**	**Examples**
0	Ord	simple/ordinary ('noun')	A 0 Φ ∞
1	Op	prefix operator	$\sum \prod \int$
2	Bin	binary operator (conjunction)	$+ \cup \wedge$
3	Rel	relation/comparison (verb)	$= < \subset$
4	Open	left/opening delimiter	$(\, [\, \{ \, \langle$
5	Close	right/closing delimiter	$) \,] \, \} \, \rangle$
6	Punct	postfix/punctuation	. , ; !

NOTE 1. Symbols of class 2 (Bin), notably the minus sign $-$, are automatically printed by LATEX as class 0 (no space) if they do not have a suitable left operand — e.g., at the beginning of a math formula or after an opening delimiter.

NOTE 2. The spacing for a few symbols follows tradition instead of the general rule: although / is (semantically speaking) of class 2, we write $k/2$ with no space around the slash rather than $k \mathbin{/} 2$. And compare `p|q` $p|q$ (no space) with `p\mid q` $p \mid q$ (class-3 spacing).

Recall from the TEXbook,

```
\def\joinrel{\mathrel{\mkern-3mu}}
\def\relbar{\mathrel{\smash-}} \def\Relbar{\mathrel=}
\def\longrightarrow{\relbar\joinrel\rightarrow}
```

we can use `\mathop{\gets\joinrel\to}\limits^{\rm bij}` to get

$$\left\{\begin{matrix}\text{homogeneous}\\ \text{quadratic polys.}\end{matrix}\right\} = \left\{\begin{matrix}\text{quad. forms}\\ k^3 \to k\end{matrix}\right\} \overset{\text{bij}}{\longleftrightarrow} \left\{\begin{matrix}\text{symmetric bilinear}\\ \text{forms on } k^3\end{matrix}\right\}$$

These command works since TEX does not put extra space when two relations are adjacent in a math formula. Similarly, using `\mathrel{\lhook\joinrel\relbar\joinrel\to}` to get $\hookrightarrow$.

1.7.2 Active character

The following code works

```
$$\catcode`p=13\def p{\partial}
  \det(pf_i/pX_j)(x)\ne0.
$$
```

$$\det(\partial f_i/\partial X_j)(x) \neq 0.$$

`\prime`
It works for `$a_1'^\alpha$`, $a_1'^\alpha$, but `$a_1^\alpha'$` will causes TEX to complain double supscript. See the definition code of `'` how it eats the following `'`s and the following `^{...}`.

```
{\catcode`\'=\active \gdef'{^\bgroup\prim@s}}
\def\prim@s{\prime\futurelet\next\pr@m@s}
\def\pr@m@s{\ifx'\next\let\nxt\pr@@@s \else\ifx^\next\let\nxt\pr@@@t
\else\let\nxt\egroup\fi\fi \nxt}
\def\pr@@@s#1{\prim@s} \def\pr@@@t#1#2{#2\egroup}
```

1.7.3 Line breaking

When you have formulas in a paragraph, TEX may have to break them between lines. This is necessary evil, something like the hyphenation of words; we want to avoid it unless the alternative is worse.

A formula will be broken only after a relation symbol like $=$ or $<$ or $\rightarrow$, or after a binary operation symbol like $+$ or $\times$, where the relation or binary operation is on the "outer level" of the formula (i.e., not enclosed in `{...}` and not part of an '`\over`' construction).

`\allowbreak`
This command gives permission to break a formula at the point in the outer level.
Example: `$(\mathcal O_X(D),\allowbreak |D|,\allowbreak s_D)$` produce

> Any Cartier divisor D on a scheme X determines a pseudo-divisor $(\mathcal{O}_X(D), |D|,$
> $s_D)$ on X.

`\*`
A "discretionary multiplication sign" is allowed in formulas: something like `\-`; namely, a line break will be allowed at that place, and a $\times$ sign leaves there.
Example:

```
Let $c = a\*b$. In the case that $c=0$ or $c=1$, let
$\Delta$ be $(\hbox{the smallest $q$})\*(\hbox{the
largest $q$})$ in the set of approximate $\tau$-values.
```

produces:

> Let $c = ab$. In the case that $c = 0$ or $c = 1$, let $\Delta$ be (the smallest $q$)(the largest $q$)
> in the set of approximate $\tau$-values.

`\-`
This command is usually equivalent to '`\discretionary{-}{}{}`'; and TEX automatically insert a '`\discretionary{}{}{}`' after each '`-`'. Also

```
\def\*{\discretionary{\thinspace\the\textfont2\char2}{}{}}
```

1.7.4 Invisible character

The TEX book has the following definition

```
{\catcode`\^^Z=\active \gdef^^Z{\not=}} % ^^Z is like \ne in math
```

Here `^^Z` can formed by the three characters, or the actual character as typed by `vim` Ctrl + V following Ctrl + Z.

1.7.5 Double accents

Define `\def\Ahat{{\hat A}}`.

You can put accents on top of accents, making symbols like $\hat{\hat{A}}$ that might cause a mathematician to squeal with ecstasy. This is produce by `\skew6\hat\Ahat`, where `\skew` is defined as

```
\def\skew#1#2#3{{\muskip0=#1mu \mkern.5\muskip0
  #2{\mkern-.5\muskip0{#3}\mkern.5\muskip0}\mkern-.5\muskip0}{}}
```

which first shifted right and makes TEX ready in this position to hat up the symbols, but secretly shifted $\hat{A}$ back. The number '6' in this example was chosen by trial and error, it fits with A but fails with B.

Recall how Plain TEX define `\cong`:

```
\def\cong{\mathrel{\lower.5pt\vbox{\lineskiplimit\maxdimen
  \lineskip-.5pt\ialign{$\hfil##\hfil$\cr\sim\cr=\cr}}}}
```

We can define $\overset{\sim}{\to}$ in a similar way:

```
\def\simto{\mathrel{\lower.5pt\vbox{\lineskiplimit\maxdimen
  \ialign{$\hfil##\hfil$\cr\scriptstyle\sim\cr\to\cr}}}}
```

And $\underset{\longrightarrow}{\lim}$ in such a way:

```
\def\injlim{\mathop{\vtop{\lineskiplimit\maxdimen%
  \ialign{$\hfil##\hfil$\cr\rm lim\cr\scriptstyle\longrightarrow\cr}}}}
```

I define the following command to produce $\rightrightarrows$:

```
\def\longrightrightarrows{\mathrel{
  \lineskiplimit=-\maxdimen \baselineskip0pt
  \def\joinrel{\mathrel{\mkern-4mu}}
  \def\lrrars{\hbox{$\scriptstyle\relbar\joinrel\relbar\joinrel\to$}}
  \vtop{\ialign{##\cr\raise2.4pt\lrrars\cr\lower1pt\lrrars\cr}}
}}
```

Double dual with double hats The double dual $\widehat{\widehat{G}}$ is obtained by `\skew3\widehat{\widehat G}`. The number '3' here was chosen by trial and error.

```
\def\skew#1#2#3{{\muskip0=#1mu \mkern. 5\muskip0
    #2{\mkern-.5\muskip0{#3}\mkern.5\muskip0}\mkern-.5\muskip0}{}}
```

1.7.6 Math code

catcode

Escape character	`\`	category 0,
Beginning of group character	`{`	category 1,
End of group character	`}`	category 2,
Math shift character	`$`	category 3,
Alignment tab character	`&`	category 4,
End of line	`<return>`	category 5,
Parameter	`#`	category 6,
Superscript	`^`	category 7,
subscript	`_`	category 8,
Ignored character	`<null>`	category 9,
Space	␣	category 10,
Letter	`[A-Za-Z]`	category 11,
Other character		category 12,

Active character		category 13,
Comment character	`%`	category 14,
Invalid character		category 15.

mathcode

```
\mathcode`<="313C
```

treat '<' in math mode as a relation (class 3) found in position `"3C` of family 1.

0 stands for Ordinary class, can be obtained by `\mathord`. 1 stands for Large operator class, can be obtained by `\mathop`. 2 stands for Binary operation class, can be obtained by `\mathbin`. 3 stands for Relation class, can be obtained by `\mathrel`. 4 stands for Opening class, can be obtained by `\mathopen`. 5 stands for Closed class, can be obtained by `\mathclose`. 6 stands for Punctuation class, can be obtained by `\mathpunct`. 7 stands for Variable family class.

INITEX has claimed that `\mathcode` $x = x$ (in class 0 and family 0). for x that is neither letters nor digits. Note that TEX defines `\textfont` 0=`\tenrm`, `\scriptfont` 0=`\sevenrm` and `\scriptscriptfont` 0=`\fivenrm`. For x a digits, `\mathcode` x = x + "7000; and for x a letter, `\mathcode` x=x+"7100.

TEX will look at the mathcode only for characters has catcode 11 (letter) or 12 (other), or `\char` ⟨number⟩.

`\mathcode` has a special value `"8000`, causes the character to behave as if it has catcode 13 (active).

```
\let\chdot\bullet
\mathcode`\"="8000
{\catcode`\"=\active \gdef"{^\bgroup\chdOt}}
\def\chdOt{\chdot\futurelet\next\chDOt}
\def\chDOt{\ifx^\next\let\nxt\chDOT \else\let\nxt\egroup\fi \nxt}
\def\chDOT#1#2{#2\egroup}
```

```
\mathchardef\sum="1350
\mathopen\mathchar"1234 is equivalent to \verb|\mathchar"4234
\def\bigl{\mathopen\big} \def\bigm{\mathrel\big}
\def\big#1{{\hbox{$\left#1\vbox to 8.5pt{}\right.\n@space$}}}
\def\n@space{\nulldelimiterspace=0pt \m@th}
```

		Right atom							
		Ord	Op	Bin	Rel	Open	Close	Punct	Inner
	Ord	0	1	(2)	(3)	0	0	0	(1)
	Op	1	1	*	(3)	0	0	0	(1)
	Bin	(2)	(2)	*	*	(2)	*	*	(2)
Left	Rel	(3)	(3)	*	0	(3)	0	0	(3)
atom	Open	0	0	*	0	0	0	0	0
	Close	0	1	(2)	(3)	0	0	0	(1)
	Punct	(1)	(1)	*	(1)	(1)	(1)	(1)	(1)
	Inner	(1)	1	(2)	(3)	(1)	0	(1)	(1)

Here 0, 1, 2, and 3 stand for no space, thin space, medium space, and thick space, respectively; the table entry is parenthesized if the space is to be inserted only in display and text styles, no in script and scriptscript styles.

delcode

```
\delimiter"426830A
```

stands for class 4 with large symbol in `268` and normal symbol in `30A`.

INITEX has mad all delimiter codes equal to −1, but there's an exception: The value `\delcode`\.=0` has been prespecified, so that '.' stands for a "null delimiter." Plain format sets up the following nine values

```
\delcode`\(="028300     \delcode`\/="02F30E     \delcode`\)="029301
\delcode`\[="05B302     \delcode`\|="26A30C     \delcode`\]="05D303
\delcode`\<="26830A     \delcode`\\="26E30F     \delcode`\>="26930B
```

It's important to note that `\delcode`\{` and `\delcode`\}` have been left equal to -1.

I fond the following codes are useful.

```
\mathcode`\^^A="8000
\mathcode`\^^V="8000
\mathcode`\^^X="8000
{\catcode`\^^A=13 \gdef^^A{^{\rm an}}}
{\catcode`\^^V=13 \gdef^^V{^\vee}}
{\catcode`\^^X=13 \gdef^^X{^\times}}
```

Notably, `^^M` is used as ⟨return⟩ which has code 13 that TEX normally places at the right end of every line of your input file. By changing the category of `^^M` you can obtain useful special effects, as we shall see later.

The control code `^^I` is also of potential interest, since it's the ASCII ⟨tab⟩. Plain TEX makes ⟨tab⟩ act like a blank space.

There's also a special convention in which `^^` is followed by two "lowercase hexadecimal digits," `0-9` or `a-f`. With this convention, all 256 characters are obtainable in a uniform way, from `^^00` to `^^ff`.

I also want to define

```
\mathcode`\^^N="202E
```

using as the intersection dot, but you need adding `\catcode`\^^N=12`, otherwise you'll get an error `Unicode character ^^N (U+000E) not set up for use with LaTeX.`

Chapter 2

Fonts

2.1 Define New Fonts

How to use the symbol “ə” (schwa) using TEX? Assume Plain TEX has been equipped, you run

```
pdftex testfont
```

and at the

```
Name of the font to test =
```

prompt you answer `tipa10` and at the next prompt you write `\table\bye`, a table of the font will be printed in the file `testfont.pdf`. You can recognize that “ə” is at position `0x40`, so write

```
    \font\tenipa=tipa10
    \def\schwa{{\tenipa\char"40}}
```

will do. Alternatively, you could also type `{\the\tenipa@}` to obtain ə.

In the position `0x12`, you find the symbol ˽ but typing `\the\tenipa^^R` may create an error, this is because a invisible character has category 15 (invalid), which is not read to be use, so you have to claim the following first

```
\catcode`\^^R=12
```

2.2 Fonts

2.2.1 `\emph` and `\em`

`emph` is like e.g. `textit` a command with an argument. `em` is the ”switch” variant, comparable to `itshape`. `em` is not an outdated TeX or LaTeX2.09 command but a real LaTeX2e command. Actually `emph` is defined through em:

```
\DeclareTextFontCommand{\emph}{\em}
```

`em` is useful for long texts (`emph` e.g. doesn’t allow the argument to contain a `par`). The commands differ (like the `\textit`/`\itshape`) in their handling of the italic correction:

```
abc\emph{lll}lll  \textit{lll}llll

abc{\em lll}lll  {\itshape lll}llll

abc{\em lll\/}lll
```

abc*lll*lll *lll*llll
abc*lll*lll *lll*llll
abc*lll*lll

2.2.2 Fonts selection in TEX

Example:

```
${\cal XYZ}\quad
{\mit AaBb\Gamma\Delta\Sigma}\quad
{\oldstyle 0123456789}$
```

produces: $\mathcal{XYZ}$ $\quad$ $AaBb\mathit{\Gamma\Delta\Sigma}$ $\quad$ 0123456789

NOTE.In LATEX I have to add definitions to make it work:

```
\font\teni=cmmi10
\def\oldstyle{\fam1 \teni}
```

All characters in math mode belong to one of sixteen families of fonts, each of which consists of three fonts, for example

```
    \textfont0 = \tenrm
    \scriptfont0 = \sevenrm
    \scriptscriptfont0 = \fiverm
```

Incidentally, TEX uses `\nullfont`, which contains no characters, for each family member that has not been defined.

Classes 7 is equivalent to classes 0 unless `\fam` (-1 as beginning of a formula) has been given a new value. Plain TEX changes `\fam` to 0 when the user types '`\rm`'.

INITEX initializes the mathcodes of all letters A to Z and a to z so that they are symbols of class 7 and family 1.

Plain TEX use family 1 for math italic letters, family 2 for ordinary math symbols, family 3 for large symbols.

LATEX defined families 0-7.

Plain TEX puts text italic in family 4, slanted roman in family 5, blod roman in famil 6, and typewrite type in family 7. But it's not what LATEX do. And it seems numbers has fam 7.

2.2.3 Fonts selection in LATEX

The following declarations let you choose between three pre-defined font families:

`rmfamily`	roman (i.e., serifed) font family
`sffamily`	sans serif font family
`ttfamily`	`monospaced (``typewriter'') font family`

Within each font family, the following declarations select the "series" (i.e., darkness or stroke width),

`mdseries`	regular
`bfseries`	**bold**

and the "shape" (i.e., the form of the letters):

`upshape`	upright
`slshape`	*slanted*
`itshape`	*italic*
`scshape`	CAPS AND SMALL CAPS

These commands are "declarations," i.e., they remain in effect until the end of the current group or environment. For each declaration there exists a text generating command as a counterpart; it typists only its argument in the desired style, e.g. `textsf` corresponds to `sffamily`. Return the document font by typing `normalfont`.

2.2.4 Families

Using `ttfamily` one can gets the visible space ␣ by typing `{\tt\char32}`.

2.2.5 Roman numbers

TeX has token-producing operations `\string`, `\number` and `\romannumeral`. If you write `\number` ⟨number⟩, you get the decimal equivalent of the ⟨number⟩; and if you write `\romannumeral` ⟨number⟩, you get the number expressed in lowercase roman numerals. For example, '`\romannumeral` 124' produces '`xxiv`, a list of four tokens each having category 12. Using '-15' you'll obtain '`-15`'; and if register `\count5` holds the value 316, then '`\number\count5`' produces '316'.

The twin operations `\uppercase`{⟨token list⟩} and `\lowercase`{⟨token list⟩} convert characters to their "uppercase" or "lowercase" equivalents. Here's how: Each of the 256 possible characters has two associated values called the `\uccode` and the `\lccode`. Conversion to uppercase means that a character is replaced by its `\uccode` value, unless the `\uccode` value is zero (when no change is made).

Remark. TeX performs the `\uppercase` and `\lowercase` transformations in its stomach, but the `\string`, `\number`, `\romannumeral` and `\csname` operations are carried out en route to the stomach (like macro expansion).

Example: Use `\uppercase\expandafter{\romannumeral23}` for example to obtain uppercase roman numerals.

You can inhibit the expansion of a control sequence that would otherwise be expanded by using `\noexpand`. You can postpone the expansion of a control sequence by using `\expandafter`. The command

`\expandafter` ⟨token$_1$⟩⟨token$_2$⟩

tells TeX to expand ⟨token$_1$⟩ according to its rules for macro expansion *after* it has expanded ⟨token$_2$⟩ by one level. In the example above, TeX reads `\romannumeral23` first, and reads `{` next which starts the operation of the `\uppercase` operator.

Example: After define `\def\aa{xyz}\tt` the command will produce as in the following.

```
[\string\aa]                                 [\aa]
[\expandafter\string\aa]                     [xyz]
[\expandafter\string\csname TeX\endcsname]   [\TeX]
```

2.3 The amsfonts collection

The following notes comes from '`texdoc amsfonts`'.

Packages for using these fonts with LaTeX:

– `amsfonts`, a package for using the fonts `msam`, `msbm`, and `eufm` in LaTeX

– `amssymb`, this package will call for `amsfonts`, then names for all the math symbols in the fonts `msam` and `msbm`

– `eucal`, a package for using the Euler script font `eusm`

Macro files for using these fonts with plain TeX:

– `amssym.tex`, a file defining names for the symbols in fonts `msam` and `msbm`, first calling for `amssym.def`

– `amssym.def`, a file that loads the fonts `msam`, `msbm`, and `eufm` and defines some control sequences required by `amssym.tex`

You put '`\input amssym`' in your main file, then `tex` will load the following:

```
(/usr/share/texlive/texmf-dist/tex/plain/amsfonts/amssym.tex
(/usr/share/texlive/texmf-dist/tex/plain/amsfonts/amssym.def))
```

There, it it defined that

```
\def\Bbb#1{{\fam\msbfam\relax#1}}
\def\frak#1{{\fam\eufmfam\relax#1}}
\let\goth\frak
```

Chapter 3

Box Construction

3.1 `\hbox` and `\vbox`

The reference is `https://www.ctan.org/pkg/texbook`.

Let's see the procedure of building an hbox. Suppose that we are given a horizontal list H. To begin with, TeX assumes a reference point and then the items are placed one by one from left to right with their baseline coincides. If a box is moved up or down with `\raise` or `\lower`, TeX uses its reference point before the move when placing it.

Similarly, TeX builds a vbox from a vertical list V by assuming a temporary reference point and then appending the items in V one by one from top to bottom with their reference point aligned vertically. `\moveleft` or `\moveright` applies after the alignment. As each box other than the first one is added, TeX puts interline glue just above it.

3.1.1 rules

To make a rule box, you type '`\hrule`' in vertical mode or '`\vrule`' in horizontal mode, for example, if

```
\vrule height4pt width3pt depth2pt
```

appears in the middle of a paragraph, TeX will typeset the black box '▮'. If you specify a dimension twice, the second specification overrules the first. If you leave a dimension unspecified, you get the following by default:

	`\hrule`	`\vrule`
`width`	`*`	`0.4pt`
`height`	`0.4pt`	`*`
`depth`	`0.4pt`	`*`

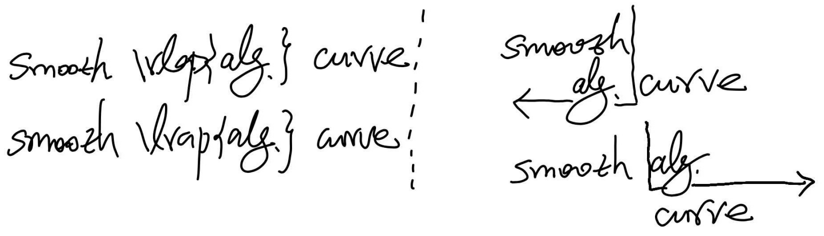
here '`*`' means that the actual dimension depends on the content; the rule will extend to the boundary of the smallest box or alignment that encloses it.

If you specify all three dimensions of a rule, there's no essential difference between `\hrule` and `\vrule`, beside you must call it an `\hrule` if you are in a vertical list, and you must call it a `\vrule` if you are in a horizontal list. If you say `\vrule` in vertical mode, TeX starts a new paragraph; if you say `\hrule` in horizontal mode, TeX stops the current paragraph and returns to vertical mode.

$*$`\lastbox` A primitive operation that looks for the last hbox or vbox in the current horizontal list or vertical list, and remove it from the list as it becomes the `\lastbox`. Say you put an `\lastbox` in the middle of a paragraph follows some `\hbox`, it probably seems that nothing happened, since it just removed the lastbox and then putted it back. In math modes, `\lastbox` is void. At the beginning of a paragraph, '`{\setbox0=\lastbox}`' removes the indentation box. Note that the `\lastbox` doesn't works in vertical mode when the main vertical list has been entirely contributed to the current page.

∗`\unskip` This operation is somethings like `\lastbox`, except that it applies to glue instead of to boxes. If the last thing on the current list is a glue item (or leaders), it is removed. You can't remove glue from the current page by using `\unskip` in vertical mode, but you can say '`\vskip-\lastskip`', which has almost the same effect.

3.1.2 `\rlap` and `\llap`



$$\overbrace{(a_1,\dots,a_r)\underbrace{\begin{pmatrix} b_{11} & \cdots & b_{1r}\\ \vdots & & \vdots \\ b_{r1} & \cdots & b_{rr}\end{pmatrix}}^{=0}\begin{pmatrix} y_1\\ \vdots \\ y_r\end{pmatrix}}_{=(x_i)} = 0$$

Exercise. How to typeset like this
  Let $X_1 = V_1/U_1$, $g_1 = \dim X_1$,
    $X_2 = V_2/U_2$, $g_2 = \dim X_2$,
    V_i complex vector spaces, U_i lattices. Then every algebraic homomorphism $f\colon X_1 \to X_2$ lifts to a complex-analytic homomorphism $\tilde{f}\colon V_1 \to V_2$.

3.2 Break and Fill

3.2.1 Paragraph

`\everypar` When TEX enters horizontal mode, it will interrupt its normal scanning to read tokens that were predefined by the command `\everypar={`⟨token list⟩`}`. For example, suppose you have said '`\everypar={A}`'. If you type '`B`' in vertical mode, TEX will shift to horizontal mode (after contributing `\parskip` glue to the current page), and a horizontal list will be initiated by inserting an empty box of width `\parindent`. Then TEX will read '`AB`', since it reads the `\everypar` tokens before getting back to the '`B`' that triggered the new paragraph. Note that the display math mode does not open a new paragraph, thus no `\everypar` inference.

3.2.2 line break

We have three commands that work the same to break lines:

```
\\ (two backslashes)
\newline
\hfil \break
```

Those commands fill the line first and then begin a new line. Sometimes we want TEX to break at a point (for example when the next word is too long), then there are commands

```
\break         =    \penalty -1000
\allowbreak    =    \penalty 0
\nobreak       =    \penalty 1000
```

They are in LaTeX because they are in Plain TeX and the original LaTeX loaded `lplain.tex` which was a slightly edited copy of `plain.tex`.

The difference between `\break` and `\linebreak` is that

```
abc\break def
```

and

```
abc\par\break def
```

mean two very different things: the first one breaks the line, the second introduces a *page* break. With LaTeX's command `\linebreak` you'd get a line break in the first case and an error in the second one.

LaTeX sets default values for built in penalties:

```
\binoppenalty=700
\brokenpenalty=100
\clubpenalty=150
\displaywidowpenalty=50
\exhyphenpenalty=50
\floatingpenalty=20000
\hyphenpenalty=50
\interlinepenalty=0
\linepenalty=10
\postdisplaypenalty=0
\predisplaypenalty=10000
\relpenalty=500
\widowpenalty=150
```

By using `latexdef linebreak` one can find out that

```
\linebreak[0]   =   \penalty 0
\linebreak[1]   =   \penalty -51
\linebreak[2]   =   \penalty -151
\linebreak[3]   =   \penalty -301
\linebreak[4]   =   \linebreak   =   \penalty -10000
```

3.2.3 page break

The difference between `clearpage` and `newpage` is that `clearpage` command ends the current page and causes all figures and tables that have so far appeared in the input to be printed. While `newpage` just break the page and leave loaded floaing object to the next page.

A second difference is `clearpage` actually always starts a new "page" while `newpage` only ends the current column — and that is a big difference in twocolumn mode.

3.2.4 The spring length `fil`

`fil` is actually a measure and one of the few keywords of TeX, some other being `plus` , `height`, `pt` etc., i.e., keywords and not macros. In the original Pascal program these measures can result in infinities and I guess that is why Knuth keeps on referring to them as infinite glues.

The coefficient before `fil(ll)` should be a decimal number less than 16384 in absolute value (there must be one). The minimum non-zero value is `2^(-16)=0.000015`, so saying `0.000014filll` is equivalent to say `0filll` (and useless, of course).

To understand the states: `fill` overwrites `fil`, think of `fil` like x for x to infinity, then `fill` is x^2 (or maybe x^x?).

They only work if the existing space is predefined, e.g. using `\vbox to ...{...}` or `\hbox to ...{...}`. Thus one can thinks of them as springs which has length equal to predefined space (or long enough since they can be easily compressed). And put two `fil` together, they will have equal length, because they have equal elasticity, and they do not give any press on other box.

One may fill confuse with choocing `fil` or `fill`. According to some people's advise, normal formatting macros should use `fil` while the user or package author can add a `fill` stretch manually which overwrites the normal `fil`s. This way the normal formatting macros can still be used in this cases.

TEX has some primitives equivalent to glue specifications:

```
\hfil    = \hskip 0pt plus 1fil minus 0pt
\hfilneg = \hskip 0pt plus -1fil minus 0pt
\hss     = \hskip 0pt plus 1fil minus 1fil
\hfill   = \hskip 0pt plus 1fill minus 0pt
```

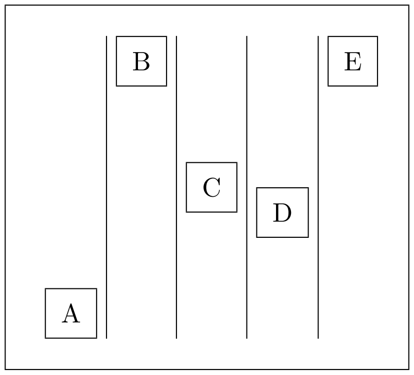
And similarly `\vfil`, `\vfill`, i.e..

The following example shows how those commends work:

```
\def\abox#1#2{%
    \vbox{\hrule
        \hbox
            {\vrule
            \hskip#1
            \vbox{\vskip#1\relax
                #2%
                \vskip#1}%
            \hskip#1
        \vrule}
    \hrule}
}

\def\testbox#1{\abox{0.2cm}{\hbox{#1}}}

\hskip 2cm
\abox{0.4cm}{\hbox{
\vbox to 4cm{\vfil\testbox A}
\vrule\ \vbox to 4cm{\testbox B\vfil}
\vrule\ \vbox to 4cm{\vfil \testbox C \vfil}
\vrule\ \vbox to 4cm{\vskip 0pt plus 3fil \testbox D \vfil\vfil}
\vrule\ \vbox to 4cm{\vfil \testbox E \vfill}}}
```



3.2.5 Space

Length

pt	point	1in=72.27pt	pc	pica	1pc=12pt
in	inch	1in=25.4mm	bp	big point	1in=72bp
cm	centimetre		mm	millimetre	
dd	didot point	1157dd=1238pt	cc cicero	1cc=12dd	
sp	scaled point	65536sp=1pt			

Synopsis, one of `\bigskip \medskip \smallskip` produce an amount of vertical space. These commands are fragile. Their work can be explained as

```
\bigskip = \vspace{12pt plus 4pt minus 4pt}
\medskip = \vspace{6pt plus 2pt minus 2pt}
\smallskip = \vspace{3pt plus 1pt minus 1pt}
```

when using in article or book documentclass i.e..

`smallbreak`, `medbreak` and `bigbreak` do almost the same, but they remove a preceding vertical space if less than what they would insert; they also terminate a paragraph and tell TeX that they mark a good point where a page break may happen. `bigskip` work between paragraphs while `bigbreak` will create new a paragrph.

`enskip`, `quad`, `qquad` leave a horizontal space of respectively `.5em`, `1em` and `2em`.

`\,` and `\!` leave respectively a thin space and its negative. For example in `D.\,E.~Knuth` [D. E. Knuth] (that somebody prefers to `D.~E.~Knuth`; [D. E. Knuth] note that `D.E.~Knuth` [D.E. Knuth] is wrong). Note that Plain TEX puts extra space at the end of a sentence; furthermore, it automatically increases the stretchability (and decreases the shrinkability) after punctuation marks. In some situation, such as 'Proc. Amer. Math. Soc.', where periods come in the middle of a sentence, you should say

```
Proc.\ Amer.\ Math.\ Soc.
```

Granted that this input looks a bit ugly, it makes the output look right. It's one of the things we occasionally must do when dealing with a computer that tries to be smart.

Manuals of style will tell you that the abbreviations 'e.g.' and 'i.e.' should always be followed by commas, never by spaces, so those particular cases shouldn't need any special treatment.

3.2.6 Insertion

An *insertion* is a vertical list containing material to be inserted into a page when TEX has finished building that page. [1]

3.2.7 Line wrap and word wrap

See wiki.

[1] The primitive `\insert` cause TEX to change page break so that it will leave room on the page for the inserted material. Later, when TEX actually breaks the page, it divides the inserted material into two groups: the material that fits on the current page and the material that doesn't. The material that fits on the page is placed into box registers, one per insertion, and the material that doesn't fit is carried over to the next page. This procedure allows TEX to do such things as distributing parts of a long footnote over several consecutive pages.

3.3 Modes

There are six modes:

- Vertical mode. [Building the main vertical list, from which the pages of output are drived.]
- Internal vertical mode. [Building a vertical list for a vbox.]
- Horizontal mode. [Building a horizontal list for a paragraph.]
- Restricted horizontal mode. [Building a horizontal list for an hobx.]
- Math mode. [Building a mathematical formula to be placed in a horizontal list.]
- Display math mode. [Building a mathematical formula to be placed on a line by itself, temporarily interrupting the current paragraph.]

Now suppose TEX is in vertical or internal vertical mode, the first token of a new paragraph (as any character, or `\char` or `\accent` or `\hskip` or `\␣` or `\vrule` or math shift `$`) changes the mode to horizontal for the duration of a paragraph.

You can also tell TEX explicitly to go into horizontal mode by saying `\noindent` or `\indent`. The `\noindent` command simply ask to enter horizontal mode if the current mode is vertical or internal vertical; it has no effect in horizontal mode. `\indent` do similarly, but it creates an empty box with width `\parindent`. When in horizontal mode, this command merely produces the empty box. When TEX is in vertical mode and sees a letter or some other inherently horizontal command, it starts a paragraph by switching to horizontal mode, doing an `\indent`, and processing the horizontal command.

Here is a little trick using the property that `\edef` will expand before defining.

```
\edef\b#1#2{\ifmmode#1\else#2\fi}
```

gives a result equivalent to '`\def\b#1#2{#1}`' if in math mode, else '`\def\b#1#2{#2}`'.

struc A *strut* is an invisible box whose width is zero and whose height and depth are slightly more than those of a "normal" line of type in the context. A `\strut` in plain TEX has height 8.5 pt and depth 3.5 pt, while a `\mathstrut` has the height and depth of a left parenthesis in the current style.

```
\vbox{\hsize = 3in \raggedright\offinterlineskip
\strut Here is the first of two paragraphs that we're
setting in a much narrower line length.\strut}
\vbox{\hsize = 3in \raggedright\offinterlineskip
Here is the first of two paragraphs that we're
setting in a much narrower line length.}
```

produces.
Here is the first of two paragraphs that we're
setting in a much narrower line length.
Here is the first of two paragraphs that we're
setting in a much narrower line length.

3.4 Alignments

3.4.1 Alignment displays

$$\texttt{\$\$}\langle\text{assignments}\rangle\texttt{\textbackslash halign}\{\langle\text{alignment}\rangle\}\langle\text{assignment}\rangle\texttt{\$\$}\ .$$

Here the ⟨assignments⟩ are optional things like parameter changes that do not produce any math lists. Such as

```
$$\tabskip=1em\halign{&\hfil#\hfil#\cr
    ...
}$$
```

This produce a normal `\halign` vertical list, expect shifting right by `\displayindent` which is initially zero.

Chapter 4

Macros and Primitives

4.0.1 The primitive `\let`

The syntax for `\let` is

`\let`⟨control sequence⟩⟨equals⟩⟨one optional space⟩⟨token⟩

so an `=` and a space are consumed while parsing. ⟨equals⟩ here is TEXbook syntax for any number of optional spaces, optionally followed by one `=`. Here is how LATEX define a space token

`\def\:{\let\@sptoken=␣}\:␣`

Since `\:␣` expands to `\let\@sptoken=␣␣` and the `=` and one optional ␣ are discarded so `\@sptoken` is let to be ␣.

NOTE.If the ⟨token⟩ in a `\let` is a single character—i.e., if it is a (character code, category code) pair—then the control sequence will behave to a certain extent like that character; but there are some differences. For example, after '`\let\zero=0`' you can't use `\zero` in a numerical constant, because TEX requires the tokens in a numerical constant to be digits, after macro expansion; `\zero` is not a macro, so it doesn't expand.

4.0.2 Verbatim listing

Plain TEX includes a macro

```
> \dospecials=macro:
->\do\ \do\\\do\{\do\}\do\$\do\&\do\#\do\^\do\_\do\%\do\~.
```

So it becomes easy to change all of the special characters to category 12 (other):

```
\def\uncatcodespecials{\def\do##1{\catcode‘##1=12 }\dospecials}
```

Say out code is stored in a file `story.tex`, we would use the following definition to typeset listings:

```
\def\listing#1{\par\begingroup\setupverbatim\input#1 \endgroup}
```

Notice that the commands '`\input#1 \endgroup`' will not be listed verbatim, even though they follow `\setupverbatim`, since they entered TEX's reading mechanism when the `\listing` macro was expanded (i.e., before the verbatim business was actually set up).

But what should `\setupverbatim` do?

```
\newcount\lineno % the number of file lines listed
\def\setupverbatim{\tt \lineno=0
  \obeylines \uncatcodespecials \obeyspaces
  \everypar{\advance\lineno by1 \llap{\sevenrm\the\lineno\ \ }}}
{\obeyspaces\global\let =\ } % let active space = control space
```

In theory, this seems like it ought to work; but in practice, it fails in two ways.

Instead of listing a file verbatim, you might want to define a `\verbatim` macro. It's somewhat dangerous to change category codes, because TEX stamps the category on each character when that character is first read from a file. Thus, if `\verbatim` were defined by a construction of the form `long\def\verbatim#1{`⟨something⟩`}`, argument `#1` would already be converted to a list of tokens when ⟨something⟩ starts. The alternative is to change category codes before scanning the argument to `\verbatim`:

One of the problems with verbatim mode is that it's hard to stop; if we turn off all of TEX's normal control capabilities, we end up "painting ourselves into a corner" and reaching a point of no return. The `\listing` macro was able to solve this problem because the end of a file brings an old token list back to life. However, here's an approach that typesets everything between `\beginverbatim` and `\endverbatim`, assuming only that the control sequence `\endverbatim` does not need to be set:

```
  \def\beginverbatim{\par\begingroup\setupverbatim\doverbatim}
  {\catcode`\|=0 \catcode`\\=12 % | is temporary escape character
    |obeylines|gdef|doverbatim^^M#1\endverbatim{#1|endgroup}}
```

This construction assumes that `\beginverbatim` appears at the end of a line. Incidentally, it isn't necessary to say that this macro is `\long`, because the `\par`'s inserted by `\obeylines` are really `^^M`'s.

Another approach is to keep one character untouchable. What I'm talking is to define the command so that '`\verbatim` ⟨char⟩⟨text⟩⟨char⟩' will typeset the ⟨text⟩ verbatim.

```
  \def\verb{\begingroup\tt\uncatcodespecials\doverba}
  \def\doverba#1{\def\next##1#1{##1\endgroup}\next}
```

4.1 Show definition of TeX commands

```
Usage:
  texdef   [options] commandname [commandname ...]
  latexdef [options] commandname [commandname ...]
```

Note that command names do not need to start with '\'. For more help, run '`texdef -h`'. For example,

```
$ latexdef sqrt

\sqrt:
macro:->\protect \sqrt

\sqrt :
\long macro:->\@ifnextchar [\@sqrt \sqrtsign

$ latexdef @sqrt

\@sqrt:
macro:[#1]->\root #1\of

$ latexdef sqrtsign

\sqrtsign:
macro:->\radical "270370\relax
```

Chapter 5

The LaTeX features

5.1 History

5.1.1 History of TEX

The story began with Knuth's book *The Art of Computer Programming.* Knuth began the project, originally conceived as a single book with twelve chapters, in 1962. The first three volumes of what was then expected to be a seven-volume set were published in 1968, 1969, and 1973. Work began in earnest on Volume 4 in 1973, but was suspended in 1977 for work on typesetting prompted by the second edition of Volume 2. In 1976, Knuth prepared a second edition of Volume 2, requiring it to be typeset again, but the style of type used in the first edition (called hot type) was no longer available. When Knuth receives the galley proofs [1] of the new book on 30 March 1977, he found them inferior. He then decided to spend some time creating something more suitable. Eight years later, he returned with TEX, which is currently used for all volumes.

On 13 May 1977, he wrote a memo to himself describing the basic features of TEX. He planned to finish it on his sabbatical in 1978, but as it happened, the language was not "frozen" (ready to use) until 1989.

The first version of TEX, called TEX78, was written in the SAIL programming language. For later versions of TEX, Knuth invented the concept of *literate programming*, a way of producing compilable source code and cross-linked documentation typeset in TEX from the same original file.

TEX82 was published in 1982. In which the original hyphenation algorithm was replaced by a new algorithm written by Frank Liang. It also uses fixed-point arithmetic instead of floating-point, to ensure reproducibility of the results across different computer hardware, and included a real, Turing-complete programming language.

In 1989, Donald Knuth released new versions of TeX and Metafont. Despite his desire to keep the program stable, Knuth realised that 128 different characters for the text input were not enough to accommodate foreign languages; the main change in version 3.0 of TEX (also called TEX90) is thus the ability to work with 8-bit inputs, allowing 256 different characters in the text input.

Since version 3, TEX has used an idiosyncratic version numbering system, where updates have been indicated by adding an extra digit at the end of the decimal, so that the version number asymptotically approaches π. This is a reflection of the fact that TEX is now very stable, and only minor updates are anticipated. The current version of TEX is 3.141592653; it was last updated in 2021. He indicated that he firmly believes that having an unchanged system that will produce the same output now and in the future is more important than introducing new features. For this reason, he has stated that the "absolutely final change

[1] In printing and publishing, proofs are the preliminary versions of publications meant for review by authors, editors, and proofreaders, often with extra-wide margins. Galley proofs may be uncut and unbound, or in some cases electronically transmitted. They are created for proofreading and copyediting purposes, but may also be used for promotional and review purposes.

(to be made after my death)” will be to change the version number to π, at which point all remaining bugs will become features.

Documentations LaTeX itself is documented in `source2e.pdf` (`texdoc source2e`) and the standard classes (article, book, report, etc) are documented in `classes.pdf` (`texdoc classes`).

5.2 Headings and sections

For the manual look at

```
$ texdoc classes
```

The macro `\@startsection` has 6 required arguments, optionally followed by a *, an optional argument and a required argument:

`\@startsection`⟨name⟩⟨level⟩⟨indent⟩⟨beforeskip⟩⟨afterskip⟩⟨style⟩ optional *
`[`⟨altheading⟩`]`⟨heading⟩

⟨level⟩ A section number will be printed if and only if ⟨level⟩ $\leq$ the value of the `secnumdepth` counter.

⟨beforeskip⟩ The absolute value of this argument gives the skip to leave above the heading. If it is negative, then the paragraph indent of the text following the heading is suppressed.

⟨afterskip⟩ If positive, this gives the skip to leave below the heading, else it gives the skip to leave to the right of a run-in heading.

For example to see how LaTeX define the `\section` command in document class article, type the following code in terminal

```
$ latex
**\relax
*\documentclass{article}
*\show\section
> \section=\long macro:
->\@startsection {section}{1}{\z@ }{-3.5ex \@plus -1ex \@minus -.2ex}{2.3ex \@p
lus .2ex}{\normalfont \Large \bfseries }.
```

To look at `\z` @ type

```
*\catcode`\@=11
*\show\z@
> \z@=\dimen12.
```

Knuth claimed `\z` @ to be a dimension with value zero to save memory.

The `\appendix` command is a macro that makes some changes in the way things are dome. In the article document class, it is defined as the following.

```
\newcommand\appendix{\par
  \setcounter{section}{0}%
  \setcounter{subsection}{0}%
  \gdef\thesection{\@Alph\c@section}}
```

5.3 The Picture Environment

▷`\vector(x-slope, y-slope){length}`

For example `\vector(1,1){1em}` produce ↗ which is a vector from the reference point to the specified (horizontal) length along the giving direction. Here (x, y) must be signed integer numbers of magnitude ≤ 6 with no common factor. The following command draws lines in a similar way.

▷`\line(x-slope, y-slope){length}`

We use the `\put` command to change the reference point of `\line` or `\vector` or other objects introduced in the following. And `\multiput` to copy objects.

```
▷\put(x-coord, y-coord){object}
 \multiput(x-coord, y-coord)(delta x, delta y){number of copies}{object}
```

Here the coordinates are in units of `\unitlength`. In LaTeX, `\show\unitlength` returns `\dimen` 90, and `\the\dimen` 90 gives the unit length = 1pt.

▷`\circle[*]{diameter}`

If the `*`-form is used, LaTeX draws a solid circle.

▷`\oval(width, height)[portion]`

produces a rectangle with rounded corners. For example

```
\indent First a circle \circle{10} and a rectangle
\oval(6,8)[t] \oval(6,8)[b] \oval(6,8)[r] \oval(6,8)[l] \oval(8,6)
and a raised circle \put(5,5){\circle{10pt}}.
\begin{picture}(20,20)(10,10)
  \put(10,10){\circle*{2pt}}
\end{picture}
```

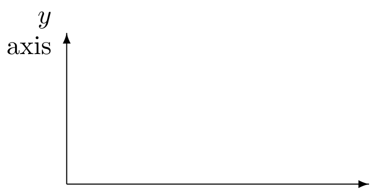
produce

First a circle ◯ and a rectangle and a raised circle.

Here `\begin{picture}(20,20)(10,10)` cut the picture inside of length 20 and height 20, with the reference point at (10,10). A command (something like `\ialign{#\cr ..}`) that produce vertical lists is provided

`\shortstack[position]{.. \\ .. \\ ..}`

As an example see how it is been used



```
\setlength{\unitlength}{1cm}
\begin{picture}(5,2.5)(-0.75,0)
   \put(0,0){\vector(1,0){4}}   % x axis
   \put(0,0){\vector(0,1){2}}   % y
   \put(-0.2,2){\makebox(0,0)[r]{%
     \shortstack[r]{$y$\\ axis}}}
\end{picture}
```

5.4 Make Table of Contents

▷`\addtocontents`{file}{text}

This command adds text (or formatting commands) directly to the file that generates the table of contents (`toc`) or list of figures (`lof`) or tables (`lot`). For example

```
\addtocontents{toc}{\setcounter{tocdepth}{1}} % Set depth to 1
\section{A section}
\subsection{Subsection}
\subsubsection{Subsubsection}
...
\addtocontents{toc}{\setcounter{tocdepth}{3}} % Reset to default (3)
```

if you want the depth of contents change for only one section.

▷`\addcontentsline`{file}{section etc.}{text}

For example

```
\section*{Embedded Curves}
\addcontentsline{toc}{section}{{\large Embedded Curves}}
```

will insert it into the table of contents without affecting the numbering of sections.

▷`\nofiles`

You put thus command in the preamble to prevent LaTeX from writing any auxiliary files. The only output will be the `.log` and `.pdf` files. Nevertheless, LaTeX will not erase any existing auxiliary files, so if you insert the `\nofiles` command after you have run the file and gotten a `.toc` then the table of contents page will continue to show the old information.

5.4.1 Protected macros

```
$latex
**\show\allowbreak
> \allowbreak=macro:
->\protect \allowbreak  .
```

Note that there is white space after the macro.

```
*\expandafter\show\csname allowbreak \endcsname
> \allowbreak =macro:
->\penalty \z@ .

*\show\protect
> \protect=\relax.
<*> \show\protect
```

As pointed out by Joseph Wright you could just type

```
\let\protect\show\LaTeX
```

To show such definitions faster, you could use a command line tool called `texdef` written by Martin Scharrer to display, for example

```
texdef -t latex allowbreak
```

There is also a tool named `latexdef`. Alternatively use quotes to add the space explicitly:

```
latexdef -t 'LaTeX'
```

Some macros are defined to be robust by expanding to `\protect` followed by a macro with almost the same name but ending with a space! If such a macro is written into external files (`.aux`, `.toc`,...) the space is not taken as part of the macro name any longer, but it will be called again.

The commands in moving arguments are normally expanded to their internal structure during the process of saving. Protecting a command, using `\protect\cmd` tells LaTeX to save `\cmd` as `\cmd`, without expanding it at all.

So, what is a “fragile command”? — it’s a command that expands into illegal TeX code during the save process.

What is a “robust command”? — it’s a command that expands into legal TeX code during the save process.

Lamport’s book says in its description of every LaTeX command whether it is “robust” or “fragile”; it also says that every command with an optional argument is fragile.

5.5 LaTeX counters

▷`\counterwithin`{c_1}{c_2}
▷`\counterwithout`{c_1}{c_2}

Links or removes the link between the counters. By 'within' the counter c_1 will be reset to 0 whenever the counter c_2 is stepped, and the command `\the`c_1 will be redefined with c_2. as its prefix. An alternative form

▷`\counterwithin*`{c_1}{c_2}

does not redefine the print-format of the counter c_1.

These two commands are originated in the `chngcntr` package, but have now been integrated into LaTeX itself.

▷`\fnsymbol`{c_1}

Commands accessing and printing counter values including

```
\arabic  \roman  \Roman  \alph  \Alph  and  \fnsymbol
```

TeX provided a command `\romannumeral` ⟨number⟩ to print lowercase roman numerals. This command is more fundamental, it accepts not only counters but plain numbers. For example, '`romannumeral124`' produces '`xxiv`', a list of four tokens each having category 12. With the help of `\uppercase` one obtain capital roman numbers.

After setting

```
\newcounter{fnc}\setcounter{fnc}{0}
```

we obtain each time step the following codes `\stepcounter{fnc}\fnsymbol{fnc}`

$*$	†	‡	§	¶	‖	$**$	††	‡‡
1	2	3	4	5	6	7	8	9

To set up the footnote number using `\fnsymbol` s, define

```
\renewcommand*{\thefootnote}{\fnsymbol{footnote}}
```

The following are default counters in LaTeX

For document structure
- part
- chapter
- section
- subsection
- subsubsection
- paragraph
- subparagraph
- page
- footnote
- mpfootnote

- secnumdepth tocdepth

For floats • equation • figure • table

For enumerate • enumi • enumii • enumiii • enumiv

For a counter, as `section`, there is a macro `\c@section` that denote the counter. Look at the `.log` file you'll probably find that `\c@section=\count178`.

What is the difference between `\number` and `\arabic`?

Question: Understand how `\label` works?

5.6 Problems

1. The next line after the usage of `\eqno` will encounter an extra white space at the beginning, such as

$$a + b = c \tag{$*$}$$

 and you see this line printed incorrect. The problem can be fixed by adding '`%`' right after '`$$`', but I want to known why LaTeX makes thus bug and is there a way to fix this problem for good.

Chapter 6

Useful Packages

6.1 The package `enumitem`

Quick review:

- To remove the vertical space altogether in a list

  ```
  \begin{enumerate}[nosep]
  ```

- To start the label at the margin and the item text at the current parindent:

  ```
  \begin{enumrate}[left=0pt .. \parindent]
  ```

- To set a numeric label with parenthesis, but a cross-reference with out them:

  ```
  \begin{enumerate}[label=(\arabic*),ref=\arabic*]
  ```

- To continue the previous list, after a "pause":

  ```
  \begin{enumerate}[resume*]
  ```

The following definition can be useful.

```
\SetEnumitemKey{medsep}{topsep=3pt,itemsep=0pt,partopsep=0pt}
```

6.1.1 Parameters

`align=left` `align=right` `align=parleft`
These are predefined through `\SetLabelAlign`:

```
\SetLabelAlign{right}{\hss\llap{#1}}
\SetLabelAlign{left}{#1\hfil}
\SetLabelAlign{parleft}{\strut\smash{\parbox[t]\labelwidth{\raggedright##1}}}
```

$$\underbrace{\boxed{\text{labelindent}}\boxed{\text{labelwidth}}\boxed{\text{labelsep} - \text{itemindent}}}_{\text{left margin}}\ \boxed{\text{itemindent}}$$

Example:

```
\begin{enumerate}[labelindent=\parindent,leftmargin=*,
    lable=\Romann*.,widest=III,align=left]
```

Let $A \to A'$ be a faithfully flat ring homomorphism, let M be an A-module and write $M' := M \otimes_A A'$.

I. If M' is flat over A', then M is flat over A.

II. If M' is of finite type over A', then M is of finite type over A.

III. If M' is of finite presentation over A', then M is of finite presentation over A.

IV. If M' is locally free of rank n over A', then M is locally free of rank n over A.

6.1.2 Cloning the basic list

`\newlist`{⟨name⟩}{⟨type⟩}{⟨max-depth⟩}

If ⟨type⟩ is `enumerate`, a set of counters with names ⟨name⟩`i`, ⟨name⟩`ii`, ⟨name⟩`iii`, ⟨name⟩`iv`, etc, (depending on ⟨max-depth⟩) is defined.

For a label to have parents, use for example

```
\newlist{legal}{enumerate}{2}
\setlist[legal]{label*=\arabic*.}
```

6.2 If you don't want to use amsmath package

Plain TEX defined the command `\langle` and `\rangle`. As `amsmath` did, we also want the following definition

```
\def\lvert{\delimiter"426A30C }\def\rvert{\delimiter"526A30C }
\def\lVert{\delimiter"426B30D }\def\rVert{\delimiter"526B30D }
```

Here the first digit 4 stands for opening delimiter while 5 closed delimiter. The next three digits, for instance 26A gives the position when TEX isn't looking for a delimiter, here 2 for family 2 which is default as `cmsy10` (`\cal`, `\textfont2`) and 6A the position in hexadecimal digits. The rightmost three digits, for instance 30C, stands for large variant's position. Compare the following

```
\def\abs#1{\left\lvert#1\right\rvert}
Use $\abs{-a}$ but not $|-a|$.
And use $\abs{a\over b}$ but not $|{a\over b}|$.
```

Use $\lvert -a \rvert$ but not $|-a|$. And use $\left\lvert \frac{a}{b} \right\rvert$ but not $|\frac{a}{b}|$.

primfrac Uploaded `amsmath`, when one use a command like `\over`, it will write a warning message to you.

```
$ latexdef -p amsmath over
\over:
\long macro:->\primfrac {over}

$ latexdef -p amsmath primfrac
\primfrac:
macro:->\protect \primfrac

\primfrac :
\long macro:#1->\PackageWarning {amsmath}{Foreign command \@backslashchar #1;
\MessageBreak\protect\frac\space or \protect\genfrac\space should be used
instead \MessageBreak }
\global\@xp\let\csname#1\@xp\endcsname\csname @@#1\endcsname\csname#1\endcsname
```

To remove the 'Warning message', define

```
\@xp\def\csname primfrac \endcsname#1{%
\global\@xp\let\csname#1\@xp\endcsname\csname @@#1\endcsname\csname#1\endcsname}
```

Equations The `amsmath` package provides an `equation*` environment which is equivalent to `\[`. While Basic LaTeX defined `displaymath` as an aliase of `\[`.

6.3 The hyperref package

6.3.1 Introduction

The `hyperref` adds two cool features to your LaTeX PDF files:

1. The paper size is set according to your specification of the driver commands.
2. All references turn into hyperlinks.

Please put `\usepackage{hyperref}` in the preamble behind other loaded packages, to give it a chance of not being over-written, since its job is to redefine many LaTeX commands. For example, `\section` commands will produce a bookmark and a link.

The following have been defined:

```
\hypersetup{
    bookmarks,
    colorlinks=false, % surround the links by colour frames (false)
        % or colour the text of the links (true)
        linkcolor=red,
        citecolor=green,
        filecolor=magenta,
        urlcolor=cyan}
```

6.3.2 Package options

The options are setted using a single 'key=value' scheme (using the `keyval` package) either in the optional argument to the `\usepackage` command, or using the `\hypersetup` macro. When the package is loaded, a file `hyperref.cfg` is read if it can be found, and this is cnvenient place to set options on a site-wide basis.

In the key descriptions that follow, many options do not need a value, as they default to the value true if used.

6.3.3 Package Commands

`\href`{url}{text}

For a url link, use

The `\href{http://www.ctan.org}{CTAN}` *website.*

for example.

If the destination of the link is a local file, just write

The complete document is `\href{manual.pdf}{here}`.

`\url`{URL}

Similar to `\href`{URL}{`\nolinkurl`{URL}}. Here `\nolinkurl` shall create not a hyperlink.

`\autoref`{label}

Instead of `section~\ref{label}` one use `\autoref{label}` which shall gives a similar results and a bigger target to click.

`\autopageref`{label}

It replaces `\pageref` and adds the name for page in front of the page reference.

Hyperref warning – Token not allowed in a PDF string The following code:

```
\subsection{The classes $\mathcal{L}(\gamma)$}
```

generates the errors:

```
Package hyperref Warning: Token not allowed in a PDF string (PDFDocEncoding):
(hyperref)                removing ‘math shift’ on input line 1938.

Package hyperref Warning: Token not allowed in a PDF string (PDFDocEncoding):
(hyperref)                removing ‘\gamma’ on input line 1938.
...
```

This is because that the bookmarks are not typeset by TeX: they simply are strings of characters, so no math or general formatting instructions are allowed. The easiest method to avoid the warnings is to use `\texorpdfstring` (defined by the package `hyperref`):

```
\subsection{The classes \texorpdfstring{$\mathcal{L}(\gamma)$}{Lg}}
```

where in the second argument you put the best approximation possible; after all the bookmarks are only a guide for consulting the document.

6.4 xcolor

blue		gray		Navy		red	
Brown	brown	Green	green	Orange	orange	Silver	
Cyan	cyan	Magenta	magenta	pink		Yellow	yellow
Gold		Maroon		Purple	purple	YellowGreen	

Note.Navy, Brown, Green, etc. is provided by `svgnames` or `dvipsnames` options, when you want to handle by a different driver for your output.

6.5 The `amsthm` Package

To define a new style:

```
\newtheoremstyle{citing}% name
  {3pt}%      Space above, empty = ‘usual value’
  {3pt}%      Space below
  {\itshape}% Body font
  {}%         Indent amount (empty = no indent, \parindent = para indent)
  {\bfseries}% Thm head font
  {.}%        Punctuation after thm head
  {.5em}%     Space after thm head: " " = normal interword space;
       %        \newline = linebreak
  {\thmnote{#3}}% Thm head spec, leave empty to use default setting
       % plain style sets \thmname{#1}\thmnumber{ #2}\thmnote{ \rm(#3)}
```

When defining theorem environments, use the command `\swapnumbers` to switch the numbering and the thmname.

Reference:

`amsthdoc.pdf` — An introduction to this package. Use ‘`texdoc amsthm`’ to view.

`thmtest.tex` — Contains some examples, including defining new styles, swap numbering. Use ‘`texdoc thmtest`’ to access `thmtest.pdf`, while for `thmtest.tex` itself, I didn’t find it in TeXLive, instead, you can find it in `https://www.ams.org/arc/tex/shortmath/thmtest.tex`

See also `num-thmtest.tex` for my setting of recounter equations.

6.6 A Package to Create a Nomenclature

How often did you try to understand a theorem in a book, but just couldn't figure out what all those strange symbols were all about?

The creation of the nomenclature list is very similar to the creation of index. You need to:

- Put `\usepackage`{nomencl} in the preamble of your document.
- Put `\makenomenclature` in the preamble of your document.
- Issue the `\nomenclature` command for each symbol you want to have included in the nomenclature list.
- Put `\printnomenclature` at the place you want to have your nomenclature list.

Let's see how it works. The command `\makenomenclature` will instruct LaTeX to open the ⟨*main*⟩`.nlo` file and to write the information from your `\nomenclature` commands to this file. The next step is to invoke *MakeIndex*. You feed *MakeIndex* the nomenclature file ⟨*main*⟩`.nlo` along a style file, say `nomencl.ist` and write out to the file ⟨*main*⟩`.nls`. So you should write something like

```
makeindex ⟨main⟩.nlo -s nomencl.ist -o ⟨main⟩.nls
```

Finally, the command `\printnomenclature` will input ⟨*main*⟩`.nls` file.

Now we introduce the usage of the main command `\nomenclature`:

```
\nomenclature [⟨prefix⟩]{⟨symbol⟩}{⟨description⟩}
```

where ⟨*prefix*⟩ is used for fine tuning the sort order. The sortkey will be ⟨*prefix*⟩⟨*symbol*⟩. Examples are

```
\nomenclature{$\pi$}{The ratio of a circle's circumference to its diameter\nomrefpage}
\nomenclature{$\ell(D)$}{The dimension of the vector space $H^0(X,\mathcal O(D))$\nomrefeq}
```

Here `\nomrefpage` and `\nomrefeq` write the page number and the equation number respectively. Also there is a command `\nomrefeqpage` to write them both.

Due to the way `\nomenclature` scans its arguments you don't need to `\protect` any macros, but you also must not have any character in front of the first or between the first and the second argument, especially no line break (even with a `%`). So

```
\nomenclature{$x$}%
  {Description}
```

does *not* work. You can have line breaks in the argument, but also no `%`.

Sorting *MakeIndex* distinguishes three kinds of sort keys:

Strings Everything that starts with a alphabetic letter (A...Z, a...z).

Numbers Everything that starts and only contains digits (0...9).

Symbols Everything else.

Each group is sorted separately, then the groups are sorted in the order symbols, numbers, strings.

Let's consider the following eight nomenclature entries (without the optional argument): `$~Ab$`, `$~aa$`, `$\Ab$`, `$\aa$`, `$Ab$`, `$aa$`, `Ab`, `aa`.

If you use the default settings (i.e. 'a' is added to every sort key, so every key is considered as a string), you will get the sort order `$\aa$`, `$\Ab$`, `$aa$`, `$Ab$`, `$~aa$`, `$~Ab$`, `aa`, `Ab`.

If you specify the option `noprefix`, then you will get `$Ab$`, `$\Ab$`, `$\aa$`, `$aa$`, `$~Ab$`, `$~aa$`, `aa`, `Ab`. The first six entries are considered as symbols and sorted according to the ASCII code.

6.7 The xspace package

Provide a macro `\xspace` which saves the user from having to type `\␣` or `{}` after most occurrences of a macro name in text.

```
\let\nospaceTeX\TeX
\def\TeX{\nospaceTeX\xspace}
\def\bt{Bib\TeX\xspace}
```

`\xspace` peeks ahead for the next token. If the token is in our exception list we break the loop and do nothing; else we try to expand the token once and start over again. If this leads us to an unexpandable token without finding one of the exceptions we insert a space.

6.8 The epsf package

This package seems only support the `tex` driver, does not support the `pdftex` driver. It's purpose is the include PostScript graphics, thus support `.ps`, `.mps`, etc. files. This package is stored in

```
/usr/share/texlive/texmf-dist/tex/generic/epsf/epsf.tex
```

See also a test file `/math/master/LieAlg/figures/test-epsf.tex`.

6.9 Xy-pic package

September 2016: Kristoffer H. Rose died Kristoffer H. Rose died on September 17th 2016 after a long battle with myelofibrosis. Kristoffer was a Debian contributor from the very early days of the project, and the upstream author of several packages, such as the LaTeX package Xy-pic and FlexML. On his return to the project after several years' absence, many of us had the pleasure of meeting Kristoffer during DebConf15 in Heidelberg.

Chapter 7

The Web2C system

Introduction

Web2c contains a set of TEX-related programs, i.e., TEX itself, METAFONT, MetaPost, BIBTEX, etc. The core TEX family components including

bibtex Maintaining bibliographies.

mpost Creating technical diagrams.

patgen Creating hyphenation patterns.

tangle web to Pascal.

tex Typesetting.

For locating files the Web2c programs use the path searching library Kpathsea. To speed up file searches the root of each tree has a file ls-R.

Examples

```
$ kpsewhich article.cls
/usr/share/texlive/texmf-dist/tex/latex/base/article.cls
```

7.1 TeX

Say `tex` in your terminal, you'll saw something like

```
This is TeX, Version 3.14159265 (TeX Live 2020/Debian) (preloaded format=tex)
**
```

The '`**`' is TEX's way of asking you for an input file name. However, if you do not want to specify any file name, you can give TEX a command that is beginning with a backslash '\', e.g. `\relax`. Then TEX returns a `*` waiting for food. In early version of TEX, users are supposed to type '`\input␣story`', (this still work); but most users prefer to put all of their commands into a file instead of typing them online, so TEX make use of the first line to deal with a file. You can just type, say, '`story`' after '`**`', it will be equivalent of saying `\input␣story.tex`.

Also TEX is all geared up for action, ready to read a long manuscript; but what you need may just a couple of command, so you can say `\relax`, telling TEX that it's all right to take things easy.

There's another difference between '`**`' and '`*`': If the first character after `**` is an ampersand ('`&`'), TEX will replace its memory with a precomputed format file before preceding. Thus, for example, you can type '`&plain␣\input␣story`' or even '`&plain␣story`'

in response to `**`, if you are running some version of TEX that might not have the plain format preloaded. NOTE.But my `tex` preload the plain format implicitly. And trying `**&latex` I saw a `Running mktexfmt latex.fmt`, and then it end the program.

What should you do when TEX stops and through you something like this:

```
! Undefined control sequence.
l.2 \vship
           1in
?
```

TEX begins its error messages with '`!`', and it shows what is was reading at the time of the error by displaying two lines of context. The top line of the pair shows what TEX has looked at so far, and where it came from ('`l.2`', i.e., line number 2); the bottom line (in this case '`1in`') shows that TEX has yet to read.

The '`?`' that appears after the context display means that TEX wants advice about what to do next. You can type '`?`' to show the following menu of options:

```
Type <return> to proceed, S to scroll future error messages,
R to run without stopping, Q to run quietly,
I to insert something, E to edit your file,
1 or ... or 9 to ignore the next 1 to 9 tokens of input,
H for help, X to quit.
```

In this case, your best bet is to type

```
I\vskip
```

(and ⟨return⟩), with no space after the '`I`'; this effectively replaces `\vship` by `\vskip`. If you had simply typed ⟨return⟩ instead of inserting anything, TEX would have gone ahead and read '`1in`'. Alternatively, you could have typed '`3`'; that would have deleted '`1in`' from TEX's input.

7.1.1 TeXLive

You can find all search pathes by `kpsexpand '$TEXMF'`, and only the local search path by `kpsexpand '$TEXMFHOME'`, also the main path by `kpsexpand '$TEXMFMAIN'`

7.2 BibTeXing

7.2.1 The BibTEX invocation[1]

BIBTEX creates a printable bibliography (`.bbl`) file from references in a `.aux` file, generally written by TEX or LATEX. Synopsis:

`bibtex [`*option*`]...` *auxfile*`[.aux]`

The basic bibliographic information comes from `.bib` files, and a BIBTEX style (`.bst`) file controls the precise contents of the `.bbl` file. BIBTEX also writes a log file `.blg`.

The name of the `.bib` and `.bst` files are specified in the `.aux` file as well, via the `\bibliography` and `\bibliographystyle` macros.

See also:

`btxdoc.tex`[2]
: Basic LATEXable documentation for general BIBTEX users.

`btxhak.tex`
: LATEXable documentation for style designers.

`btxdoc.bib`
: BIBTEX database file for the two above documents

[1]Reference: `$TEXMFMAIN/doc/web2c/web2c.pdf`
[2]`$TEXMFMAIN/doc/bibtex/base/btxdoc.tex`

7.2.2 Description

For most documents, this one included, you produce the reference list by: running LaTeX on the document (to produce the `aux` file(s)), then running BibTeX (to produce the `bbl` file), then LaTeX twice more (first to find the information in the `bbl` file and then to get the forward references correct).

When used in the `bib` file, the entry-type name is preceded by an character.

article An article from a journal or magazine. Required fields: `author`, `title`, `journal`, `year`. Optional fields: `volume`, `number`, `pages`, `month`, `note`.

book A book with an explicit publisher. Required fields: `author` or `editor`, `title`, `publisher`, `year`. Optional fields: `volume`, `number`, `series`, `address`, `edition`, `month`, `note`.

incollection A part of a book having its own title. Required fields: `author`, `title`, `booktitle`, `publisher`, `year`. Optional fields: `editor`, `volume` or `number`, `series`, `type`, `chapter`, `pages`, `address`, `edition`, `month`, `note`.

mastersthesis A Master's thesis. Required fields: `author`, `title`, `school`, `year`. Optional fields: `type`, `address`, `month`, `note`.

phdthesis A PhD thesis. Required fields: `author`, `title`, `school`, `year`. Optional fields: `type`, `address`, `month`, `note`.

unpublished A document having an author and title, but not formally published. Required fields: `author`, `title`, `note`. Optional fields: `month`, `year`.

misc Use this type when nothing else fits. Required fields: none. Optional fields: `author`, `title`, `howpublished`, `month`, `year`, `note`.

`\nocite{*}` With this single command, you include in the reference list every entry in the database files, without having to explicitly `\cite` or `\nocite` each entry. Giving this command, in essence, `\nocite` s all the enties in the database, in database order, at the very spot in your document where you give the command.

The `author` field accepts author name(s) separated by "and", for example

```
author={Arrow, Kenneth J. and Leonid Hurwicz and Hirofumi Uzawa},
```

Here, you could also write `Kenneth J. Arrow` or `Kenneth Joseph Arrow`, BibTeX will recognise the author's last name as "Arrow". So if you wan to refer to one whose last name is "Maynard Smith", you need to format the name as `Maynard Smith, John` or `John {Maynard Smith}`. Note that it is not exactly the case that BibTeX takes the string following the last space to be the last name. If that string is preceded by a string or strings that start with lowercase letters, those strings as well as the final string are treated as the last name. For example it is the same to write `Pasquale del Pezzo` as `del Pezzo, Pasquale` in the author field.

The `year` field can be written as

```
year=1961,
```

More generally, if the value of a field consists only of digits, the delimiters can be omitted.

7.2.3 Basic BibTeX style files

plain Sorts entries alphabetically, with numeric labels. The other style files listed here are based on `plain`.

abbrv First names, month names, and journal names are abbreviated.

acm Names are printed in small caps.

alpha Alphanumeric labels, e.g., '`Knu66`'.

apalike No labels at all; instead, the year appears in parentheses after the author.

ieeetr Numeric labels, entries in citation order, IEEE abbreviations, article titles in quotes.

siam Numeric labels, alphabetic order. *Math. Reviews* abbreviations, names in small caps.

unsrt Lists entries in citation order, i.e., unsorted.

See also Martin Osborne's website.

7.2.4 History

BibTeX was created by Oren Patashnik and Leslie Lamport in 1985. It is written in WEB/Pascal. Version 0.98f was released in March 1985. With version 0.99c (released February 1988), a stationary state was reached for 22 years. In March 2010, version 0.99d was released to improve URL printing.

7.3 The pdfTeX user manual

7.3.1 Form XObjects

Recall first that if we want to use a font in TeX, we need first prepare the font, then use it. For example

```
\font\tenrm=cmr10
{\tenrm \char"41}
```

will print the character 'A'.

In PDFTeX one can create an object by using `\pdfobj`, after which `\pdflastobj` returns its number. So

```
\pdfobj{<< /Type/ExtGState /LW 2 >>}
```

creates a "graphics state" object setting the line width to 2 units. Unless objects are referenced by others, they will just end up as isolated entities, not doing any real harm but bloating the PDF file.

Similarly, to insert a image one write

```
\pdfximage{somefile.png}\pdfrefximage\pdflastximage
```

It resize needed

```
\pdfximage width=2in {somefile.png}\pdfrefximage\pdflastximage
```

For refer an image many times, write

```
\pdfximage{somefile.png}
\edef\logo{\pdfrefximage\the\pdflastximage\relax}
```

since `\edef` will define a macro with replacement text has been expanded.

7.4 web – The original literate programming

Web is a computer programming system created by Donald E. Knuth as the first implementation of what he called "literate programming": the idea that one could create software as works of literature, by embedding source code inside descriptive text, rather than the reverse.

The system processes 'web' files in two ways: firstly to rearrange them to produce compilable code (using the program `tangle`), and secondly to produce a TeX source (using the program `weave`) that may be typeset for comfortable reading.

The most significant programs written in Web are TEXand Metafont. Modern TeX distributions use another program Web2C to convert Web source to C.

ref: webman

WEB-like languages have been implemented with many pairs of base languages: Cweb provides C and Troff; CWEB provides C and TEX; Spiderweb provides C, C++, Awk, Ada, many others, and TEX; and, of course, the original WEB provides Pascal and TEX.

The original WEB language is documented in the file `webman.tex`.

7.4.1 Tangle: Translate WEB to Pascal

7.4.2 weave – translate WEB to TeX

The `weave` program is used to create a TEX file for viewing a Web program.

Chapter 8

Using vim for TeX

8.1 vimtex

8.2 UltiSnips

8.3 Zathura

Zathura is a highly customizable document viewer with vi-styled keybindings.

command	**behaviour**	**command**	**behaviour**
`a`	page-fit	`s`	fit width
`c`	rotate pages	`d`	view pages side-by-side
`C-r`	black background	`f`	highlight all links