1997

Guide d'introduction français au traitement de textes LaTeX

Frédéric Geraerds
Belgique

Table des matières

1 Introduction **1**

2 Pourquoi LaTeX ? **1**

3 Vocabulaire **2**

4 Principe de fonctionnement **3**

4.1 Rédaction . . . 3
4.2 Compilation . . . 3
4.3 Prévisualisation . . . 3
4.4 Impression . . . 4
4.5 Conversion en PostScript . . . 4

5 Syntaxe **4**

5.1 Caractères de base . . . 4
5.2 Caractères accentués . . . 5
5.3 Caractères spéciaux . . . 5
5.4 Les commentaires . . . 6

6 Document de base **6**

7 Les commandes **7**

7.1 Commande simple . . . 7
7.2 Arguments d'une commande . . . 7
7.3 Argument optionnel dŠune commande . . . 8
7.4 Attention aux espaces . . . 8
7.5 Avantages . . . 8
7.6 Redéfinition d'une commande . . . 9

8 Les environnements **9**

8.1 Fonctionnement général . . . 9
8.2 Définition d'un environnement simple . . . 10
8.3 Argument d'un environnement . . . 10
8.4 Argument optionnel dŠun environnement . . . 10
8.5 Redéfinition d'un environnement . . . 11

9 Les théorèmes **11**

9.1 Définition simple . . . 11
9.2 Numérotation . . . 11
9.3 Changement de la numérotation . . . 12
9.4 Un exemple concret . . . 12
9.5 Argument optionnel . . . 13

10 Fichier de style **13**

11 Mise en forme de base **14**
11.1 Taille des caractères . . . 14
11.2 Polices de caractères . . . 14
11.3 Les paragraphes . . . 15
11.3.1 Création . . . 15
11.3.2 Les espaces entre paragraphes . . . 15
11.3.3 Les retraits de paragraphe . . . 15
11.3.4 Taille du retrait . . . 15
11.4 Soulignement . . . 15

12 Les subdivisions **16**
12.1 La base . . . 16
12.2 Numérotation des parties . . . 16
12.3 Les annexes . . . 16

13 Les espaces **17**
13.1 Espace entre lignes . . . 17
13.2 Espace horizontal . . . 17
13.3 Espace vertical . . . 17
13.4 Espace extensible . . . 17
13.5 Espace insécable . . . 18

14 Les coupures **19**
14.1 Mots . . . 19
14.2 Lignes . . . 19
14.3 Colonnes et pages . . . 20

15 Les récapitulatifs **20**
15.1 Table des matières . . . 21
15.2 Listes des tableaux et des figures . . . 21
15.3 Index . . . 21

16 Les en-têtes et pieds de page **22**
16.1 Le style de page . . . 22
16.2 Numérotation des pages . . . 23

17 Les références croisées **23**
17.1 Définition d'une étiquettte . . . 23
17.2 Numéro associé à une étiquette . . . 23
17.3 Page associée à une étiquette . . . 24

18 La bibliographie **24**
18.1 Création . . . 24
18.2 Référencer un objet de la bibliographie . . . 24
18.3 Programme spécifique . . . 24

19 Sectionner le fichier d'entrée **24**
19.1 Les commandes d'inclusion . . . 25
19.2 Conseil pratique . . . 25

20 Les éléments flottants **25**
20.1 Les deux types . . . 26
20.2 Identification et numérotation . . . 26
20.3 Liste récapitulative . . . 26
20.4 Paramètres . . . 26

21 Les environnements prédéfinis **27**
21.1 Alignement du texte . . . 27
21.2 Les dérivés des listes . . . 27
21.2.1 Liste de points . . . 28
21.2.2 Enumératon . . . 28
21.2.3 Description . . . 29
21.3 Citation et poème . . . 29
21.4 Tabulation . . . 30
21.5 Tableau . . . 31
21.6 Texte littéral . . . 32

22 Les boites **33**
22.1 Le mode LR . . . 33
22.2 Le mode paragaphe . . . 34
22.3 Les boites noires . . . 34
22.4 Les boites décalées . . . 35
22.5 Sauvegarde . . . 35

23 Formules mathématiques **36**
23.1 Les environnements mathématiques . . . 36
23.2 Structures courantes . . . 36
23.2.1 Indices et exposants . . . 36
23.2.2 Fractions . . . 36
23.2.3 Racines . . . 37
23.2.4 Points de suspension . . . 37
23.2.5 Symboles spéciaux . . . 37
23.2.6 Symboles à taille variable . . . 37
23.3 Tableaux mathématiques . . . 37
23.4 Aligner les formules . . . 37
23.5 Placer l'un au dessus de l'autre . . . 38
23.5.1 Les accents mathématiques . . . 38
23.5.2 Sou– et sur– ligner . . . 38
23.5.3 Accolades horizontales . . . 38
23.5.4 *stacking symbol* . . . 38
23.6 Espaces en mathématique . . . 38

23.7 Les délimiteurs 39

24 Les erreurs **39**

25 Mise en page **39**

26 Dessins **39**

27 Inclure une image **39**

28 Divers **40**

28.1 Espaces et points 40

28.2 Points de suspension 40

28.3 Crochets et arguments optionnels 40

28.4 Les longueurs 40

29 Fragile et robuste **41**

30 Sources d'informations **41**

31 Tableaux récapitulatifs **41**

1 Introduction

LaTeX n'est pas un traitement de texte "WYSIWYG" ("What You See Is What You Get") comme Word, WordPerfect, Works ou d'autres. A l'heure du Web, on peut comparer LaTeX au "HTML". Pour composer un texte avec ce dernier langage, on crée, à l'aide d'un éditeur de textes, un fichier contenant à la fois le texte et des commandes de mise en page et de mise en forme de caractères. En LaTeX, le principe est identique mais plus puissant. En effet, contrairement au HTML, il est possible, par exemple, de spécifier la taille des pages (largeur et hauteur du texte) et d'insérer aisément des formules mathématiques.

Un des autres avantages de LaTeX est qu'il existe de nombreuses applications différentes, sur des plates-formes (système d'exploitation et type de machine) différentes qui sont compatibles. Par exemple, on peut travailler un texte sur un Macintosh, et lorsque l'on a le résultat souhaité, envoyer les fichiers à un collègue qui travaille exclusivement sur des stations UNIX pour qu'il en effectue la lecture et la correction et vous retourne ensuite le ou les fichiers corrigés. Dans ces manipulations, il n'y aura eu aucune perte de mise en page du document.

LaTeX est même plus qu'un traitement de texte, c'est un langage de programmation qui permet à l'utilisateur de définir ses propres commandes pour automatiser la mise en forme ou effectuer des tâches répétitives.

Dans ce document, nous supposerons que le lecteur dispose d'une version de LaTeX et que celle-ci est déjà installée. Nous n'envisageons donc pas l'installation d'un logiciel car celle-ci dépend trop fortement du système d'exploitation qu'on utilise. Nous expliquons la création d'un document qui est quasi identique sur toutes les versions du logiciel.

Remarque *Bien que de nombreuses versions de LaTeX existent et sont compatibles pour toutes les commandes standards, il existe une commande dont l'utilisation est libre et transparente dans les compilateurs LaTeX. En fait, elle est généralement destinée au programme d'impression ou au programme de visualisation à l'écran. Elle permet, par exemple, d'inclure une image dans un document.*

Frédéric Geraerds **1997**

2 Pourquoi LaTeX?

Beaucoup de personnes se demandent pourquoi utiliser un traitement de texte comme LaTeX si "rébarbatif" alors que l'on développe des traitements de texte WYSIWYG de plus en plus performant avec des correcteurs orthographiques et grammaticaux, avec création de tableaux par simples clics de souris, avec des dizaines de polices de caractères différentes, avec des éditeurs de formules WYSIWYG également,...

La réponse n'est pas simple. Ou plutôt, elle l'est trop: "C'est une question de goût.". Mais donnons quand même quelques éléments de réponse tout en laissant à l'utilisateur le soin de se faire sa propre opinion.

1. LaTeX est une vrai merveille si on doit rédiger un document avec de nombreuses formules mathématiques car la mise en forme de celles-ci est très soignée. De même, la création et modification de celles-ci est relativement simple.

2. LaTeX permet à l'utilisateur de se concentrer sur le contenu du document sans se soucier de la mise en forme qui sera effectuée automatiquement. Par exemple, on change de taille de caractères relativement à une taille par défaut qui est précisée au début du document. Donc si on modifie la taille par défaut, les tailles relatives sont modifiées également et il n'est pas nécessaire de mettre à nouveau tout le document en page.

3. LaTeX est multi plates-formes ce qui me permet de travailler partout sans difficulté et d'échanger des documents avec la plupart des personnes. Il est donc très facile de rédiger une aide d'un logiciel et de la placer sur un serveur Web pour la mettre à disposition du grand public.

4. Il existe de nombreuses versions Sharware du logiciel disponible sur l'Internet. Il est donc possible de se procurer des versions qui ne coûtent pas trop chères.

Frédéric Geraerds 1997

3 Vocabulaire

Afin de faciliter la compréhension de ce document, il est utile de donner quelques définitions de termes que nous utilisons courramment. Ces définitions sont cependant la plupart du temps redondante avec ce qui est écrit plus avant dans ce texte mais elles sont regroupées ici pour une compréhension plus aisée.

préambule Le préambule d'un document est la partie située entre la première instruction d'un document et le début de l'environnement principale de celui-ci.

fichier d'entrée C'est un fichier texte qui contient le texte qui doit être imprimé ainsi que les instructions qui permettent de mettre celui-ci en page et en forme.

fichier aux Dans ce fichier créé lors de la compilation, LaTeX mémorise les informations concernant les titres, les définitions d'étiquettes,... qu'il répartira éventuellement dans d'autres fichiers pour la création de textes automatiques comme la table des matières.

fichier dvi C'est le fichier créé par le compilateur LaTeX après exécution sur un fichier d'entrée. Il est indépendant du support (*device independent*) c'est-à-dire qu'il peut être utilisé sur toutes les plates-formes.

fichier log C'est le fichier dans lequel LaTeX écrit les commentaires de la compilation d'un fichier d'entrée. Il y indique les erreurs éventuelles ainsi que différentes informations sur les polices utilisées, la taille du fichier dvi généré ou encore les textes spécifiés par l'utilisateur.

fichier toc Dans ce fichier, LaTeX mémorise les informations utiles pour la création de la table des matières.

fichier de style C'est un fichier qui contient des définitions de commandes, de théorèmes ou d'environnements ou des commandes de mise en page de document. On regroupe souvent ces instructions dans un même fichier pour pouvoir les inclure plus facilement dans les différents documents que lŠon rédige.

commande Ce sont les instructions de base de LaTeX qui permettent de changer des paramètres de mise en forme du texte ou l'insertion d'une partie de texte mémoriée. Les commandes sont identifiées par des noms qui commencent tous par un \.

environnement Ils permettent de mémoriser une mise en forme de texte et de l'appliquer ainsi à plusieurs endroits du documents. Il existe des environnements prédéfinis pour créer par exemple des listes, des tableaux,... mais il est également possible de définir ses propres environnements. La modification de la définition d'un environnement entraine la modification de la mise en forme du texte partout où l'on a utilisé cet environnement.

théorème Afin de regrouper les structures similaires telles que les définitions, les propositions, les lemmes,..., LaTeX propose un environnement générique qui permet de dériver ces structures avec ajout de numérotations.

élément flottant C'est une structure particulière de LaTeX qui permet d'inserer un élément sans spécifier exactement son emplacement, celui-ci sera décider lors de la compilation afin de laisser le moins d'espace vide possible.

mode texte On dit que LaTeX est en mode texte s'il compile une portion de texte comme un paragraphe, un tableau, une énumération,... S'il n'est pas en mode texte, il est en mode mathématique. Certaines instructions ne peuvent pas être utilisées dans ce mode.

mode mathématique LaTeX se trouve en mode mathématique s'il compile une formule mathématique. Il utilise une mise en forme totalement différente de celle du mode texte. Par exemple, il ne respecte pas les espaces qui sont insérés, ceux-ci ne servant alors qu'à aérer le fichier d'entrée ou séparer les noms des commandes.

4 Principe de fonctionnement

La création d'un document en LaTeX est composée de plusieurs étapes. Nous ne les détaillerons pas toutes dans ce document car certaines dépendent de la plate-forme sur laquelle on travaille. Nous nous intéresserons essentiellement à la partie concernant la rédaction d'un document.

4.1 Rédaction

Il y a tout d'abord l'écriture d'un *fichier d'entrée* qui contiendra non seulement le texte mais aussi les commandes de mise en page et de mise en forme de ce texte. La syntaxe d'écriture de ce fichier sera envisagée dans la suite. Signalons déjà qu'on ne peut utiliser qu'un nombre réduit de caractères différents dans ce fichier.

4.2 Compilation

Dans cette seconde étape, le programme vérifie la syntaxe du fichier d'entrée et compose un deuxième fichier appelé *fichier dvi (device independent).* Il crée en même temps deux fichiers: *un fichier log* et *un fichier aux.* Dans le premier, il indique toute erreur ou avertissement qu'il rencontre ainsi que des statistiques, des commentaires et de nombreuses autres informations dont nous ne nous préoccuperons pas ici. Dans le second, il mémorise toute instruction utile pour créer des références, la table des matières,...

Il génère également d'autres fichiers qu'il utilise pour mémoriser différentes informations utiles. Par exemples, si on demande l'insertion d'une table des matières, il crée *un fichier toc.* Du fait que LaTeX doit créer celui-ci, il faut compiler deux fois le document pour être certain que la table des matières soit correcte.

4.3 Prévisualisation

A partir du fichier dvi créé à la deuxième étape, il est possible de visualiser à l'écran le résultat de la compilation. On y voit alors exactement ce qui sera éventuellement imprimé par la suite.

De nombreux programmes existent pour effectuer cette opération et dépendent du système sur lequel on travaille. Par exemple sous Windows, on peut utiliser DVIWin[1] qui est un programme spécifique pour prévisualiser et imprimer les fichiers dvi sur nŠimporte quelle imprimante que Windows reconnaît.

[1] DVIWin est un shareware proposé par Hippocrates Sendoukas fonctionnant sous Windows 3.1 et NT. Il comprend des commandes spéciales pour inclure des images telles que wmf et bmp.

Certains programmes LaTeX incorporent un prévisualisateur mais offrent la possibilité dŠen utiliser un autre. De plus, comme le fichier dvi existe généralement sur le disque, il est possible dŠutiliser celui de son choix.

4.4 Impression

Le fichier dvi peut également être imprimé. Là également il y a différents programmes qui existent en fonction de la plate-forme et même parfois de l'impimante. Comme dit ci-dessus, DVIWin permet d'imprimer des fichiers dvi en utilisant les ressources d'impression de Windows.

4.5 Conversion en PostScript

Le fichier dvi nécessite de nombreuses polices de caractères différentes pour être visualisé ou imprimé. Si on veut le lire sur un autre ordinateur, il faut impérativement avoir toutes ces polices, ce qui n'est pas toujours le cas. Elles peuvent être générées automatiquement mais il faut du temps et de l'espace disque. De plus, il y a souvent une multitude de fichiers d'entrée différents vu que l'on sectionne le fichier d'entrée en plusieurs fichiers et qu'il est possible d'inclure des images *eps*.

Afin d'éviter ces inconvénients, il est possible de convertir le fichier dvi en PostScript et obtenir ainsi un seul fichier qui pourra être lu sur la quasi totalité des ordinateurs moyennant l'utilisation d'un programme adéquat. Par exemple, les programmes comme GhostScript permettent d'effectuer la prévisualisation et l'impression de tels fichiers et existent sur Windows et Unix notamment.

L'inconvénient du PostScript est la taille du fichier généré. On arrive assez vite à plusieurs centaines de Ko.

Remarque *Dans la suite, nous détaillons principalement la première étape et nous aborderons juste en quelques mots la phase de compilation avec l'explication de quelques erreurs courantes.*

Remarque *Beaucoup de versions de LaTeX sont en fait composées de plusieurs applications, chacune destinée à effectuer une tâche précise (édition, compilation, prévisualisation, impression, ...). Souvent, l'éditeur permet d'exécuter automatiquement les tâches au moyen de menus ou de commandes au clavier.*

5 Syntaxe

Comme nous venons de le voir, pour rédiger un document à l'aide de LaTeX, il faut créer un fichier texte appelé fichier d'entrée. Celui-ci doit satisfaire un grand nombre de contraintes afin de pouvoir être traduit vers un fichier dvi en vue d'être imprimé ou simplement visualisé. Nous introduisons ici la syntaxe de base d'un tel fichier.

5.1 Caractères de base

En standard, LaTeX ne comprend qu'un nombre limité de caractères. Les seuls caractères autorisés dans le fichier d'entrée sont les suivants: les lettres minuscules et majuscules, les chiffres et les 16 caractères de ponctuation suivants

`. ; , ? : ! ' ‘ ( ) [ ] - / * @`

Les 10 caractères suivants, que nous appellerons *caractères réservés*, sont utilisés pour les commandes LaTeX.

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

Les 5 caractères

`+  =  <  >  |`

sont utilisés uniquement en mode mathématique bien que les deux premiers puissent être utilisés en mode normal. Le caractère `"` n'est jamais utilisé.

Remarque *Il existe un environnement (verbatim, voir 21.6) qui permet d'imprimer un texte tel qu'il est introduit et qui peut contenir n'importe quel caractère. Dans cet environnement, aucune commande ne pourra être utilisée.*

5.2 Caractères accentués

Comme en HTML, pour obtenir les caractères accentués, il faut utiliser une séquence d'autres caractères. Cependant, les nouvelles versions du logiciel sont équipées de convertisseurs automatiques qui comprennent les caractères accentués mais pas certains autres symboles pour lesquels il faut toujours utiliser des séquences de caractères appelées *commandes*. Pour ceux-ci, nous donnons dans la section 31 différents tableaux reprennant ces différentes commandes.

Pour pouvoir utiliser les caractères accentués, il suffit d'inclure une des lignes suivantes dans le préambule du document.

```
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

La première méthode utilise les polices de caractères *cm* qui ne sont pas spécialement conçues pour les caractères accentués alors que la deuxième méthode utilise les polices *dc* mieux conçues pour ceux-ci. L'utilisation de cette dernière nécessite la présence des polices adéquates alors que la première utilise les polices standard de LaTeX. Généralement, les différentes versions proposent logiciel pour créer les polices nécessaires à la visualisation ou l'impression d'un fichier dvi mais il faut du temps et de l'espace disque pour créer et mémoriser celles-ci.

Remarque *Afin dŠutiliser ces lignes, il est nécessaire de disposer des fichiers de style adéquats.*

5.3 Caractères spéciaux

Nous avons donné en 5.1 les caractères autorisés dans le texte fourni au compilateur. De nombreux autres caractères sont cependant disponibles mais s'obtiennent par des commandes LaTeX. Il est très important de respecter l'écriture (majuscules ou minuscules) des commandes car LaTeX en tient compte.

Par exemple, les caractères réservés #, $, \, &, %, { et } s'obtiennent grâce aux commandes respectives `\#`, `\$`, `$\backslash$`, `\&`, `\%`, `\{` et `\}`

De nombreux autres caractères spéciaux sont disponibles et sont regroupés dans des tableaux fournis à la section 31. Par exemple, l'instruction `$\heartsuit$` place le symbole $\heartsuit$ dans le texte. Remarquons que le mode mathématique (voir 23.1) est nécessaire pour la plupart de ces symboles spéciaux.

5.4 Les commentaires

Comme LaTeX est un traitement de texte qui nécessite une compilation, il est aisément possible d'insérer des commentaires dans le fichier d'entrée. Un commentaire apparaît dans ce fichier mais n'est pas répercutés lors de l'impression ou de la prévisualistion du fichier dvi généré par compilation de celui-ci.

La syntaxe utilisée par LaTeX pour les commentaires est la suivante. Sur une ligne du fichier d'entrée, tout texte qui suit un `%` non directement précédé d'un `\` est ignoré lors de la compilation. Il y a deux exceptions à cette règle, la première dans l'environnement `verbatim` et la seconde dans la commande `\verb` (voir 21.6 pour plus de détail sur cet environnement et cette commande).

Nous conseillons l'utilisation de cette possibilité essentiellement lors de la définition des commandes, environnements et autres que nous détaillons ailleurs dans ce document. Par exemple, insérez un commentaire pour décrire ce que doit faire la commande que vous avez définie. Si nécessaire, placez également un exemple si son utilisation en est particulière. Ainsi, si une tierce personne lit votre fichier (de style ou d'entrée), elle comprendra l'utilité des définitions que vous avez utilisées.

6 Document de base

Nous allons ensemble créer un premier document. Celui-ci sera le plus simple possible et affichera juste les mots "Voici mon premier document en LaTeX.". Remarquons que le dernier mot de cette phrase est tout à fait spécifique vu la disposition des caractères.

La structure d'un fichier d'entrée se présente comme suit:

```
\documentclass[option]{style}[date]
préambule
\begin{document}
texte
\end{document}
```

La première ligne définit le style du document. Les plus courants sont *article*, *report*, *book* et *letter*. Le style décrit la manière générale d'organiser le document. Par exemple, un livre (dont le style est *book*) peut être divisé en parties puis en chapitres. Par contre un rapport ne peut contenir qu'une partie et plusieurs chapitres et un article ne contiendra aucun chapitre mais uniquement des sections.

On trouve également sur cette première ligne une zone *facultative* placée entre crochets où il est possible de préciser des options de présentation du document. Parmi celles-ci, nous ne citerons que la taille des caractères qui peut être précisée par `11pt` ou `12pt`. Le défaut le LaTeX est 10 points (et dans ce cas, il ne faut pas mettre les crochets). Par exemple, si on spécifie une taille de 12 points (`12pt` comme option du style), le texte d'un paragraphe sera tapé en police 12 sauf mention explicite du contraire dans le document comme nous le décrivons dans la section 11.1. La taille des titres est également proportionnelle à cette valeur. La deuxième zone facultative peut contenir une date (au format AAAA/MM/JJ) qui permet de spécifier unedate minimal pour le style. On ne pourra utiliser un style dont la date est antérieur à celle spécifiée.

Le *préambule* contient la définition de la taille des pages (voir 25) et les définitions des commandes (voir 7), des environnements (voir 8) et des théorèmes (voir 9). Il peut également contenir des commandes d'inclusion de *fichiers de style* (voir 10).

Les troisième et cinquième lignes sont les délimiteurs principaux du document. En fait, c'est l'*environnement* (voir 8) de base du document. Elles ne sont présentes qu'une et une seule fois par document. (Nous faisons déjà remarquer qu'un document peut se composer de plusieurs fichiers d'entrée comme nous le précisons dans la section 19.) Entre ces deux lignes, on place le texte (du document) qui sera imprimé avec les instructions de mise en page et de mise en forme.

Le fichier d'entrée de notre premier document sera alors constitué des lignes suivantes:

```
\documentclass{article}
\begin{document}
Voici mon premier document en \LaTeX.
\end{document}
```

La première ligne indique que le style du document est *article*. Comme il n'y a pas d'option, la taille par défaut est appliquée, c'est-à-dire 10 points. Comme le préambule est vide, LaTeX utilisera les options de pagination par défaut qui sont précisées par le style du document.

Cet exemple nous permet encore d'introduire les *commandes*, une notion qui sera envisagée en détail dans la section 7 de ce document. En tapant `\LaTeX` dans le fichier d'entrée, on demande à LaTeX d'introduire un texte qu'il a mémorisé (avec sa mise en forme); dans notre cas, il s'agit du texte LaTeX. Remarquons que cette commande est, par défaut, connue de LaTeX. On dira que c'est une *commande interne* au programme.

Frédéric Geraerds 1997

7 Les commandes

La commande constitue l'un des principaux composants de LaTeX. Chaque commande est identifiée par un nom unique qui doit commencer par le caractère \ et se poursuivre par une séquence de lettres. Remarquons que les minuscules et les majuscules sont considérées comme différentes par LaTeX.

7.1 Commande simple

L'utilisateur peut créer des commandes supplémentaires qui lui permettront, par exemple, d'abréger l'écriture de son texte. Supposons que vous rédigez un rapport sur la comparaison de trois produits pharmaceutiques dont l'un d'eux est pourvu d'un nom complexe, mettons "Truc machine 500mg effervescent". Vous devez bien entendu indiquer le nom complet chaque fois que vous faites référence à ce produit et cela est très fatiguant. LaTeX vous permet de simplifier fortement la rédaction du rapport en définissant une commande courte qui, chaque fois qu'elle sera insérée dans le document, sera remplacée à la compilation par le texte associé.

La définition d'une commande de nom *nom* et de texte associé *texte* se fait au moyen de la ligne
`\newcommand{`*nom*`}{`*texte*`}`
qui peut se trouver dans le préambule ou dans le texte du document. Nous conseillons cependant de regrouper les définitions de commande dans le préambule afin de pouvoir facilement les retrouver pour d'éventuelles modifications. Nous suggérons d'utiliser les possibilités de commentaires de LaTeX pour bien organiser le préambule.

Dans le cas de notre produit pharmaceutique, nous avons la ligne
`\newcommand{\truc}{Truc machine 500mg effervescent}`
si on décide de nommer la commande par `\truc`.

7.2 Arguments d'une commande

L'instruction `\newcommand` admet également un paramètre optionnel qui permet de spécifier le nombre d'arguments de la commande définie. On a alors

`\newcommand{`*nom*`}[`*nargs*`]{`*texte*`}`
où *nargs* est ce paramètre optionnel qui doit être un nombre entier entre 1 et 9. Pour faire référence à ces arguments qui seront spécifiés entre accolades après le nom de la commande lors de l'appel, on utilise les caractères `#n` où `n` est le numéro de l'argument.

Dans notre exemple, on pourrait faire du poids (le nombre 500), un argument de la commande. On aurait alors
`\newcommand{\truca}[1]{Truc machine #1mg effervescent}`
et le nom du produit serait obtenu par l'instruction `\truca{500}`. On peut encore définir la commande `\truc` qui évitera l'introduction du paramètre supplémentaire au moyen de l'instruction suivante
`\newcommand{\truc}{\truca{500}}`
Remarquons que dans les deux cas, la commande `\truc` produit le même effet dans le document mais dans le deuxième exemple, on est plus général dans le sens où on peut facilement placer le même nom de produit mais qui n'aurait qu'un poids de 250.

7.3 Argument optionnel dŠune commande

Certaines commandes de LaTeX peuvent avoir un argument optionnel qui est alors spécifié entre crochets. Par exemple, la commande `\documentclass` introduite précédemment peut avoir différentes options de style facultatives.

Il est également possible de créer ses propres commandes qui auront au maximum un argument optionnel. Pour cela, on utilise un deuxième argument optionnel à la commande `\newcommand`.
`\newcommand{`*nom*`}[`*nargs*`][`*défaut*`]{`*texte*`}`
où *défaut* est la valeur qui sera attribuée au premier argument (spécifié par `#1` dans le texte) sŠil nŠest pas précisé.

Supposons par exemple que lŠon désire créer une commande permettant dŠabréger lŠécriture de $x_1, \ldots, x_n$ où x devra être spécifié mais n pourra être omis. On peut alors écrire la définition
`\newcommand{\seq}[2][n]{$#2_1,\ldots,#2_#1$}`

Ainsi définie, la commande `\seq{x}` donnera la formule précisée ci-dessus et `\seq[k]{y}` produira $y_1, \ldots, y_k$.

7.4 Attention aux espaces

Il faut faire attention avec l'utilisation des commandes car LaTeX ignore un espace qui suit immédiatement la commande lors de sont insertion dans le texte vu qu'il considère que celui-ci indique la fin de son nom. Par exemple, le texte
`\LaTeX est génial.`
produira
LaTeXest génial.
On remarque qu'il manque un espace après LaTeX. Comme un espace ou dix, cela revient au même, il est inutile de mettre plusieurs espaces car ça produira un texte identique. Pour forcer un espace après une commande, il suffit d'insérer la commande `\␣` (un antislash suivit directement d'un espace) juste dernière le nom de la commande. Pour produire le texte désiré, on introduit l'instruction
`\LaTeX\ est génial.`

7.5 Avantages

En plus de l'avantage de réduire la frappe, il y a la facilité de modification et la sûreté de ne jamais faire de faute dans le texte. En effet, supposons que le produit que vous analysez soit un produit en développement dont le nom n'était pas encore bien fixé et qui vient de changer. Si vous avez rédigé le document en tapant chaque fois le nom complet, il vous faudra le remplacer partout dans le rapport alors que si vous avez défini une commande, vous le modifiez une seule fois.

De plus, comme il n'est tapé qu'une seule fois, il est aisé de vérifier son orthographe à cet endroit et vous avez l'assurance qu'il sera identique partout.

7.6 Redéfinition d'une commande

Pour en terminer avec les commandes, il nous faut encore envisager une instruction qui peut parfois porter à confusion. Celle-ci permet de redéfinir une commande c'est-à-dire de définir une commande qui possède un nom déjà existant. Cette instruction procède exactement comme `\newcommand` et sa syntaxe est

`\renewcommand{`*nom*`}[`*nargs*`]{`*texte*`}`

Si une telle ligne est placée dans le document, tout appel à cette commande dans le texte qui suit sera un appel à cette nouvelle définition, la précédante n'existant plus.

Nous déconseillons l'utilisation de cette instruction sauf éventuellement dans le préambule. Dans ce dernier cas en effet, la commande produira le même résultat tout au long de ce document. **Frédéric Geraerds 1997**

8 Les environnements

Les environnemetns sont les fondements de la mise en page de LaTeX. Ils permettent de définir une présentation du texte qui se situe à l'intérieur de façon identique tout au long du document. C'est un peu l'équivalent des *styles* de Word avec plus de possiblités comme nous allons le détailler dans la suite.

LaTeX possède de nombreux environnements prédéfinis qui permettent de créer des listes numérotées ou non, des citations, des textes littéraux, des tableaux,... Avant d'en arriver à la description de ceux-ci dans la section 21, nous expliquons leur méthode de fonctionnement générale.

8.1 Fonctionnement général

Un environnement est identifié par un nom unique pour lequel LaTeX distingue les majuscules et les minuscules. Contrairement aux commandes qui doivent avoir un nom qui commence par le caractère réservé \, les noms des environnements peuvent être constitués de n'importe quelle séquence de caractères standards.

Un environnement permet de spécifier une présentation particulière du texte. Pour cela, il faut placer une instruction en indiquant le début et une autre en indiquant la fin. Pour un environnement intitulé *nom*, on procède comme suit

`\begin{`*nom*`}`
texte
`\end{`*nom*`}`

De la sorte, LaTeX présentera le *texte* avec la mise en page spécifiée dans la définition de l'environnement.

Remarque *Nous avons déjà vu un environnemnt de LaTeX dans la section 6. Il s'appelle et c'est l'environnement principal de tout document qui doit se trouver une et une seule fois dans tout texte rédigé pour LaTeX.*

Remarque *Nous conseillons d'aérer assez le fichier d'entrée afin d'en faciliter la modification. Ainsi, insérez un retour chariot dans le fichier d'entrée avant et après les instructions de début et de fin* *d'environnement comme ci-dessus.*

8.2 Définition d'un environnement simple

LaTeX propose déjà un grand nombre d'environnements prédéfinis mais il permet également à l'utilisateur d'en définir de nouveaux propres à ses documents. Ce dernier peut donc mémoriser une présentation particulière du texte et l'utiliser lorsqu'il en a besoin. S'il modifie la définition de son environnement, pour mettre, par exemple, le texte en italique, tout texte formatté avec cet environnement sera alors modifié dans l'entièreté du document lors de la compilation suivante.

Voyons donc comment on peut définir son propre environnement. L'instruction
`\newenvironnement{`*name*`}{`*beg_texte*`}{`*end_texte*`}`
définit un nouvel environnement. Celui-ci débutera par *beg_texte* et se terminera par *end_texte*. En d'autres termes, l'instruction `\begin{`*name*`}` sera remplacée par *beg_texte* et `\end{`*name*`}` par *end_texte*. *name* est la séquence de caractères qui permet d'identifier l'environnement et *begin_texte* et *end_texte* sont des instructions de LaTeX.

Par exemple, lorsque l'on rédige un document mathématique, on est souvent amené à démontrer des théorèmes. Nous indiquerons dans la section 9 comment insérer l'énoncé d'un théorème mais voyons ici comment insérer la démonstration. Nous désirons que toute démonstration débute par le mot *Démonstration* en gras et qu'elle se termine par un petit losange vide ($\diamondsuit$). Nous insérons alors l'instruction suivante dans le préambule du document

```
\newenvironment{Dem}{\bf Démonstration}{$\diamondsuit$}
```

Cette définition comporte des inconvénients et nous expliquons à la section 9.4 comment réaliser cela au moyen de la structure de théorème. Les inconvénients sont les suivants. Il est impératif d'insérer manuellement un passage à la ligne à l'aide de l'instruction `\\` avant et après l'utilisation de cet environnement en fonction du texte qui l'entoure. On ne peut insérer celles-ci dans la définition sinon elles provoquent éventuellement des avertissements lors de la compilation et l'ajout d'espaces indésirés. De plus, le fait d'insérer un retour chariot ne produit pas un espace supplémentaire pour aérer un peu le document alors que c'est souhaitable. Tout ceci fait que nous spécifierons une autre définition pour cet environnement dans la section 9.4.

8.3 Argument d'un environnement

Il est possible de fournir un ou plusieurs argument à un environnement. Pour cela, on définit l'environnement au moyen de l'instruction
`\newenvironnement{`*name*`}[`*args*`]{`*beg_texte*`}{`*end_texte*`}`
où `n` est un entier entre 1 et 9. Pour accéder aux arguments dans *beg_texte* et *end_texte*, on utilise la séquence de caractères `#n` où `n` est la position de l'argument qui seront précisés entre accolades dans le fichier d'entrée. En fait, le principe est identique à celui expliqué en détail pour les commandes dans la section 7

8.4 Argument optionnel dŠun environnement

Nous avons spécifié pour une commande comment permettre un argument optionnel. La même syntaxe est utilisable pour les environnements.
`\newenvironnement{`*name*`}[`*args*`][`*défaut*`]{`*beg_texte*`}{`*end_texte*`}`

8.5 Redéfinition d'un environnement

Après avoir défini un environnement, il est possible d'en changer la définition au cours du document au moyen de la commande
`\renewenvironnement{`*name*`}[`*args*`]{`*beg_texte*`}{`*end_texte*`}`
Celle-ci fonctionne exactement comme celle de définition.

Nous déconseillons encore une fois cette manière de procéder car elle mène souvent à des ambiguïtés sur la définition qui est utilisée. **Frédéric Geraerds** **1997**

9 Les théorèmes

Les textes mathématiques contiennent souvent des théorèmes et des structures similaires telles que axiomes, définitions, lemmes, propositions,... Avoir un environnement particulier pour chaque structure ne serait pas pratique car elles sont toutes similaires et en très grand nombre.

9.1 Définition simple

LaTeX propose la commande `\newtheorem` pour de telles structures. Cette commande possède deux arguments. Le premier est le nom de l'environnement et le second est le titre qui servira à l'identifier. Comme pour les environnements, le nom d'un théorème est une séquence de caractères standards et LaTeX fait la distinction entre majuscules et minuscules.

Considérons l'exemple d'un théorème. Nous choisissons de le nommer `The` et son titre est évidemment `Théorème`. La définition est alors
`\newtheorem{The}{Théorème}`
Elle peut apparaître à n'importe quelle place dans le texte mais nous recommandons de regrouper de telles définitions dans le préambule ou dans un fichier de style (voir 10).

L'utilisation d'une structure définie par la commande `\theorem` se fait exactement de la même manière que pour les environnements. Ainsi, pour créer un théorème dans un texte suivant notre exemple, on utilise les lignes suivantes.
`\begin{The}`
Le texte du théorème.
`\end{The}`
Le résultat est

Théorème 1 *Le texte du théorème.*

9.2 Numérotation

On peut donc définir les différentes structures nécessaires pour les documents que l'on va créer. Celles-ci sont numérotées, c'est-à-dire que le nom est suivi d'un numéro (comme le montre notre exemple ci-dessus) qui débute à 1 au début du document et est incrémenté d'une unité à chaque fois que l'on utilise la structure correspondante. On peut donc avoir une définition 1 et un théorème 1. Il n'est alors pas toujours facile de localiser un élément particulier à partir de son numéro, surtout s'il n'y en a pas beaucoup dans le document. En effet, supposons que l'on cherche la définition 4 et que l'on trouve sur la page 2 la troisième définition et sur la page 34, la cinquième. A ce momment, rien ne permet de savoir où se trouve la définition recherchée et l'on doit alors parcourir les 30 pages pour la trouver (si on n'a pas de chance).

LaTeX propose deux solutions à ce problème. La première permet de lier la numérotation aux chapitres, sections,... du document et la seconde lie les compteurs les uns aux autres.

Pour lier la numérotation aux chapitres par exemple, on utilise un troisième argument optionnel à la commande `\newtheorem`. Celui-ci est placé entre crochets de la manière suivante.

```
\newtheorem{The}{Théorème}[chapter]
```

Pour lier les compteurs les uns aux autres, on utilise également un argument optionnel, mais celui-ci se place entre les deux autres arguments. Par exemple, pour lier le compteur des propositions à celui des théorèmes, on utilise les définitions suivantes.

```
\newtheorem{The}{Théorème}
\newtheorem{Pro}[The]{Proposition}
```

De la sorte, la proposition qui suit le théorème 5 sera numérotée par 6 et le théorème ou la proposition suivant portera le numéro 7.

Bien entendu, il est possible de combiner les deux méthodes de numérotation pour soumettre les compteurs aux numéros de chapitre et les faire dépendre les uns par rapport aux autres. Par exemple, on définira les propositions et les théorèmes par les commandes qui suivent.

```
\newtheorem{The}{Théorème}[chapter]
\newtheorem{Pro}[The]{Proposition}
```

Avec cette dernière méthode, il est beaucoup plus facile de rechercher un théorème ou une proposition à partir de son numéro. En effet, un numéro est composé du numéro du chapitre et ensuite d'un numéro séquentiel commun aux deux structures.

9.3 Changement de la numérotation

Dans cette section, voyons comment il est possible de changer manuellement le style d'écriture du compteur d'un théorème. Supposons pour cela que l'on désire créer une structure pour les exemples numérotée en chiffres romains. Nous avons vu que la définition s'obtenait au moyen de l'instruction

```
\newtheorem{Exp}{Exemple}
```

si on désire l'intituler "Exp" (attention, la majuscule est importante). Lorsque LaTeX compile cette ligne, il crée un compteur de nom `Exp` qui mémorise le numéro de la prochaine occurence. Il crée également la commande `\theExp` qui mémorise la mise en forme du numéro. Par exemple, elle mémorise le fait que le compteur est soumis à la numérotation des chapitres.

Pour changer la mise en forme de la numérotation, il suffit de redéfinir la commande `\theExp` au moyen de l'instruction

```
\renewcommand{\theExp}{\roman{Exp}}
```

La numérotation est alors en chiffres romains minuscules.

9.4 Un exemple concret

Nous avons envisagé dans la section 8.2 un environnement qui permettait de faire une démonstration. Nous avions expliqué qu'il comportait des inconvénients et nous allons donc donner ici une structure beaucoup plus flexible ne conduisant plus à des erreurs. La définition de l'environnement de démonstration s'obtient au moyen des lignes suivantes que nous conseillons de placer dans un fichier de style.

```
\newtheorem{Demonstration}{Démonstration}
  % Définition d'une structure de théorème intitulé Demonstration
\renewcommand{\theDemonstration}{}
  % Suppression de la numérotation de la structure Demonstration
\newenvironment{Dem}{\begin{Demonstration}}{$\diamondsuit$\end{Demonstration}}
```

```
% Création de l'environnement Dem basé sur la structure Demonstration avec
% insertion du symbole particulier en fin de structure
% REM: Suivant notre habitude d'écriture du fichier d'entrée, nous passons à la
%  ligne avant d'insérer l'instruction de fin d'environnemnt. Dans ce cas, il
%  ne faut pas oublier de mettre un % sur la dernière ligne sans espace entre
%  le mot et celui-ci pour que le symbole colle à la fin et ne soit placé à la
%  ligne. Par exemple
%  \begin{Dem}
%  ..., cqfd%
%  \end{Dem}
```

9.5 Argument optionnel

Les structures définies à partir de l'instruction `\newtheorem` admettent lors de l'utilisation, un argument optionnel qui suit immédiatement l'accolade fermante après le nom. Précisons son utilité sur l'exemple d'un théorème. Le texte produit par les lignes

```
\begin{The}[Pythogore]
La somme des carrés des longueurs côtés adjacents à l'angle droit d'un
triangle rectangle est égale au carré de la longueur de l'hypothénuse
de ce triangle.
\end{The}
```

est

Théorème 2 (Pythogore) *La somme des carrés des longueurs des côtés adjacents à l'angle droit d'un triangle rectangle est égale au carré de la longueur de l'hypothénuse de ce triangle.*

Frédéric Geraerds 1997

10 Fichier de style

A différentes reprises dans ce texte, nous définissons des instructions qui peuvent être utiles dans plusieurs documents. Nous conseillons souvent de placer les définitions dans le préambule afin d'augmenter la lisibilité du document. L'inconvénient de cette méthode est que si on décide de modifer l'une des définitions, il faut effectuer cette modification dans chaque document qui la contient. La création d'un *fichier de style* permet d'éviter cela. Une modification dans le fichier de style est automatiquement répercutée lors de la compilation suivante dans tous les documents qui y font référence.

Un *fichier de style* est un fichier texte auquel on attribue souvent l'extension `.sty`. Il ne contient généralement pas les lignes indiquant le début et la fin du document. Par contre, il contient des commandes ou des définitions de commandes et d'environnements que nous introduisons par ailleurs afin que celles-ci soient utilisables dans le fichier d'entrée qui fait référence à ce fichier de style.

On créera par exemple un fichier nommé `A4.sty` qui contiendra la définition des dimensions des feuilles de format A4. Ce fichier ne sera donc constitué que de quelques lignes. Pour utiliser ces instructions dans un fichier d'entrée, on insère dans le préambule de ce document la ligne
`\usepackage{A4}`

Il est préférable de créer un fichier de style par type d'information que l'on désire y placer. Par exemple, on crée un fichier pour chaque format de page (`A4.sty`, `A5.sty`, ...). On en crée également un pour la définition des commandes, environnements et théorèmes génériques (utilisées dans la majorité des documents que l'on crée). Les instructions spécifiques à un document seront insérées dans le préambule de ce document ou dans un fichier de style spécifique si on doit réaliser plusieurs documents sur le même thème.

Frédéric Geraerds 1997

11 Mise en forme de base

Nous avons expliqué dans la section 6 la création d'un document et nous savons donc maintenant comment créer un document simple. Nous allons envisager la création de paragraphes, le changement de type de caratères, les espacements,...

11.1 Taille des caractères

La taille des caractères est en premier lieu définie par les options de style spécifiées au début d'un document comme indiqué en 6. En plus de cela, il existe 10 tailles que l'on peut spécifier par les commandes du tableau 1. L'effet d'une telle commande est de changer la taille des caractères jusqu'à la fin de l'environnement courant (éventuellement jusqu'à la fin du document). Pour ne changer que la taille de quelques mots, on les place entre accolades avec la commande de taille au début. Par exemple, un texte en petit est obtenu par `{\small un texte en petit}`

Tableau 1: Taille des caractères

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

Remarque *Dans le document, le changement de taille de caractères est relatif à la taille par défaut spécifiée dans les options de style du document. De ce fait, un changement de cette taille générale n'affecte en rien les autres changements. Si par exemple, dans un paragraphe, on désire noter en petit une remarque entre parenthèses, on place alors ce texte entre des accolades avec l'instruction* `\small` *au début. Si la taille par défaut est de 12 points, la taille de ce texte sera de 11 points alors que si la taille par défaut est 11, elle sera de 10.*

11.2 Polices de caractères

La police par défaut de LaTeX est la police roman. D'autres polices sont également disponibles pour mettre en évidence certaines parties de texte. Le tableau ci-dessous reprend toutes les polices utilisables en standard dans l'ancien LaTeX. A gauche, on trouve le résultat de l'instruction placée à droite.

This is a bold type style.	{\bf This is a bold type style.}
This is an italic type style.	{\it This is an italic type style.}
This is an emphasized type style.	{\em This is an emphasized type style.}
This is a sans serif type style.	{\sf This is a sans serif type style.}
This is a slanted type style.	{\sl This is a slanted type style.}
THIS IS A SMALL CAPS TYPE STYLE.	{\sc This is a Small Caps type style.}
`This is a typewriter type style.`	{\tt This is a typewriter type style.}
This is a roman type style.	{\rm This is a roman type style.}

Dans le cas où la police italique (ou emphasized) ne porte que sur une partie d'un mot, il faut ajouter `\/` avant de repasser en mode normal. Ainsi *bon*jour et *bon*jour sont différents et obtenus en tapant respectivement `{\em bon}jour` et `{\em bon\/}jour}`. Remarqué bien la taille entre le n et le j.

Deux instructions du même changement de police ont un effet d'annulation. Ainsi l'instruction `{\em Un {\em texte accentué} dans un texte accentué}` produit le texte *Un* texte accentué *dans un texte accentué.*

Remarque *Pour obtenir un style d'écriture large et gras, il faut utiliser la combinaison de commandes* `\large\bf` *et non l'inverse.*

11.3 Les paragraphes

Comme le retour chariot dans le fichier d'entrée n'indique pas un paragraphe mais un espace, il est nécessaire d'utiliser une autre méthode pour créer des paragraphes.

11.3.1 Création

Pour créer un paragraphe, il faut laisser une ligne complètement vide dans le fichier d'entrée. Elle ne peut pas contenir de caractères, même un commentaire.

11.3.2 Les espaces entre paragraphes

Par défaut, LaTeX n'insère pas d'espace supplémentaire entre deux paragraphes. Vu que l'espace laissé est minimum, le texte est très condensé et donc difficile à lire. Voyons comment il est possible de changer ce paramètre afin de mieux séparer deux paragraphes.

La dimension (voir 28.4) `\parskip` correspond à l'espace additionnel entre deux paragraphes. Par défaut, elle vaut 0 et elle peut être modifiée uniquement dans le préambule.

Par exemple, placer l'instruction `\parskip=5pt` dans le préambule d'un document provoque un espacement vertical de 5 points supplémentaire entre chaque paragraphe.

11.3.3 Les retraits de paragraphe

A la base, LaTeX est un logiciel anglais et sa mise en page peut ne pas convenir à un texte français. Un des défauts de la mise en page anglaise consiste à ne pas insérer de retrait de paragraphe pour le premier paragraphe qui suit un titre.

Les commandes suivantes permettent de modifier cette lacune. Elles doivent être placées dans un fichier de style et inclus dans le préambule au moyen de la commande `\usepackage`.

```
\let\@afterindentfalse\@afterindenttrue
\@afterindenttrue
```

Remarque *Créez donc un fichier nommé* `indent.sty` *et placez-y ces deux lignes. Lorsque vous désirez indenter la première ligne du paragraphe dans un document, incluez ce style dans votre préambule.*

11.3.4 Taille du retrait

Il est possible de modifier la taille du retrait de paragraphe en attribuant une nouvelle valeur à la dimension `\parindent`.

11.4 Soulignement

Il est possible de souligner un texte au moyen de la commande `\underline`. Celle-ci est cependant fragile et ne peut donc pas être utilisée dans n'importe quelle condition.

Remarque *En fait, nous n'avons presque jamais utilisé cette commande en LaTeX, nous utilisons de préférence les autres possiblités de mise en forme.*

Frédéric Geraerds 1997

12 Les subdivisions

Lorsque l'on rédige un document, il faut le subdiviser en parties logiques numérotées et correctement espacées. Nous détaillons dans cette section les possibilités offertes par LaTeX.

12.1 La base

La méthode de subdivision d'un document dépend du style du document. Il exite sept possibilités de subdivisions mais elles ne sont pas toutes utilisables avec certains styles. Les sept commandes permettant de créer les subdivisions sont

`\part`	`\subsection`	`\paragraph`
`\chapter`	`\subsubsection`	`\subparagraph`
`\section`		

Toutes ces commandes doivent avoir un et un seul argument qui est le titre de la séparation que l'on désire créer. Par exemple, pour créer cette section, nous avons introduit la commande
`\subsection{La base}`.

Dans un livre et un rapport, toutes les subdivisions sont utilisables et nous laissons le soin à l'utilisateur de faire quelques essais pour observer les effets qu'elles produisent. Remarquons que l'effet de ces commandes n'est pas toujours identique dans tous les styles. Par exemple, dans un style, une commande provoque un saut de page et dans l'autre pas. Dans un article, il ne peut y avoir de chapitre mais toutes les autres commandes sont disponibles. Dans une lettre, il ne peut y avoir aucune séparation.

12.2 Numérotation des parties

Par défaut, LaTeX numérote les chapitres en indiquant le mot "Chapitre" suivi du numéro et du nom du chapitre à la ligne suivante. Ensuite, les sections sont numérotées en faisant référence au chapitre dans lequel elles se trouvent et le nom suit directement le numéro. Par exemple, la deuxième section du troisième chapitre sera numérotée "3.2". Un principe similaire est appliqué pour les sous-sections mais ensuite, il n'y a plus de numérotation. On dira que la profondeur de numérotation est de 2 (car deux numéros peuvent suivre le numéro de chapitre).

Il est possible de modifier la profondeur de numérotation de sorte de LaTeX pratique une numérotation plus en profondeur. Cela est obtenu en redéfinissant le compteur *secnumdepth* au moyen de la commande
`\setcounter{secnumdepth}{3}`
si on désire une numérotation jusqu'aux sous sous-sections, c'est-à-dire une profondeur de 3.

12.3 Les annexes

Dans un livre ou un rapport par exemple, il y a encore une subdivision de document qu'il est possible de créer, les *annexes*. Elles sont placées uniquement en fin de document. Pour les obtenir, on insère la commande `\appendix` dans le fichier d'entrée et la commande `\chapter` provoque non plus l'affichage du mot *chapitre* avec un chiffre mais le mot *annexe* avec une lettre. Le reste des subdivisions ne changent pas mais les numérotations sont bien entendu adaptées. **Frédéric Geraerds 1997**

13 Les espaces

Dans le texte du fichier d'entrée, placer un espace ou dix, cela revient au même. Il est donc impossible d'utiliser les espaces pour aligner du texte. Un retour chariot est également considéré comme un espace et non comme la fin d'un paragraphe. La seule exception à cette règle est dans l'environnement *verbatim* pour lequel le texte entré correspond exactement au texte imprimé. Remarquons que lŠutilisation des tabulations (voir 21.4) est fort complexe en LaTeX et elles ne sont envisageables que pour de brefs extraits.

13.1 Espace entre lignes

Par défaut, l'espace laissé entre deux lignes par LaTeX est minimum. Le texte est alors assez condensé et peut être difficile à lire. Pour modifier ce paramètre, il suffit de changer la dimension `\baselineskip` qui correspond à l'espace entre deux lignes consécutives. Ce changement peut être opéré n'importe où dans le document, la valeur en fin de paragraphe étant d'application pour celui-ci. Une fois changée, elle le reste jusqu'à une autre modification.

Par exemple, l'instruction `\baselineskip=16pt` signifie qu'au sein d'un paragraphe, entre le haut d'une ligne et le haut de la ligne suivante, il y a 16 points d'espacement.

13.2 Espace horizontal

La commande la plus simple pour ajouter un espace est `\␣` (un backslash suivit d'un espace). Celle-ci ajoute un espace de la valeur d'un caractère (comme un espace traditionnel dans un traitement de texte).

Il existe en mode mathématique quatre commandes dont une (`\,`) est utilisable également en mode texte. Elle produit un petit espace.

`\hspace{esp}`, `\hspace*{esp}`
: Produit un espace horizontal de `esp` (une longueur positive ou négative). Dans le premier cas, cet espace disparaît s'il survient à une coupure de ligne, dans le second pas.

13.3 Espace vertical

Les commandes suivantes permettent d'ajouter un espace vertical supplémentaire entre deux éléments

`\vspace{exp}` et `\vspace*{esp}`
: Produit un espace vertical de `esp` (longueur positive ou négative). Si cette commande apparaît dans un paragraphe, l'espace est passé en dessous de la ligne dans laquelle elle apparaît. L'espace disparaît dans le premier cas s'il se produit à une coupure de colonne.

`\smallskip`, `\medskip` et `\bigskip`
: Ces commandes sont des raccourcis pour des `\vspace{...}` avec des longueurs prédéfinies qui dépendent du style du document.

13.4 Espace extensible

La longueur fournie aux commandes ci-dessus est fixe c'est-à-dire qu'elle est connue lors de la rédaction du document. Il existe également des longueurs extensibles c'est-à-dire qu'elles prennent le plus d'espace possible sur une page. La commande `\fill` agit comme cela et `\hfill` est un raccourcis pour `\hspace{\fill}`. Son effet est simple. Ce qui se trouve après est mis le plus à droite possible, donc contre la marge de droite dans un paragraphe habituel. Si deux de ces commandes se trouvent sur une même ligne, l'élément du milieu se trouve au centre de l'espace qu'il y a entre les deux éléments extérieurs. La commande `\vfill` est un raccourcis pour `\vspace{\fill}` et agit de manière similaire.

La commande `\dotfill` (`\hrulefill` resp.) agit comme `\hfill`, au fait près qu'elle remplit le vide par des pointillés (une ligne pleine resp.).

Exemple *LŠexemple le plus courant dŠutilisation de ces commandes est la création dŠune page de garde et lŠalignement dŠun texte à la marge de droite. Pour la page de garde, on place un premier texte qui doit apparaître au sommet de la page, ensuite un*

et le titre qui devra être centré en hauteur, encore

et le pied de page.

13.5 Espace insécable

Dans de nombreuses circonstances, il faut séparer deux mots par un espace mais il est interdit que ceux-ci soient séparés par une fin de ligne. Par exemple, si on écrit "Vous trouverez en page 5 le...", le chiffre 5 ne doit pas être sur une autre ligne que le mot page mais séparé d'un espace. Pour cela, dans le fichier d'entrée, on remplace l'espace entre les deux mots par un tilde (~). LaTeX considère alors les deux mots comme un seul pour faire sa mise en page. **Frédéric Geraerds 1997**

14 Les coupures

14.1 Mots

LaTeX met en page les paragraphes afin de limiter autant que possible les espaces entre les mots. Afin d'effectuer au mieux cette tâche, il coupe automatiquement les mots trop longs en fin de ligne en plaçant un trait d'union. Malheureusement, sauf configuration de la part de l'utilisateur, il utilise des règles de coupure anglaises qu'il n'est pas facile de modifier. Il arrive alors d'avoir de mauvaises coupures en fin de ligne qu'il faut manuellement forcer au bon endroit. Il est également possible de créer un LaTeXrespectant les règles en vigueur pour le français mais ce nŠest généralement pas le cas.

Il est également possible de forcer LaTeX à ne pas appliquer ses règles de coupure pour un ou plusieurs paragraphes. Il ne faut cependant pas utiliser trop souvent cette possibilité car la présentation n'est pas toujours très estétique.

Ceci peut s'obtenir au moyen des instructions suivantes.

`\-`
: Utilisée dans un mot, cette commande autorise LaTeX à couper le mot à cet endroit. Le mot ne peut être coupé en un autre endroit. Plusieurs commandes peuvent se trouver dans un même mot. La coupure dans le texte final est marquée par un trait d'union.

`\fussy` et `\sloppy`
: Ces deux commandes définissent la manière dont LaTeX réagit lorsqu'il se trouve face à un `overfull hbox` (voir 24). La valeur par défaut est `\fussy`. Dans ce cas, si le texte est trop long pour tenir sur la ligne et qu'une coupure ne peut être effectuée en fonction des règles du logiciel, il y a génération d'un avertissement et le texte est laissé sur la ligne, dépassant de la sorte dans la marge de droite. Dans l'autre cas, le texte est mis à la ligne et produit également un avertissement. La manière dont un paragraphe est compilé dépend de la valeur en fin de celui-ci. Ces commandes s'appliquent jusqu'à la fin de l'environnement courant.

`\begin{sloppypar}`*pars*`\end{sloppypar}`
: Compile le(s) paragraphe(s) *pars* avec l'option `\sloppy`.

`\hyphenation{`*mots*`}`
: Déclare une liste de mots pouvant être coupés. Les coupures autorisées sont indiquées par des tirets (-). Cette déclaration est globale (valable dans tout le document à partir du point où elle se situe).

  Par exemple, `\hyphenation{envi-ron-ne-ment}` autorise la coupue du mot *environnement* à trois endroits uniquement.

14.2 Lignes

Les retours chariots dans le fichier d'entrée sont interprétés comme des espaces lors de la compilation d'un texte et une ligne vide de tout caractère indique un nouveau paragraphe. Il est possible de créer un retour marge sans provoquer un nouveau paragraphe au moyen des commandes décrites ci-dessous.

`\linebreak[num]` et `\nolinebreak[num]`
: La commande `\linebreak[num]` encourage et `\nolinebreak[num]` décourage une coupure de ligne suivant `num` (argument optionnel) qui est un nombre de 0 à 4 (défaut 4). Une valeur de 4 oblige la coupure ou l'interdit. Un avertissement (`underfull hbox`, voir 24) se produit si trop d'espace entre les mots résulte de cette commande.

`\\[len]` et `\newline`
: Les deux commandes provoquent une coupure de ligne sans justification de la ligne sur laquelle cette coupure se produit et sans retrait de paragraphe pour la nouvelle ligne. Le texte est aligné à gauche. Le paramètre optionel `len` permet d'ajouter un espace vertical supplémentaire entre les lignes. Ces commandes ne peuvent se trouver en fin de paragraphe sinon elles provoquent un avertissement.

14.3 Colonnes et pages

Tout comme pour les lignes, LaTeX dispose au mieux le texte sur la hauteur de la page tout en respectant des règles précises. Cependant, il arrive qu'il termine la page à un endroit inopportunt et il est alors nécessaire de le corriger manuellement en insérant un saut de page ou en le forçant à ne pas couper à un endroit particulier.

Les commandes qui sont décrites ci-dessous ont souvent un comportement différent suivant le style du document ainsi qu'en fonction de ses options de présentation. Il faut également bien faire attention à l'emplacement que vont prendre les *éléments flottants* (voirr̃efflottants) qui peuvent être reportés après un saut de page si on n'y prend garde.

`\pagebreak[num]` et `\nopagebreak[num]`
: La commande `\pagebreak[num]` facilite et la commande `\nopagebreak[num]` décourage la coupure d'une colonne suivant le paramètre `num` (nombre de 0 à 4, défaut 4). Lorsqu'elles sont utilisées dans un paragraphe, elles s'appliquent à la fin de la ligne dans laquelle elles apparaissent.

`\samepage`
: La commande `\samepage` prévient d'une coupure de page dans certains cas (voir guide de référence p. 90).

`\clearpage`
: Elle oblige LaTeX à afficher tous les tableaux et figures qui ne le sont pas encore et à debuter une nouvelle page.

`\cleardoublepage`
: La commande `\cleardoublepage` agit comme `\clearpage` et en plus, dans le cas d'un texte sur deux faces, force le passage à une page de droite.

`\newpage`
: La commande `\newpage` force le passage à une nouvelle colonne mais ne force pas l'affichage des tableaux et figures.

`\enlargethispage{`*size*`}`
: Cette commande permet de forcer LaTeXà agrandir une page afin de contenir une ligne en plus ou en moins. LŠargument size doit être une longueur positive ou négative.

Frédéric Geraerds **1997**

15 Les récapitulatifs

Nous envisageons dans cette section les commandes qui permettent d'extraire de l'information qui se trouve tout au long du texte. La forme la plus courante de récapitulatif est la table des matières mais il y a également la liste des tableaux et celle des figures ainsi que l'index. Ce dernier est un peu plus complexe à créer mais LaTeX nous fournit une aide précieuse pour cette pénible tâche.

15.1 Table des matières

La création d'une table des matières est très simple et entièrement automatique. La commande `\tableofcontents` insère à l'endroit où elle est placée, une table des matières qui comprend, par défaut, les numéros, noms et pages des chapitres, sections et sous-sections du document.

La table des matières est précédée d'un titre. Suivant le style du document, elle provoque ou non le passage à la page avant et après. Elle est placée à l'endroit même où on place l'instruction dans le fichier d'entrée.

Comme pour la numérotation des parties (voir 12.2) d'un document, il est possible de changer la profondeur des éléments insérés dans la table des matières en changent le paramètre `tocdepth`. Si on désire avoir une profondeur de 3 pour la table des matières, on place la ligne suivante dans le préambule du document.
`\setcounter{tocdepth}{3}`

Remarque *Afin d'obtenir la table des matières correcte, il est impératif de compiler deux fois le document. LaTeX doit en effet mettre à jour son fichier dans lequel il mémorise les informations pour la construction de la table.*

15.2 Listes des tableaux et des figures

Les tableaux et les figures sont les deux éléments flottants de LaTeX que nous introduisons dans la section 20. Nous y avons écrit qu'il était possible d'associer un titre à une telle structure au moyen de la commande `\caption`, celle-ci pouvant avoir un argument optionnel qui serait inséré dans la liste récapitulative correspondante. Si l'argument optionnel est absent, c'est le titre qui est placé dans cette liste.

La table des figures (des tableaux resp.) reprend la listes des titres des figures (des tableaux resp.) qui sont présents dans le document avec l'indication de la page sur laquelle ils se trouvent. Cette liste est précédée d'un titre (dépendant du type de la liste) et comme pour la table des matières, la présentation dépend du style du document.

Les deux commandes pour l'obtention des listes récapitulatives sont `\listoftables` pour les tableaux et `\listoffigures` pour les figures.

15.3 Index

Comme dit plus haut, l'index n'est pas aussi simple à créer que la table des matières ou les listes des tableaux ou des figures. En effet, les titres des parties ou des éléments flottants sont clairement indiqués dans le document de telle sorte que LaTeX peut aisément les extraires pour en faire un récapitulatif. Pour l'index, il est inconcevable d'y inclure tous les mots du document et LaTeX ne peut donc pas décider sans aide des mots qui doivent y apparaître. On ne peut pas non plus préciser une liste de mots dont LaTeXdevrait rechercher les occurrents car il est possible quŠà certains endroits il est en majuscule (donc sans accents), à dŠautres, il est au pluriel,... L'utilisateur devra donc inclure des instructions dans son document afin que le compilateur les mémorise pour en faire un index.

Pour concevoir un index, il faut exécuter plusieurs opérations que nous décrivons ci-dessous.

1. Il faut placer la commande `\makeindex` dans le préambule du document. Celle-ci a pour effet de créer un nouveau fichier d'extension `.idx` (Si le fichier d'entrée s'appelle `myfile.tex`, le fichier créé s'appelle donc `myfile.idx`.) qui contiendra des instructions associant le mot devant apparaître dans l'index et la ou les pages sur lesquelles il se trouve. Par exemple, si le mots *compiler* apparaît sur la page 24 du document, il contiendra la commande `\indexentry{compiler}{24}`.

2. Chaque fois que l'on désire faire apparaître un mot dans l'index, supposons encore *compiler*, on doit placer à côté de lui la commande `\index{compiler}`. Par exemple, on peut avoir la ligne
   `Il faut compiler\index{compiler} le fichier...`
   Il est conseillé de placer la commande contre le mot sans laisser d'espace, comme cela, la référence à la page sera correcte même si c'est le dernier mot de la page. Remarquons que lŠon peut très bien mettre cette instruction à côté de nŠimporte quel mot car LaTeX ne fais pas attention à la sémantique mais uniquement à la syntaxe.

   Remarqons encore que cette commande ne produit aucune sortie dans le fichier dvi, c'est juste la demande d'écriture d'une instruction dans le fichier idx. De plus, l'argument de cette commande peut comprendre n'importe quel caractère, y compris les caractères réservés au fait près que les accolades doivent être présentes par paire (une ouvrante et une fermante). Auncun caractère spécial ne peut être utilisé si la commande se trouve comme argument d'une autre commande.

3. A partir du fichier idx, il faut créer un ficher d'entrée compréhensible pour LaTeX. De nombreux programmes fournissent une aide pour la création de ce document. Le nouveau document sera alors inclu dans le fichier d'entrée pour apparaître dans le fichier imprimable.

Remarque *Des instructions tout à fait équivalentes existent pour créer un glossaire. On a* `\glossary` *et* `\glossaryentry`.

Frédéric Geraerds 1997

16 Les en-têtes et pieds de page

La définition des en-têtes et des pieds de page est avant tout contrôlée par le style du document. Suivant celui-ci, ces éléments peuvent être différents sur la page de gauche et sur la page de droite. Un de ces deux éléments contient généralement la numérotation des pages que nous envisageons également ici.

16.1 Le style de page

La déclaration d'un style de document au début entraine de manière implicite la définition d'un style de page par défaut. Il est cependant loisible à l'utilisateur de modifier celui-ci pour une ou plusieurs pages au moyen des commandes décrites dans cette section.

`\pagestyle{`*style*`}` Il y a essentiellement 4 styles différents qui peuvent être fournis comme argument à cette commande.

plain est le style par défaut pour lequel il n'y a généralement que la numérotation des pages qui apparaît dans le pied de page.

empty permet de ne rien inclure dans les deux éléments.

headings est dépendant du style du document. Généralement, le pied de page est vide et l'en-tête contient des informations dépendant du chapitre et de la section dans lesquel on se trouve.

myheadings est le style permettant à l'utilisateur de définir soi-même le contenu des deux éléments au moyen des commandes `\markboth` et `\markright`.

La commande `\pagestyle` peut apparaître n'importe où dans le document mais son emplacement le plus courant est au début d'un chapitre.

`\thispagestyle{`*style*`}` Comme la commande `\pagestyle` sauf que son application est limitée à la page courante, c'est-à-dire celle ou apparaît la commande lors de la compilation.

`\markright{`*right*`}` et `\markboth{`*left*`}{`*right*`}` Ces deux commandes permettent de personnaliser les en-têtes des pages. Nous laissons le soin à l'utilisateur de tester ces commandes.

16.2 Numérotation des pages

La numérotation des pages fait partie soit de l'en-tête soit du pied de page. A l'aide du style, il est déjà possible de les changer de position et de les supprimer. Voyons à présent comment changer la forme de la numérotation en passant par exemple de nombres arabes traditionnels en nombres romains.

La commande `\pagenumbering{`*style*`}` permet de préciser le format des nombres. Dans celle-ci, *style* est un des mots `arabic` (nombres arabes), `roman` (nombres romains minuscules), `Roman` (nombres romains majuscules), `alph` (lettres minuscules) ou `Alph` (lettres majuscules).

Par exemple, pour obtenir un document qui commence par la numérotation romaine pour l'introduction, la table des matières,... et qui se poursuit par la numérotation arabe dans le premier chapitre, placez la commande `\pagenumbering{roman}` avant le texte et `\pagenumbering{arabic}` juste après la commande `\chapter` du premier chapitre.

Frédéric Geraerds 1997

17 Les références croisées

Dans un document, on désire assez souvent se référer à un numéro de page où se trouve une certaine information, au numéro d'un théorème, d'une définition, d'un tableau, d'une figure,... ou encore au numéro d'une subdivision (chapitre, section,...) du document. C'est ceci que nous appelons les *références croisées* de LaTeX.

17.1 Définition d'une étiquettte

Afin d'en expliquer le principe, considérons un exemple. Pour obtenir le titre de cette section, nous avons tapé

```
\section{Références croisées}
```

et nous y avons ajouté `\label{crossref}` juste à côté. Cet ajout ne produit aucun texte dans le document mais positionne une étiquette (ici `crossref`) à cet endroit. Lors de la compilation, LaTeX inscrit dans son fichier aux le nom de cette étiquette ainsi que le numéro de la section à laquelle elle se rapporte et la page sur laquelle elle apparaît.

Une étiquette peut être utilisée dans tout environnement portant un numéro (les théorèmes, les chapitres, les sections,...). Lorsqu'elle est utilisée avec la commande `\caption` dans une figure ou un tableau, la commande `\label` doit se trouver après cette instruction.

17.2 Numéro associé à une étiquette

Lorsqu'une étiquette est définie, on peut insérer dans le document les informations qui y sont associées. Considérons tout d'abord les cas du numéro associé.

Pour notre exemple, la commande `\ref{crossref}` produit le texte *17*. Celle-ci peut être placée aussi bien avant qu'après la définition de l'étiquette mais il est primordiale de compiler deux fois le document avant d'obtenir les références correctement. Notons quand même que cette instruction placée dans un environnement tel que `verbatim` ne produira pas le texte souhaité de par la nature même de cet environnement.

17.3 Page associée à une étiquette

Exactement comme pour le numéro, il est possible d'insérer la page associée à une étiquette dans le texte au moyen de la commande `\pageref`. Ainsi, `\pageref{crossref}` produira 23.

Frédéric Geraerds **1997**

18 La bibliographie

Une partie importante d'un document est sa bibliographie. Elle doit être rigoureuse et claire. LaTeX vous aide en proposant un environnement spécialement conçu pour cela.

18.1 Création

La création de la bibliographie se fait au moyen de l'environnement appelé `thebibliography`. La syntaxe en est la suivante

`\begin{thebibliography}{`*max-larg*`}`
entrée
`\end{thebibliography}`
où

max-larg représente la largeur maximale des étiquettes des entrées de la bibliographie. Ce paramètre contrôle la mise en page. Par exemple, si on a 7 éléments dans la bibliographie, on indiquera 9 et si on en a entre 10 et 99, on placera 99.

entrée est une liste d'entrée commançant par
`\bibitem[`*etiq*`]{`*citeEtiq*`}`
qui génère une entrée dont l'étiquette est *etiq*. Si ce paramètre est manquant, c'est le numéro de l'entrée qui est affiché. Le second paramètre est l'étiquette pour les références par la commande `\cite` (voir plus bas).

18.2 Référencer un objet de la bibliographie

La référence à un objet de la bibliographie se fait de manière similaire à celle d'une étiquette (voir section 17.1). La différence est que l'on utilise la commande
`\cite[`*texte*`]{`*citeEtiqListe*`}`
qui produit l'affichage des étiquettes (définies par la commande `\bibitem`) correspondant aux étiquettes de référence de la liste. Si *texte* est présent, il est ajouté comme commentaire à cette référence.

18.3 Programme spécifique

Il existe également des programmes qui permettent de gérer de manière plus perfectionnée les bibliographies. Dans une mise à jour ultérieure de ce document, nous espérons décrire un de ces programmes particuliers. **Frédéric Geraerds** **1997**

19 Sectionner le fichier d'entrée

Nous avons déjà introduit la notion de fichiers de style (voir section 10) qui permettent de regrouper les définitons de commandes, les commandes de mise en page,... Un fichier de style sera alors généralement inséré dans le préambule du document.

Lorsque l'on rédige un long document, il n'est pas intéressant de travailler dans un seul fichier. En effet, le temps de sauvegarde devient plus long et la navigation n'y est pas toujours aisée, surtout que l'on travaille dans un éditeur de textes. LaTeX propose alors plusieurs manières de s'organiser pour palier à ces inconvénients. Nous décrivons ici le principe que nous adoptons pour la création de nos documents relativement longs.

19.1 Les commandes d'inclusion

LaTeX propose deux commandes pour insérer le contenu d'un fichier dans un autre.

`\input[`*file*`]` importe le texte qui se trouve dans le fichier *file* dans le fichier qui contient la commande, exactement comme si ce texte faisait partie de ce document. Cette commande peut être utilisée n'importe où dans le document (sauf dans un environnement `verbatim`) et peut même apparaître dans un fichier appelé par une de ces deux commandes.

`\include[`*file*`]` importe le texte qui se trouve dans le fichier *file* dans le fichier qui contient la commande, exactement comme si ce texte faisait partie de ce document au fait près qu'il insère un saut de page avant et après ce texte. Elle ne peut ni apparaître dans le préambule ni dans un document inclus par cette commande.

Avec cette deuxième commande, il y en a une autre associée.

`\includeonly[`*file-list*`]` conditionne les inclusions de la commande `\include`. C'est-à-dire que si le fichier *file* de la commande `\include` est un des noms de la liste *file-list* (liste de noms séparés par des virgules), le fichier est compilé par LaTeX alors que s'il n'apparaît pas dans la liste, les références et les entrées de tables récapitulatives sont conservées mais le fichier n'est pas compilé. Il y a alors un gain de temps apréciable lors de la compilation.

Remarquons que la commande `\includeonly` ne peut apparaître que dans le *préambule* du document.

19.2 Conseil pratique

Nous présentons dans cette section, la méthode que nous utilisons pour sectionner notre document.

Nous plaçons chaque chapitre dans un fichier différent. De même pour la bibliographie et l'introduction. Nous créons également un fichier principal (*fichier maître*) qui contiendra la structure générale du document avec son préambule. Pour seules lignes de texte, il y aura les commandes d'inclusion `include` et éventuellement des commandes pour la tables des matières, la numérotation des pages,.... Dans le *préambule*, nous plaçons également la commande `includeonly` avec le nom du fichier qui contient le chapitre sur lequel on travaille. En général, on ne travaille que sur un chapitre mais il n'y a aucune restriction et on peut y placer la liste des fichiers qui sont en cours de modification. Cette subdivision permet d'économiser beaucoup de temps de compilation et conserve toujours les références croisées aisni que la numérotation des pages. **Frédéric Geraerds 1997**

20 Les éléments flottants

S'il y a une structure que je n'ai jamais rencontrée ailleurs, c'est bien celle des *éléments flottants*. Remarquons qu'elle est possible car LaTeX est un compilateur de textes plutôt qu'un traitement de textes au sens habituel du terme. Elle est principalement utilisée pour insérer des images, des dessins ou des tableaux dont la position dans le texte n'a que peu d'importance car on le désignera au moyen d'une référence croisée (voir section 17).

20.1 Les deux types

LaTeX propose deux environnements différents pour positionner ces éléments flottants. Ceux-ci ont le même effet au fait près qu'ils identifient les éléments par des noms différents et des numérotations distinctes (voir la commande `\caption` plus bas).

`table` est libellé par Tableau et
`figure` est libellé par Figure.

La commande pour ces environnements est
`\begin{`*name*`}[`*loc*`]`
body
`\end{`*name*`}`
où

name est le nom de l'environnement (`table` ou `figure`),

loc est une séquence d'au plus 4 caractères parmi `h t b p` désignant l'emplacement désiré (`h (here)` emplacement à l'endroit d'insertion dans le texte si cela est possible, `t (top)` en haut d'une page, `b (bottom)` au bas de la page, `p (page of floats)` sur une page séparée ne contenant que des éléments flottants), un `!` permet en plus dŠignorer les paramètres.

body est le texte (ou autre) à placer.

L'utilité de ces environnements est très simple. Supposons que l'on désire placer un tableau qui demande les trois-quarts d'une page. Si ce tableau est inséré sur une page qui contient une moitié de texte, il y aura un blanc d'une demi page. Avec les éléments flottants, le tableau sera placé au début de la page qui suit mais du texte sera inséré sur la demi page qui aurait été vide (pour autant qu'il y ait du texte qui suit le tableau).

20.2 Identification et numérotation

Pour obtenir l'intitulé de l'environnement ainsi qu'un numéro, il faut utiliser la commande
`\caption[`*entree*`]{`*texte*`}`
qui provoque l'affichage du texte après l'intitulé et le numéro. Plusieurs commandes `\caption` peuvent apparaître dans un même environnement. Le texte optionnel *entree* sert d'entrée pour la liste des figures ou tableaux.

20.3 Liste récapitulative

Rappelons qu'il est possible de créer une liste des tableaux et une liste des figures. Pour plus de détail, le lecteur consultera la section 15.2 en page 21.

20.4 Paramètres

Parmi les paramètres qu'il est possible de fournir lors de la création d'un élément flottant, il y a la position d'insertion dans le texte. Si on précise `h` pour une insertion immédiate dans le texte, il est possible que LaTeX déplace malgré tout l'élément vers le début de la page suivante. En effet, s'il ne dispose plus d'assez de place sur la page, il va le positionner à cet endroit et continuer la page précédente avec le texte qui suivait.

Si on utilise abondamment les éléments flottants, il se peut qu'il y en ait plusieurs en début de page. Cependant, le nombre d'éléments et la proportion d'espace occupé par ceux-ci est contrôlée par LaTeX. Le nombre d'éléments est contrôlé par le compteur `\topnumber` et la proportion par la commande `\topfraction`. Par exemple, pour autoriser les éléments flottants à occuper 75% de la page, on insère la ligne suivante dans le préambule.
`\renewcommand{\topfraction}{0.75}`
Il existe également le compteur `\bottomnumber` et la commande `\bottomfraction` pour contrôler les espaces flottants dans le bas de la page. De nombreux autres paramètres existent et le lecteur consultera le guide de référence pour en savoir plus.

Il existe aussi une commande permettant dŠannuler les paramètres pour une page particulière. Plus précisément, il est possible de supprimer le placement dŠéléments flottant sur une page au moyen de la commande
`\suppressfloats[`*pos*`]`
où *pos* est un argument optionel qui vaut soit `b` soit `t`. Attention que lŠutilisation dŠun ! dans les arguments dŠun élément flottant est prioritaire sur cette commande. **Frédéric Geraerds1997**

21 Les environnements prédéfinis

Nous avons déjà cité quelques environnements qui sont prédéfinis par LaTeX. Nous décrivons ici leur fonctionnement avec plus de précisions.

21.1 Alignement du texte

Dans la plupart des ouvrages, le texte est justifié, c'est-à-dire qu'une ligne de texte débute à la marge de gauche et se termine à la marge de droite, les espaces entre les mots étant ajustés en fonction de cet impératif. Pour certains extraits, on peut désirer supprimer cette contrainte en choisissant d'aligner le texte à gauche ou à droite ou de le centrer sur la largeur de la page. Pour cela, on utilise l'un des trois environnements `center`, `flushleft` ou `flushright`

`center` est utilisé pour produire une ou plusieurs lignes de texte centré sur la page. La commande `\\` permet de passer à la ligne. Les paragraphes sont également permis mais ne sont pas indentés.

`flushleft` produit un texte aligné à gauche et non justifié (les lignes ne se terminent pas toutes à la marge de droite).

`flushright` produit un texte aligné à la marge de droite et non justifié (donc ne commence pas nécessairement à la marge de gauche).

Les commandes `\centering`, `\raggedleft` et `\raggedright` modifient les paramètres de mise en page de LaTeX et s'utilise dans un environnement pour produire un effet similaire aux trois environnements définis ci-dessus. Le placement d'une de ces commandes dans l'environnement principale change donc la pagination de la suite du document tout entier.

21.2 Les dérivés des listes

LaTeX propose trois environnements basés sur les listes. La structure de ceux-ci est identique mais c'est la manière de présenter le texte qui diffère.

Pour les trois, chaque nouvel élément débute par la commande `\item`. Elle accepte un argument optionnel dont l'effet dépend de l'environnement.

Remarque *LATEXpropose un environnement plus général intitulé* `list` *qui permet à lŠutilisateur de définir ses propres environnements de liste. Nous espérons introduire celui-ci dans une future mise à jour de ce guide.*

LATEX autorise qu'un élément d'une liste soit également une liste. Dans les explications ci-dessous, un élement d'une liste insérée dans le document est de profondeur 1. Si un élément d'une liste est encore une liste, les éléments de cette deuxième liste auront une profondeur 2 et ainsi de suite.

21.2.1 Liste de points

Par défaut, `itemize` commence chaque nouvelle ligne par un symbole dépendant du style et de sa profondeur. Par exemple, un gros point noir pour les éléments de la profondeur 1, un tiret pour ceux de profondeur 2,...

L'argument optionnel d'un nouvel élément remplace le symbole. Il est aligné à droite avec les autres symboles et peut même déborder dans la marge de gauche sans provoquer d'erreur.

Voici un exemple.

- Voici le premier élement de cette liste. Vous constatez que les lignes sont décalée de la marge de gauche du texte et sont alignées derrière le symbole. Les lignes sont cependant justifiées comme dans tout le texte si on ne modifie pas cette option de LATEX.
  - Comme deuxième élément de la liste, nous avons placé une autre liste. On remarque le changement de symbole pour désigner un nouvel élément ainsi que le décalage de l'alignement vers la droite.
  - Le deuxième élément de cette liste s'aligne sur le premier.

N.B. Fournir un argument optionnel pour un nouvel élément provoque le remplacement du symbole.

Cette liste est obtenue par le texte d'entrée suivant.

```
\begin{itemize}
\item Voici le premier élement de cette liste. Vous constatez que les lignes sont
décalée de la marge de gauche du texte et sont alignées derrière le symbole. Les
lignes sont cependant justifiées comme dans tout le texte si on ne modifie pas cette
option de \LaTeX.
\begin{itemize}
\item Comme deuxième élément de la liste, nous avons placé une autre liste. On
remarque le changement de symbole pour désigner un nouvel élément ainsi que le
décalage de l'alignement vers la droite.
\item Le deuxième élément de cette liste s'aligne sur le premier.
\item[N.B.] Fournir un argument optionnel pour un nouvel élément provoque le
remplacement du symbole.
\end{itemize}
\end{itemize}
```

21.2.2 Enumératon

Une énumération est fort similaire à une liste de points. La différence est le symbole qui identifie un nouvel élément. Dans ce cas, le symbole n'est plus identique pour chaque élément d'une même profondeur mais c'est un symbole qui est incrémenté à chaque nouvel élément. Par exemple, pour la profondeur 1, c'est un nombre et pour la 2, c'est une lettre.

Un argument optionnel remplace encore une fois le numéro et ne provoque pas son incrémentation comme le montre l'exemple ci dessous.

1. Le premier élément de profondeur 1 est numéroté par un chiffre.
   (a) Un énumération dans une autre provoque le changement du type de numérotation
   N.B. L'argument optionnel annule la numérotation
   (b) C'est le troisième élément mais il est identifié par le deuxième symbole car l'argument optionnel du deuxième élément ne provoque pas l'incrémentation.

Cette énumération est obtenue par le texte suivant.

```
\begin{enumerate}
\item Le premier élément de profondeur 1 est numéroté par un chiffre.
\begin{enumerate}
\item Un énumération dans une autre provoque le changement du type de numérotation
\item[N.B.] L'argument optionnel annule la numérotation
\item C'est le troisième élément mais il est identifié par le deuxième symbole car
l'argument optionnel du deuxième élément ne provoque pas l'incrémentation.
\end{enumerate}
\end{enumerate}
```

21.2.3 Description

Une description se présente comme les deux précédents environnements mais il n'y a pas de symbole qui identifie un nouvel élément. Cependant, la première ligne débute avec un retrait négatif par rapport aux autres. Si un argument optionnel est fourni, il est placé en gras avant la ligne et sert donc d'identificateur de la ligne.

Comme aucun argument optionnel n'est fourni, le texte est simplement placé légèrement en retrait.

option L'argument optionnel (option) est placé en gras et en retrait. Observez comment se comporte da deuxième ligne de texte dans une description.

Ce texte est obtenu grâce aux lignes suivantes.

```
\begin{description}
\item Comme aucun argument optionnel n'est fourni, le texte est simplement placé
légèrement en retrait.
\item[option] L'argument optionnel (option) est placé en gras et en retrait.
Observez comment se comporte da deuxième ligne de texte dans une description.
\end{description}
```

21.3 Citation et poème

LaTeX propose trois environnements pour les citations dans lesquels les marges gauche et droite sont déplacées de la même distance. Un poème est similaire à ceux-ci.

`quote` est utilisé pour de brèves citations, séparées par un ligne vide (dans le texte d'entrée). Il n'y a pas de retrait de paragraphe et un espace vertical supplémentaire est ajouté entre les paragraphes.

quotation est utilisé pour des citations de plus d'un paragraphe, séparés par une ligne vide également. Il y a un retrait de paragraphe et l'espace normal entre les paragraphes est utilisé.

verse est utilisé pour un poème. Les lignes d'une même strophe sont séparées par \\ et les strophes sont determinées par une ou plusieurs lignes vides.

poetry Cet environnement est similaire à `verse`. Une nouvelle strophe s'obtient en laissant une ligne vide. Une nouvelle ligne s'obtient par un double \.

Nous laissons le soin à lŠutilisateur de tester ces environnements que nous utilisons rarement.

21.4 Tabulation

Les tabulations sont assez complexes à réaliser en LaTeX et doivent être limitée autant que possible à notre avis. Nous allons tout d'abord décrire systématiquement le fonctionnement des tabulations et ensuite envisager un exemple qui, nous l'espérons, sera beaucoup plus compréhensible.

Cet environnement `tabbing` produit une séquence de lignes (en mode texte) avec un alignement basé sur des tabulations. Les tabulations sont numérotées 0, 1, 2,... Une tabulation est dite définie si on lui a assigné une position horizontale sur la page. La tabulation 0 est toujours définie par défaut (côté gauche de la page) et si une tabulation i (un nombre) est définie, les précédentes le sont également.

Le fonctionnement est basé sur les valeurs `next_tab_stop` et `left_margin_tab`. Initialement la valeur de `next_tab_stop` est 1 et celle de `left_margin_tab` est 0. La valeur de `next_tab_stop` est incrémentée par `\=` ou `\>` et réinitialisée par `\\` ou `\kill`. Les commandes suivantes peuvent être utilisées dans l'environnement.

\= Si la valeur de `next_tab_stop` est i, alors la tabulation i est initialisée à cette position et la valeur de `next_tab_stop` est incrémentée.

\> si `next_tab_stop` vaut i, alors le texte qui suit débute à la tabulation i (même si elle est à gauche de la position où l'on se trouve) et la valeur de `next_tab_stop` est incrémentée.

**** Débute une nouvelle ligne en initialisant `next_tab_stop` à `left_margin_tab`. Donc le texte de la nouvelle ligne débutera à la tabulation `left_margin_tab`.

\kill Elimine la ligne courante en gardant les définitions des tabulations de cette ligne, initialise `next_tab_stop` à `left_margin_tab`. Elle permet de définir des tabulations en se basant sur une ligne fictive.

\+ Augmente d'une unité la valeur de `left_margin_tab`.

\- Diminue la valeur de `left_margin_tab`, qui doit être positive, d'une unité.

\< Diminue la valeur de `next_tab_stop` d'une unité. Cette commande ne peut être utilisée qu'en début de ligne pour annuler l'effet d'un `\+` pour cette ligne.

\' Place le texte juste à droite de la tabulation suivante ou contre la marge de gauche.

**\`** Place le texte contre la marge de droite. Il ne peut y avoir de `\>`, `\=` ou de \ commande après sur cette ligne.

\pushtabs Sauve les positions des tabulations pour les réutiliser lors de la commande `\poptabs` suivante.

\poptabs voir `\pushtabs`

`\a...` Les commandes `\=`, `\'` et `` \` `` produisent généralement des accents mais sont redéfinies dans un `tabbing`. Les commandes `\a=`, `\a'` et `` \a` `` remplacent ces commandes.

Considérons à présent un exemple très simple. Nous allons donner le texte introduit pour créer les lignes suivantes.

pomme	50	bonne qualité
poire	41	qualité moyenne
annanas	87	médiocre

Le texte placé dans le fichier d'entrée est le suivant.

```
\begin{tabbing}
xxxxxxxxx\=xxxx\= \kill
pomme \>50 \> bonne qualité\\
poire \> 41 \> qualité moyenne\\
annanas \> 87 \> médiocre
\end{tabbing}
```

Remarque *Dans ce cas, il aurait certainement été plus facile dŠutiliser un tableau.*

21.5 Tableau

Pour mettre en forme un tableau avec LaTeX, il existe deux environnements: `tabular` et `array`. Ceux-ci s'utilisent de manière similaire. La différence entre les deux est simple: par défaut, le mode texte est utilisé pour les éléments des cellules dans un `tabular` alors que c'est le mode mathématique pour un `array`. On distingue les deux environnements essentiellement par l'utilité que l'on en fait: `array` est utilisé pour créer des matrices (ou tout autre tableau dont les éléments nécessitent le mode mathématique) et `tabular` pour créer des tableaux dont les éléments sont en mode texte.

Un tableau est considéré par LaTeX comme une boite (voir section 22) et donc doit être plus petit qu'une page.

Décrivons les notions qui sont communes aux deux. Ce que nous écrivons pour `tabular` dans ces paragraphes reste valable pour `array` sauf mention explicite du contraire.

Ces environnements nécessitent un argument supplémentaire pour spécifier le nombre de colonnes. La syntaxe d'un tableau est
`\begin{tabular}[`*pos*`]{`*cols*`}`
lignes
`\end{array}`
où

pos est un argument optionnel qui spécifie la position en hauteur du tableau sur la ligne; par défaut, le tableau est centré, un `t` aligne le dessus avec la ligne et un `b`, le dessous. C'est un paramètre courant pour les boites.

cols précise le format des colonnes. Chaque caractère correspond à une colonne ou une séparation de colonne.

- **l** Une colonne où les éléments sont alignés à gauche.
- **r** Une colonne où les éléments sont alignés à droite.
- **c** Une colonne où les éléments sont centrés.

| Une ligne verticale entre deux colonnes.

@{*texte***}** Insère le *texte* dans toutes les lignes de la colonne. Le texte est en mode texte ou mathématique suivant que l'on est dans un `tabular` ou un `array`. Cette commande annule l'espace normal entre les deux colonnes où elle se trouve.

p{*larg***}** produit une colonne de largeur *larg* où le texte est comme dans une commande `\parbox`.

{nbr***}{***cols***}** est équivalent à *nbr* copies de *cols* qui peut lui même contenir une *–expression.

lignes est une séquence de lignes séparées par `\\`. Chaque ligne est une séquence d'éléments séparés par `&` et il doit y avoir le même nombre d'éléments que de colonnes spécifiées par *cols*. Chaque élément est compilé comme s'il se trouvait entre accolades, et donc l'effet d'une macro (de changement de caractères par exemple) se limite à cet élément. Les commandes suivantes peuvent apparaître dans les lignes.

`\multicolumn{`*num*`}{`*col*`}{`*texte*`}` fait de *texte* l'élément d'une cellule qui prend la place de *num* cellules dans la lignes où elle se trouve. Si *num*=1, alors la commande sert uniquement à changer la position de cet élément dans la cellule. *col* peut contenir exactement les mêmes caractères que *cols* mais ne peut contenir qu'une seule colonne.

`\vline` Lorsqu'elle est utilisée à l'intérieur d'une colonne, elle produit une ligne verticale de la hauteur de la cellule.

Les instructions suivantes peuvent venir entre les lignes pour produire des lignes horizontales.

`\hline` produit une ligne horizontale sur toute la largeur du tableau.

`\cline{i - j}` produit une ligne horizontale de la colonne `i` à la colonne `j` incluses.

La largeur des colonnes est automatiquement ajustée à la largeur du plus large élément de cette colonne. Il ne peut pas y avoir un `&` après le dernier élément d'une ligne. On peut inclure un tableau dans un tableau (récursivement).

Considérons un exemple pour éclairer tout cela et donnons le code pour créer le tableau qui suit.

Fruit	Vente		
	1995	1996	1997
Poire	6.5	7.8	7.2
Pomme	15	16	15.5
Prune	7.8	9	9.2

```
\begin{tabular}{|l|c|c|c|}\hline
 & \multicolumn{3}{c|}{Vente}\\\cline{2-4}
\multicolumn{1}{|c|}{\raisebox{6pt}[0pt][0pt]{Fruit}}
  & 1995 & 1996 & 1997 \\\hline
Poire & 6.5 & 7.8 & 7.2 \\
Pomme & 15 & 16 &15.5 \\
Prune & 7.8 & 9 & 9.2 \\\hline
\end{tabular}
```

21.6 Texte littéral

L'environnement `verbatim` restitue exactement le texte d'entrée en sortie. Il respecte tous les caractères (même réservés) ainsi que les espaces et les fins de ligne. La police utilisée est de type *typewriter*. Cet environnement commence une nouvelle ligne et passe à la ligne lorsqu'il se termine avec ajout d'un espace vertical.

Il est possible de produire quelques caractères affichés exactement comme introduits mais qui ne se trouvent pas sur une ligne séparée. La commande à utiliser est `\verb` et sa syntaxe est un peu particulière. Cette commande prend un argument (le texte à afficher) délimité par une paire de caractères identiques (à l'exception de l'espace, de `*` et d'une lettre). Ainsi les commandes `\verb-$\é &-` et `\verb+$\é &+` produisent le même texte qui est `$\é &`.

Il y a également l'environnement `verbatim*` et la commande `\verb*`. Elles agissent comme `verbatim` et `\verb` au fait près qu'un espace produit ␣. Ainsi `\verb*-$\é &-` produit `$\é␣&`.

Il y a trois restrictions à l'utilisation de cet environnement et de cette commande: (1) ils ne peuvent apparaître dans un argument d'une commande mais bien dans un autre environnement, (2) il ne peut y avoir aucun espace entre la commande `\verb` ou `\verb*` et son argument et (3) il ne peut y avoir d'espace entre `\end` et `{verbatim}`.

Frédéric Geraerds **1997**

22 Les boites

LaTeX considère certains éléments comme des objets qu'il ne peut diviser pour les placer correctement sur une ligne ou une page. Par exemple, lorsque l'on crée un tableau avec l'environnement `tabular`, celui-ci est considéré comme un seul bloc et le compilateur ne le coupera jamais en plusieurs parties. Il y a d'autres environnements et commandes qui permettent de créer également de tels objets que l'on appelle des *boites*. Remarquons donc qu'une boite peut être très grande mais ne pourra pas dépasser la taille d'un page.

Il existe tois sortes de boites suivant le mode de compilation du texte qui se trouve à l'intérieur. Lors de la compilation, LaTeX prend le style en cours pour le texte contenu dans une boite sauf si on se trouve en mode mathématique auquel cas il prend le style qui était en cours juste avant.

22.1 Le mode LR

Les commandes ci-dessous traitent le texte qui se trouve à l'intérieur en mode LR c'est-à-dire en mode texte où il ne peut y avoir de paragraphe. Ainsi, placer une telle commande en mode mathématique permet d'insérer quelques mots en mode texte. Remarquons que comme c'est une boite, ce texte doit tenir sur une seule ligne.

`\makebox` est la commande générale pour créer un boite. Elle prend deux arguments obligatoires plus un optionnel. La syntaxe est
`\makebox{`*length*`}[`*pos*`]{`*texte*`}`
où

- *length* spécifie la longueur de la boite et doit être une unité de mesure. Par exemple `10mm`.
- *pos* précise la position du texte au sein de la boite, le défaut étant centré, la lettre `l` l'alligne à gauche et `r` à droite.
- *texte* est le contenu de la boite qui sera compilé en mode texte.

`\mbox` est un identique à la première commande au fait pret qu'elle ne prend que le texte en argument. La longueur de la boite est calculée automatiquement pour être égale à la longeur du texte.

`\framebox` fonctionne comme `\makebox` mais insère un cadre autour de la boite.

`\fbox` est un raccourcis de `\framebox` fonctionnant de la même manière que `\mbox` pour `\makebox`.

Remarquons encore qu'un espace avant ou après ces commandes produit un espace comme entre deux mots. Il est également possible d'utiliser des boites de longeur 0 (en fait, il faut spécifier `0mm`) pour aligner du texte dans un tableau par exemple ou placer un texte dans la marge.

22.2 Le mode paragaphe

Les deux commandes ci-dessous permettent de créer une boite dans laquelle le texte est compilé en mode paragraphe exactement comme dans le texte normal. Remarquons que les environnements flottants créent également de telles boites.

La commande `parbox` est le premier moyen de créer un telle boite et la syntaxe est
`\parbox[`*pos*`]{`*width*`}{`*texte*`}`
où

width spécifie la largeur de la boite. C'est une longueur et elle doit être inférieure à la largeur de la page.

texte est le contenu de la boite qui sera compilé en mode paragraphe.

pos permet de préciser l'alignement du texte par rapport à la ligne de texte dans laquelle il se trouve. Par défaut, c'est centré mais si *pos*=`t`, la ligne supérieure du bloc est alignée avec le la ligne courante du texte et si *pos*=`b`, la ligne inférieure est alignée avec le texte.

Remarquons que dans cette boite, il est possible de créer des paragraphes mais ils ne seront pas indentés comme dans le reste du texte. En effet, LaTeX change la valeur du paramète `\parindent` pour la mettre à zéro. Si on désire une indentation, il faut attribuer une nouvelle valeur à cette longueur.

L'environnement `minipage` est le second moyen de créer une boite compilée en mode paragraphe. Elle s'utilise un peu différement d'un environnement habituel car il faut lui préciser un argument obligatoire et éventuellement un facultatif de la manière suivante.
`\begin{minipage}[`*pos*`]{`*width*`}`
texte
`\end{minipage}`
Les arguments ont la même signification que ci-dessus.

La différence entre les deux est l'utilisation des notes de bas de page. Dans une *minipage*, les notes de bas de page apparaissent dans le bloc de texte produit par l'environnement. De plus, dans cet environnement, on peut utiliser tous les autres environnements comme `itemize`,... ce qui n'est pas le cas dans une commande `\parbox`.

22.3 Les boites noires

LaTeX permet de créer des rectangles complètement noir. Pour cela, on utilise la commande
`\rule[`*raise*`]{`*width*`}{`*height*`}`
où

raise permet de spécifier le décalage par rapport à la ligne. Une valeur positive le place au dessus alors qu'une négative le place en dessous.

width est la largeur du rectangle.

height est sa hauteur.

La taille de la boite est précisée par les deux derniers arguments.

Il est possible de créer une boite noire de longeur 0 mais qui a une certaine hauteur. Lors de la compilation, LATEX ajustera alors les espaces en tenant compte de cette contrainte bien qu'elle soit invisible dans le document.

22.4 Les boites décalées

Pour les boites noires, il est possible de préciser un paramètre de décalage du texte par rapport à la ligne courante. La commande vue dans cette section permet d'avoir une boite compilée en mode LR mais décalée sur la ligne. Elle permet également de modifier la taille du texte aux yeux de LATEX.

`\raisebox{`*raise*`}[`*above*`][`*below*`]{`*texte*`}`
où

raise spécifie le décalage verticale du texte par rapport à la ligne.

above est la taille que le compilateur considère comme étant au-dessus de la ligne même si effectivement ce n'est pas la réalité.

below fait de même mais en dessous de la ligne.

texte est le texte à écrire à cet endroit.

22.5 Sauvegarde

La définition de commandes permet de gagner du temps lors de la rédaction du document en raccourcissant le texte qu'il est nécessaire de taper. Cependant, à la compilation, ce texte doit chaque fois être recompilé ce qui peut occasionner une perte de temps spécialement si le texte est compliqué. C'est pourquoi LATEX permet de sauvegarder des boites, c'est-à-dire non seulement le contenu mais également la manière de le présenter. Il le considère donc comme un seul caractère.

Il y a plusieurs étapes et commandes nécessaires pour ces opérations.

Déclaration d'un nom La commande `\newsavebox{`*name*`}` permet de déclarer une variable *name* qui pourra contenir une boite. Le nom doit être un nom de commande non encore connu de LATEX. C'est une commande dont l'effet est global c'est-à-dire que si on l'utilise au sein d'un environnement, elle est encore valable en dehors de celui-ci.

Sauvegarde d'une boite Lorsque l'on a déclaré un nom de variable, il est possible de lui attribuer une boite au moyen de
`\savebox{`*name*`}[`*length*`]{`*texte*`}`
où

name est le nom de la variable dans laquelle on désire sauvegarder la boite.

length est la longueur de la boite.

texte est le contenu de la boite que l'on désire mémoriser.

Cette commande a un effet local uniquement c'est-à-dire que si on l'utilise dans un environnement, la valeur mémorisée n'est valable que dans celui-ci et ceux inclus dans celui-ci.

Il existe la commande `\sbox` qui est identique au fait près qu'elle n'a pas d'argument optionnel précisant la longueur.

Utilisation d'une boite sauvegardée Après sauvegarde d'une boite dans une variable *name*, on peut l'utiliser au moyen de la commande `\usebox{`*name*`}`.

23 Formules mathématiques

Des formules mathématiques peuvent être placées à l'intérieur du texte ou sur un ligne séparée. Plusieurs environnements sont disponibles pour introduire de telles formules.

On dit que LaTeX est en mode mathématique (à l'opposé du mode texte ou LR mode) s'il est à l'intérieur d'un environnement mathématique. Dans ce mode, il ignore les espaces introduits dans le texte. Les espaces sont cependant indispensables pour séparer les éléments, notamment les commandes. De même, les espaces permettent d'aérer le texte d'entrée pour des raisons de clarté. Pour insérer des espaces dans le texte, on consultera la section 23.6.

23.1 Les environnements mathématiques

Pour placer une formule dans un texte, on utilise l'environnement `math`. LaTeX propose deux raccourcis pour cet environnement (`\(...\)` et `$...$`) qui s'utilisent à la place de `\begin{math}...\end{math}`.

Pour créer une formule non numérotée sur une ligne séparée, on utilise l'environnement `displaymath` qui possède deux raccourcis: `\[...\]` et `$$...$$`.

Pour créer une formule numérotée sur une ligne séparée, on utilise l'environnement `equation` qui ne possède pas de raccourci. Nous verrons dans la partie 17 comment faire référence au numéro de la formule.

D'autres environnements mathématiques comme `array` seront vus dans la suite.

Remarque *La mise en forme des formules diffère suivant que la formule se trouve dans un texte ou sur une ligne séparée. Cependant il est possible de forcer le style que l'on désire (par exemple une mise en forme comme dans un texte pour une formule séparée). Le style correspondant à une formule dans un texte est* `\textstyle` *et celui d'une formule centrée* `\displaystyle`*. Il suffit, pour forcer un style, d'entourer la formule (sans la déclaration d'environnement) d'accolades et du nom de la commande au début. (Ex.* `${\displaystyle...}$`*)*

23.2 Structures courantes

Envisageons les différentes structures les plus couramment rencontrées dans les environnements mathématiques.

23.2.1 Indices et exposants

Un indice s'obtient à l'aide de `_` et un exposant à l'aide de `^`.

x_i	`x_i`	x^2	`x^2`
x_i^2	`x_i^2`	x_{n_i}	`x_{n_i}`

23.2.2 Fractions

La commande `\frac{`*num*`}{`*den*`}` produit une fraction de numérateur *num* et de dénominateur *den*.

$\frac{4x+5}{2x-3}$	`\frac{4x+5}{2x-3}`

23.2.3 Racines

La racine carrée s'obtient par `\sqrt{...}` et la racine $n^{\text{ième}}$ par `\sqrt[n]{...}`.

$\sqrt{7x^2+5}$	`\sqrt{7x^2+5}`
$\sqrt[3]{7x^2+5}$	`\sqrt[3]{7x^2+5}`

23.2.4 Points de suspension

LaTeX propose quatre types de points de suspension résumés dans le tableau suivant:

$\ldots$	`\ldots`	$\cdots$	`\cdots`
$\vdots$	`\vdots`	$\ddots$	`\ddots`

La commande `\ldots` peut s'utiliser également en mode texte. La distinction entre `\ldots` et `\cdots` est la hauteur sur la ligne. Par exemple on utilise `\cdots` dans $x_1 + \cdots + x_n$ tandis que l'on utilise `\ldots` dans $x_1, \ldots, x_n$.

23.2.5 Symboles spéciaux

En mathématique, on utilise de nombreux symboles tels que les relations ensemblistes, les sommes, les intégrales, les limites, les lettres grecques, les lettres caligraphiques, les flèches,... Ces symboles sont repris dans les tableaux de la section 31 en fin de document (regroupés pour d'évidentes raisons de recherche).

23.2.6 Symboles à taille variable

Certains symboles, tels que somme et intégrale, changent de taille suivant les conditions. Un tableau contenant ces symboles se trouve également dans la section 31. Ces symboles peuvent posséder des indices et exposants comme pour le signe d'intégration où cela représente les bornes. Par exemple l'intégrale de zéro à l'infini de la fonction f s'obtient par le texte `\int_{0}^{\infty}f(x)\,dx` et produit

$$\int_{0}^{\infty} f(x)\,dx$$

Le but du `\,` est de fournir un petit espace entre la fonction et le `dx`.

23.3 Tableaux mathématiques

Un tableau en mathématique s'obtient par l'environnement `array`. Nous avons vu au point 21.5 son utilisation qui est identique à celle de `tabular`. Dans un `array`, il ne peut y avoir de `\\` sur la dernière ligne sinon cela produit un espace supplémentaire dans le texte et aucun avertissement n'est donné.

23.4 Aligner les formules

Certaines formules sont trop longues et doivent être écrites sur plusieurs lignes. On peut aussi désirer écrire une suite d'équations alignées sur plusieurs lignes... LaTeX propose deux environnements pour réaliser ces formules: `eqnarray` et `eqnarray*`.

L'environnement `eqnarray` peut être vu comme un tableau (centré sur la page) à trois colonnes dont la première et la troisième sont en mode mathématique et celle du milieu en mode texte. Les lignes sont séparées par une `\\` et les colonnes par &. En plus, chaque ligne porte un numéro d'équation. On peut le supprimer en insérant sur la ligne la commande `\nonumber`.

L'environnement `eqnarray*` est identique à `eqnarray` sauf qu'il n'insère pas de numéro.

La commande `\lefteqn{...}` permet de positionner totalement à gauche la première ligne dans un des deux environnements. LaTeX pense que la longueur de la formule est alors 0.

23.5 Placer l'un au dessus de l'autre

Les tableaux peuvent être utilisés pour positionner des éléments les uns au dessus des autres mais ne peuvent résoudre tous les problèmes.

23.5.1 Les accents mathématiques

Dans le texte, les accents sont obtenus à l'aide des commandes du tableau 2 ou par des convertisseurs automatiques . En mathématique, on ne peut les utiliser mais il existe d'autres commandes regroupées dans le tableau 3.

Les accents `\hat` ($\hat{}$) et `\tilde` ($\tilde{}$) possèdent une version qui s'élargit en fonction de son argument. Ces accents sont obtenus par les commandes `\widehat` et `\widetilde`. Ainsi on peut avoir $\widehat{1-x^2}$.

Les lettres i et j possèdent toujours un point. En mathématique, il existe deux commandes produisant les lettres sans les points pour pouvoir par exemple les utiliser avec les accents. On a la commande `\imath` pour un $\imath$ et `\jmath` pour $\jmath$.

23.5.2 Sou– et sur– ligner

La commande de soulignement est `\underline` et celle de surlignement est `\overline`. La commande de soulignement peut aussi être utilisée en mode texte.

23.5.3 Accolades horizontales

Une accolade peut être placée au-dessus ou au-dessous d'éléments par les commandes `\overbrace` et `\underbrace`. Un texte peut être ajouté au-dessus d'une accolade qui se trouve au-dessus d'une formule par `^` et inversement, par `_` pour l'autre cas.

`$\overbrace{a+b+c}$` produit $\overbrace{a+b+c}$

`$\overbrace{a+b+c}^d$` produit $\overbrace{a+b+c}^d$

23.5.4 *stacking symbol*

La commande `\stackrel` place un élément au-dessus d'un autre. Le symbole de définition $\stackrel{\triangle}{=}$ est obtenu par `\stackrel{\triangle}{=}`.

23.6 Espaces en mathématique

Comme dit plus haut, LaTeX ignore complètement les espaces dans les formules mathématiques. Plus exactement, il adapte lui-même les espaces. Dans certains cas, il est utile d'augmenter l'espace et dans d'autre de le diminuer. Pour cela, LaTeX offre quatres commandes:

`\,`	petit espace	`\:`	moyen espace
`\!`	petit espace négatif	`\;`	large espace

Seule `\,` peut aussi être utilisé sans le mode mathématique. Il faut utiliser ces commandes mais il ne faut pas en abuser car LaTeX fonctionne généralement bien. Il faut ajouter un petit espace avant le dx dans une intégrale, juste après une fraction et une racine. Un espace négatif est utilisé pour rapprocher les signes d'intégration d'une intégrale double ainsi qu'après la barre de fraction / d'un quotient.

23.7 Les délimiteurs

Un délimiteur est une barre, une accolade qui se place devant ou derrière un élément en s'ajustant à sa taille. Par exemple, devant un système d'équation, on place une grande accolade. LaTeX considère tous les symboles du tableau 12 comme des délimiteurs et apparaissent avec la même taille. Pour obtenir des délimiteurs qui s'ajustent à la taille des éléments qu'ils délimitent, on utilise les commandes `\left` et `\right` placées juste devant le délimiteur. Ces commandes vont toujours par paire (un gauche et un droit) et s'ajustent à la taille de ce qui se trouve entre les deux. Le délimiteur gauche peut ne pas être le même que celui de droite. Il est possible d'obtenir un délimiteur invisible simplement en plaçant un point (.) à la place du délimiteur.

Par exemple,

$$f(x) = \left\{\begin{array}{ll} 4x^2-3 & \mbox{ si x est positif}\\ x^2-2x-4 & \mbox{ si x est négatif} \end{array}\right.$$

est obtenu par

```
$$f(x) = \left\{\begin{array}{ll}
4x^2-3 & \mbox{ si $x$ est positif}\\
x^2-2x-4 & \mbox{ si $x$ est négatif}
\end{array}\right.$$
```

Frédéric Geraerds 1997

24 Les erreurs

Cette section doit encore être rédigée.

25 Mise en page

Cette section doit encore être rédigée.

26 Dessins

Cette section doit encore être rédigée.

27 Inclure une image

Cette section doit encore être rédigée.

28 Divers

28.1 Espaces et points

LaTeX ajuste les espaces d'une ligne en fonction du texte et de la largeur spécifiée. Cependant, l'espace entre un point terminant une phrase et le début de la phrase suivante est plus long que l'espace séparant deux mots. Pour déterminer si un point termine une phrase ou non, LaTeX utilise une règle très simple: un point (suivit d'un espace) termine une phrase sauf s'il suit une lettre majuscule. En générale, cela fonctionne très bien mais il peut arriver qu'il y ait confusion comme dans l'abréviation "etc." qui constitue une des rares exceptions.

Pour indiquer à LaTeX qu'un point ne termine pas une phrase, il suffit de placer `\␣` directement après le point (sans espace entre le point et le `\`). On peut laisser plusieurs espaces après si on le désire mais aucun avant. Dans le cas inverse, on indique à LaTeX qu'un point qui suit une majuscule termine une phrase par un `\@` juste avant le point.

Si un point précède directement une parenthèse fermante, LaTeX ajoute également un espace supplémentaire après la parenthèse (si elle est suivie d'un espace). Comme pour le point, on indique par un `\` suvit d'un espace directement après la parenthèse que l'espace doit être normal.

La même règle doit également être utilisée pour le point d'interrogation (`?`), pour le point d'exclamation (`!`) et pour les deux points (`:`).

28.2 Points de suspension

Les points de suspension ne peuvent pas être obtenus en plaçant trois points successifs car les espaces entre les points ne sont pas corrects dans ce cas. Pour obtenir "...", on utilise la commande `\ldots` (qui peut s'utiliser en mode texte comme en mode mathématique). On consultera la section 23.2.4 pour plus de détails sur les points de suspension en mode mathématique.

28.3 Crochets et arguments optionnels

La commande `\item` peut prendre un argument optionnel placé entre crochets. Si, comme argument, on désire mettre un texte entre crochets ou si on désire commencer le texte par un crochet, il peut y avoir confusion pour LaTeX. On y remédie comme suit: si l'argument optionnel comporte des crochets, on place l'argument entre des accolades et les accolades entre crochets; si le texte commence par un crochet, on place ce texte entre accolades.

28.4 Les longueurs

Une *longueur* est une mesure de distance. Elle doit sŠexprimer en fonction dŠune unité comme le centimètre (`cm`), le milimètre (`mm`), le pouce (`in`) ou le point (`pt`) qui est une unité traditionnelle en informatique. Ces unités de mesure sont dites fixes car elles ne dépendent pas du contexte dans lequel on les utilise.

Par opposition aux unités fixes, LaTeX connaît également des unités rigides dont la valeur dépend du style du document, de la police utilisée et dŠautres paramètres. On utilise souvent `em` pour la largeur de la lettre "M"et `ex` pour la hauteur de la lettre "x". Il est aisé de comprendre que ces longueurs dépendent de lŠendroit où on les utilise et de la taille de la police courante. Le plus souvent, on utilisera la première pour des espaces horizontaux et la deuxième pour des espaces verticaux.

Il est aussi possible dŠutiliser les commandes de longueur, cŠest-à-dire des commandes dont la valeur est une longueur. Par exemple, `\parindent` repsésente la longueur du retrait de paragraphe. Si on écrit

`2.5\parindent`, cela représente 2.5 fois la longueur précisée par `\parindent`.

Pour définir une commande de longueur, supposons vouloir attribuer une valeur de 5 milimètres à `\long`, on écrit `\long=5mm`.

Remarque *Pour préciser une longueur nulle, on doit absolument préciser une unité de mesure. Donc 0 nŠest pas valide mais bien* `{0mm}, \verb`*0pt,...*

29 Fragile et robuste

Certaines commandes de LaTeX sont dites fragiles et d'autres robustes. Les commandes robustes ne posent aucun problème et peuvent être utilisées dans toutes les circonstances. Par contre, les commandes fragiles doivent faire l'objet d'une plus grande attention car elles peuvent ne pas engendrer le résultat désiré.

On peut considérer que toute commande qui possède un argument optionnel est fragile et que les commandes du mode mathématique sont robuste. Nous avons rarement indiquez dans ce document de quel type étaient les commandes précisées. Nous laissons le soin à l'utilisateur de consulter le livre de référence pour obtenir ce renseignement s'il observe des erreurs bizarres dans la présentation de son texte.

Certains arguments de commandes sont dits *arguments mouvants*. Lorsqu'une commande fragile apparaît dans un tel argument, il est nécessaire de la protéger au moyen de la commande `\protect`. Cette commande s'applique uniquement à la commande qu'elle précède directement. Chaque commande fragile doit posséder sa commande de protection.

Nous laissons le soin à l'utilisateur de consulter le guide pour plus de détail.

30 Sources d'informations

livres

sites internet

Mailing listes

31 Tableaux récapitulatifs

Frédéric Geraerds **1997**

Tableau 2: Les accents

ò	`` \`{o} ``	ó	`\'{o}`	ô	`\^{o}`	ö	`\"{o}`	õ	`\~{o}`
ō	`\={o}`	ȯ	`\.{o}`	ŏ	`\u{o}`	ǒ	`\v{o}`	ő	`\H{o}`
o͡o	`\t{oo}`	o̧	`\c{o}`	ọ	`\d{o}`	o̲	`\b{o}`		

Tableau 3: Les accents mathématiques

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

Tableau 4: Taille des caractères

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

Tableau 5: Lettres grecques

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

Tableau 6: Symboles mathématiques

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

Tableau 7: Symboles de relation

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

La négation des symboles s'obtient en ajoutant `\not` devant l'expression de ceux-ci. Par exemple, $\not<$ et $\not\leq$ s'obtiennent respectivement en tapant `\not<` et `\not\leq`.

Tableau 8: Symboles à taille variable

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

Tableau 9: Flèches

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

Tableau 10: Divers symboles

<table>
<tr><td>$\aleph$</td><td>\aleph</td><td>$\hbar$</td><td>\hbar</td><td>$\imath$</td><td>\imath</td><td>$\jmath$</td><td>\jmath</td><td>ℓ</td><td>\ell</td></tr>
<tr><td>$\wp$</td><td>\wp</td><td>$\Re$</td><td>\Re</td><td>$\Im$</td><td>\Im</td><td>$\mho$</td><td>\mho</td><td>$\prime$</td><td>\prime</td></tr>
<tr><td>$\emptyset$</td><td>\emptyset</td><td>∇</td><td>\nabla</td><td>$\surd$</td><td>\surd</td><td>$\top$</td><td>\top</td><td>$\bot$</td><td>\bat</td></tr>
<tr><td>$\|$</td><td>\|</td><td>$\angle$</td><td>\angle</td><td>$\forall$</td><td>\forall</td><td>$\exists$</td><td>\exists</td><td>$\neg$</td><td>\nag</td></tr>
<tr><td>$\flat$</td><td>\flat</td><td>$\natural$</td><td>\natural</td><td>$\sharp$</td><td>\sharp</td><td>$\backslash$</td><td>\backslash</td><td>∂</td><td>\partial</td></tr>
<tr><td>∞</td><td>\infty</td><td>$\Box$</td><td>\Box</td><td>$\Diamond$</td><td>\Diamond</td><td>$\triangle$</td><td>\triangle</td><td>$\clubsuit$</td><td>\clubsuit</td></tr>
<tr><td>$\diamondsuit$</td><td>\diamondsuit</td><td>$\heartsuit$</td><td>\heartsuit</td><td>$\spadesuit$</td><td>\spadesuit</td><td></td><td></td><td></td><td></td></tr>
</table>

Tableau 11: Fonctions mathématiques

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

Tableau 12: Délimiteurs

<table>
<tr><td>$($</td><td>(</td><td>$)$</td><td>)</td><td>$[$</td><td>[</td><td>$]$</td><td>]</td><td>$\{$</td><td>\{</td></tr>
<tr><td>$\}$</td><td>\}</td><td>$\lfloor$</td><td>\lfloor</td><td>$\rfloor$</td><td>\rfloor</td><td>$\lceil$</td><td>\lceil</td><td>$\langle$</td><td>\langle</td></tr>
<tr><td>$\rangle$</td><td>\rangle</td><td>$/$</td><td>/</td><td>$|$</td><td>|</td><td>$\|$</td><td>\|</td><td>$\uparrow$</td><td>\uparrow</td></tr>
<tr><td>$\downarrow$</td><td>\downarrow</td><td>$\updownarrow$</td><td>\updownarrow</td><td>$\Uparrow$</td><td>\Uparrow</td><td>$\Downarrow$</td><td>\Downarrow</td><td>$\Updownarrow$</td><td>\Updownarrow</td></tr>
</table>