Не надто короткий вступ до $\LaTeX\,2_\varepsilon$

Або $\LaTeX\,2_\varepsilon$ за 130 хвилин

Tobias Oetiker

Hubert Partl, Irene Hyna та Elisabeth Schlegl

Version 4.12, 05 May, 2003

Переклад: **Максим Поляков**

5 травня 2003 р.

Подяка

Джерелом значної частини матеріалу, використаного у цьому документі, є австрійський вступ до LaTeX 2.09 німецькою мовою, авторами якого є:

Hubert Partl `<partl@mail.boku.ac.at>`
Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna `<Irene.Hyna@bmwf.ac.at>`
Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl `<no email>`
in Graz

Якщо ви маєте бажання ознайомитись із німецькою версією документа, ви можете знайти версію, яку Jörg Knappen оновив для LaTeX 2_{ε}, за адресою `CTAN:/tex-archive/info/lshort/german`.

Готуючи цей документ, я отримав багато відгуків від учасників `comp.text.tex`. Перелічені нижче особи допомогли з виправленнями, порадами, та пропозиціями щодо покращень. Вони доклали багато зусиль, щоб допомогти мені довести документ до його теперішнього стану. Я щиро їм усім дякую. Природно, всі помилки, які ви знайдете в цій книзі, — мої. Якщо ви випадково знайдете вірно написане слово, це виключна заслуга котрогось із них.

Rosemary Bailey, Marc Bevand, Friedemann Brauer, Jan Busa, Markus Brühwiler, Pietro Braione, David Carlisle, José Carlos Santos, Mike Chapman, Pierre Chardaire, Christopher Chin, Carl Cerecke, Chris McCormack, Wim van Dam, Jan Dittberner, Michael John Downes, Matthias Dreier, David Dureisseix, Elliot, Hans Ehrbar, Daniel Flipo, David Frey, Hans Fugal, Robin Fairbairns, Jörg Fischer, Erik Frisk, Mic Milic Frederickx, Frank, Andy Goth, Kasper B. Graversen, Arlo Griffiths, Alexandre Guimond, Cyril Goutte, Greg Gamble, Neil Hammond, Rasmus Borup Hansen, Joseph Hilferty, Björn Hvittfeldt, Martien Hulsen, Werner Icking, Jakob, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones, Johannes-Maria Kaltenbach, Michael Koundouros, Andrzej Kawalec, Alain Kessi, Christian Kern, Jörg Knappen, Kjetil Kjernsmo, Maik Lehradt, Rémi Letot, Johan Lundberg, Alexander Mai, Martin Maechler, Aleksandar S Milosevic, Henrik Mitsch, Claus Malten, Kevin Van Maren, Philipp Nagele, Lenimar Nunes de Andrade, Urs Oswald, Demerson Andre Polli, Maksym Polyakov Hubert Partl, John Refling, Mike Ressler, Brian Ripley, Young U. Ryu, Bernd Rosenlecher, Chris Rowley, Risto Saarelma, Hanspeter Schmid, Craig Schlenter, Baron Schwartz, Christopher Sawtell, Geoffrey Swindale, Laszlo Szathmary, Boris Tobotras, Josef Tkadlec, Scott Veirs, Didier Verna, Fabian Wernli, Carl-Gustav Werner, David Woodhouse, Chris York, Fritz Zaucker, Rick Zaccone, and Mikhail Zotov.

Перелічені нижче особи допомогли зауваженнями, виправленнями, порадами щодо українського перекладу.

Тарас Бейко, Дмитро Ковальов, Володимир Лісівка, Анатолій Маляренко, Олександр Мороцький, Дмитро Редчук, Валентин Соломко.

Передмова

LaTeX [2] — система комп'ютерного набору, призначена для створення наукових та математичних документів високої типографської якості. Вона також цілком придатна для створення інших видів документів, починаючи від звичайних листів, і закінчуючи цілими книгами. Механізмом форматування для LaTeX слугує пограма TeX[3].

Цей короткий *Вступ* описує LaTeX 2_ε і повинен бути достатнім для більшості застосувань LaTeX. Повний опис системи LaTeX можна знайти в [2, 4].

Цей *Вступ* розбито на п'ять розділів:

Розділ 1 розповідає про базову структуру документів LaTeX та дає коротке знайомство з його історією. Ви дізнаєтеся також про історію LaTeX. Після читання цього розділу ви повинні мати загальне уявлення про те як працює LaTeX.

Розділ 2 заглиблюється в деталі набору. Він пояснює більшість важливих команд і оточень LaTeX. Після прочитання цього розділу ви зможете створювати перші документи.

Розділ 3 пояснює, як набирати формули в LaTeX. Велика кількість прикладів допоможе вам зрозуміти, як використовувати цю, одну із найважливіших можливостей LaTeX. Наприкінці цього розділу ви знайдете таблиці з переліком більшості відомих у LaTeX математичних символів.

Розділ 4 розповідає про створення предметного покажчика і бібліографії та включення EPS графіки. Він ознайомить вас із створенням PDF документів за допомогою pdfLaTeX, та з деякими іншими корисними розширеннями, наприклад, XY-pic.

Розділ 5 покаже, як користуватися LaTeX для малювання графіки. Замість створення малюнка за допомогою графічного редактора, а потім збереження і включення його в документ, можна описати малюнок таким чином, що LaTeX відобразить його.

Розділ 6 містить потенційно небезпечні відомості про те, як змінювати стандартний макет документа. Необережне використання цих можливостей може призвести до небажаного результату.

Важливо читати розділи послідовно. Зрештою, брошура не така вже й велика. Уважно читайте приклади, які містять велику кількість корисної інформації.

LaTeX працює на більшості комп'ютерів, починаючи з IBM PC чи Mac, і закінчуючи потужними системами UNIX чи VMS. У багатьох університетських мережах система уже встановлена і готова до роботи. Інформація про те, як використовувати локальну установку LaTeX, можна знайти в *Local Guide* [5]. Якщо у вас будуть проблеми з початком роботи, зверніться по допомогу до того, хто надав вам цей буклет. Мета даного *Вступу* полягає *не* у тім, щоб навчити вас встановлювати і настроювати систему LaTeX, а щоб навчити писати документи так, щоб вони могли бути оброблені LaTeX.

Якщо вам буде потрібний будь-який, інший матеріал стосовно LaTeX, пошукайте його у одному з архівів `CTAN`[1]. Домашня сторінка `CTAN` має адресу `http://www.ctan.org`. Усі пакети можна отримати з ftp архіву `ftp://www.ctan.org` та з його дзеркал по усьому світу. Їх можна знайти за адресами `ftp://ctan.tug.org` (США), `ftp://ftp.dante.de` (Німеччина), `ftp://ftp.tex.ac.uk` (Великобританія). Якщо ви в іншій країні, виберіть найближчий до вас архів.

На протязі цієї книги ви знайдете й інші посилання на `CTAN`, особливо стосовно програмного забезпечення та документів, які ви, можливо, захочете звантажити. Замість написання повних URL, я просто писав `CTAN:`, і слідом — відповідну адресу в межах дерева `CTAN`.

Якщо ви бажаєте мати LaTeX на вашому власному комп'ютері, подивіться, що можна знайти за адресою `CTAN:/tex-archive/systems`.
Якщо у вас є думки з приводу того, що варто додати, вилучити, чи змінити в цьому документі, будь ласка, дайте мені знати. Я особливо зацікавлений у відгуках новачків LaTeX стосовно того, які частини *Вступу* легко зрозуміти, і що варто було б пояснити краще.

Tobias Oetiker `<oetiker@ee.ethz.ch>`

Department of Electrical Engineering,
Swiss Federal Institute of Technology

Остання версія цього документа доступна за адресою
`CTAN:/tex-archive/info/lshort`.

[1] Comprehensive TeX Archive Network — Всеохоплююча Мережа Архівів TeX.

Зміст

Перелік ілюстрацій

Перелік таблиць

Розділ 1

Це варто знати

Перша частина цього розділу містить короткий огляд філософії та історії LaTeX. Друга частина зосереджується на основних структурах документів LaTeX 2_{ε}. Після читання цього розділу ви повинні мати загальне уявлення про те, як працює LaTeX. Це потрібно вам для розуміння решти книги.

1.1 The Name of the Game

1.1.1 TeX

TeX — це комп'ютерна програма, створена Дональдом Кнутом (Donald E. Knuth) [3]. Вона призначена для верстки тексту і математичних формул. Кнут почав писати TeX у 1977 році, щоб дослідити потенціал цифрового друкарського обладнання, яке в той час набувало поширення в поліграфічній промисловості. Він сподівався, що зможе переломити тенденцію деградації типографської якості, від чого, на його думку, потерпали його власні книги і статті. TeX, у його теперішньому вигляді, вийшов у світ у 1982 році, з деякими вдосконаленнями, доданими у 1989 році для кращої підтримки 8-бітних символів та багатомовності. TeX відомий надзвичайною стабільністю, здатністю працювати на багатьох комп'ютерних платформах і операційних системах, а також практично повною відсутністю помилок. Номер версії TeX сходиться до числа π і зараз дорівнює 3.14159.

TeX вимовляється «тех». У середовищі `ASCII` TeX пишеться `TeX`.

1.1.2 LaTeX

LaTeX — макропакет, що дозволяє авторам здійснювати набір і друкувати їхні роботи з високою типографською якістю за допомогою професійно розроблених макетів. LaTeX був написаний Леслі Лампортом (Leslie Lamport) [2]. Він використовує TeX як набірника. Зараз LaTeX підтримується Франком Міттельбахом (Frank Mittelbach), LaTeX вимовляється «лейтех» або «латех». Якщо ви посилаєтеся на LaTeX у середовищі `ASCII`, пишіть `LaTeX`. LaTeX 2_ε пишеться `LaTe2e`.

1.2 Основи

1.2.1 Автор, дизайнер і набірник

Для того, щоб опублікуватися, автори віддають свої рукописи до видавництва. Там один з дизайнерів визначає макет документа (ширину стовпчиків, шрифти, інтервали перед та після заголовків, та ін.). Дизайнер записує свої інструкції у рукопис і віддає його набірнику, який здійснює набір книги згідно з цими інструкціями.

Дизайнер–людина намагається зрозуміти, що автор мав на увазі, коли створював рукопис. Покладаючись на свій професійний досвід та виходячи із змісту рукопису, він вирішує, які елементи є заголовками розділів, цитатами, прикладами, формулами, і т. ін.

У середовищі LaTeX, LaTeX бере на себе роль дизайнера книги, використовуючи TeX як набірника. Але LaTeX — це «лише» комп'ютерна програма, і тому потребує чітких інструкцій. Автор повинен надати додаткову інформацію для опису логічної структуру його роботи. Ця інформація записується в текст у вигляді «команд LaTeX».

Це суттєво відрізняється від підходу WYSIWYG[1], прийнятого в більшості сучасних текстових процесорів, таких як *MS Word* чи *Corel WordPerfect*. У цих програмах автори форматують документ інтерактивно одночасно з набором тексту на комп'ютері. У процесі роботи вони можуть бачити на екрані, який вигляд матиме документ, коли його буде надруковано.

Використовуючи LaTeX, як правило, неможливо побачити кінцеву картину під час набору тексту. Відформатований документ, однак, можна проглянути на екрані після обробки файлу LaTeX. При потребі, виправлення можна зробити перед, власне, роздруковкою.

1.2.2 Дизайн макета

Типографський дизайн — це професія. Недосвідчені автори часто припускаються серйозних помилок форматування, гадаючи, що дизайн книги — це здебільшого питання естетики: «якщо документ виглядає художньо, він має гарний дизайн». Але, через те, що документ призначений для читання, а не для вивішування в картинній галереї, зручність його читання і розуміння набагато важливіша, ніж краса. Наприклад:

- Розмір шрифтів і нумерація заголовків повинні вибиратися так,

[1] What you see is what you get — що ви бачите, те й отримаєте.

щоб зробити структуру розділів і підрозділів зрозумілою для читача.

- Рядок повинен бути досить коротким, щоб не напружувати ока читача, і досить довгим для красивого заповнення сторінки.

Користуючись системами WYSIWYG, автори часто створюють естетично привабливі документи зі слабко вираженою або непослідовною структурою. LaTeX дозволяє запобігти таких помилок форматування, примушуючи автора декларувати *логічну* структуру його документа. Потім уже LaTeX вибирає найбільш придатний макет.

1.2.3 Переваги і недоліки

Коли люди зі світу WYSIWYG зустрічаються з користувачами LaTeX, вони часто обговорюють «переваги LaTeX над звичайним текстовим процесором», чи навпаки. Найкраще, що ви можете зробити, коли починається така дискусія, — не висовуватися, тому, що вона часто виходить з-під контролю. Однак, іноді ви не зможете уникнути ...

Ну, то ось вам деякі аргументи для використання в подібних дискусіях. Основні переваги LaTeX над звичайними текстовими процесорами:

- Існують професійно виконані макети, що надають документам вигляду «як з видавництва».
- Зручність набору математичних формул.
- Користувачу потрібно вивчити лише кілька зрозумілих команд, що визначають логічну структуру документа. Йому практично ніколи не потрібно возитися з власне макетом документа.
- Легко створюються навіть складні структури, такі, як примітки, зміст, бібліоґрафія, предметний покажчик, та інше.
- Для вирішення багатьох типографських задач, які не підтримуються прямо базовим LaTeX, є вільно розповсюджувані додаткові пакети. Наприклад, існують пакети для включення PostScript графіки, або для форматування бібліоґрафії у точній відповідності з конкретними стандартами. Багато які з цих додаткових компонентів описані в *The LaTeX Companion* [4].
- LaTeX заохочує авторів писати добре структуровані документи, тому що саме так LaTeX і працює — визначаючи структуру.
- TeX, механізм форматування LaTeX 2_ε, — надзвичайно мобільний і вільно доступний. Тому система працює практично на всіх існуючих платформах.

LaTeX має також і деякі недоліки, але, я гадаю, мені буде важко знайти серед них помітні, хоча, я впевнений, інші вам налічать сотні `;-)`

- LaTeX погано працює у того, хто продав свою душу ...
- Хоча деякі параметри можуть бути налаштовані в межах заздалегідь виготовлених макетів, створити цілком новий макет документа не дуже просто, і це займає багато часу.[2]
- Дуже складно писати неструктуровані і неорганізовані документи.
- Незважаючи на деякі підбадьорливі перші кроки, ваш хом'ячок ніколи не зможе повністю збагнути суть Логічної Розмітки.

1.3 Вхідні файли LaTeX

Вхідними даними для LaTeX є звичайний текстовий файл `ASCII`. Його можна створити в будь-якому текстовому редакторі. Він містить текст документа разом з командами, що вказують LaTeX, як здійснювати набір.

1.3.1 Пробіли

«Порожні» символи, такі, як пробіл або табуляція, LaTeX трактує однаково, як «пробіл». *Кілька послідовних* порожніх символів трактуються як *один* «пробіл». Порожні символи на початку рядка звичайно ігноруються, а одиничний кінець рядка сприймається як «пробіл».

Порожній рядок між двох рядків тексту визначає кінець абзаца. *Кілька* порожніх рядків трактуються так само, як *один* порожній рядок. Нижче наведено приклад. Ліворуч — текст вхідного файлу, праворуч — форматований вивід.

```
Немає значення, вводите ви
один чи                кілька
пробілів між словами.

Порожній рядок починає
новий абзац.
```

Немає значення, вводите ви один чи кілька пробілів між словами.

Порожній рядок починає новий абзац.

1.3.2 Зарезервовані символи

Наступні символи є зарезервованими символами, тобто такими, що або мають у LaTeX спеціальне значення, або наявні не у всіх шрифтах. Якщо ви введете їх у текст прямо, вони звичайно не надрукуються, а змусять LaTeX робити непередбачені речі.

[2]Ходять чутки, що це — одна з основних цілей майбутньої системи LaTeX3.

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

Як ви пізніше побачите, ці символи можна використовувати у ваших документах, додавши попереду «\»:

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

$ % ^ & _ { } ~

Інші символи, і ще багато чого, можна надрукувати з використанням спеціальних команд. Знак «\» *не можна* отримати за допомогою комбінації (\\), оскільки ця команда використовується для розриву рядка[3].

1.3.3 Команди LaTeX

Команди LaTeX чутливі до регістра і мають один із двох наступних форматів:

- Вони починаються символом «\» і продовжуються назвою, що складається тільки з літер. Кінцевою межею назви команди є пробіл, цифра, чи будь-яка інша «не-літера».
- Вони складаються з «\» і рівно однієї «не-літери».

LaTeX ігнорує пробіли після команд. Щоб отримати пробіл після команди, потрібно увести «{}» і пробіл, або спеціальну команду пробілу після назви команди. «{}» не дає LaTeX «з'їсти» всі пробіли після назви команди.

```
Я читав, що Кнут поділяє
людей, що працюють з \TeX{}
на \TeX{}ніків та \TeX пертів.\\
Сьогодні "--- \today
```

Я читав, що Кнут поділяє людей, що працюють з TeX на TeXніків та TeXпертів.
Сьогодні — 5 травня 2003 р.

Деяким командам потрібні параметри, що повинні бути задані між фігурними дужками «{ }» після назви команди. Деякі команди підтримують необов'язкові параметри, що додаються після назви команди в квадратних дужках «[ ]». Наступний приклад показує використання кількох команд LaTeX. Не хвилюйтесь, вони будуть розтлумачені пізніше.

```
Ви можете на мене
\textsl{покластися}!
```

Ви можете на мене *покластися*!

[3] Скористайтеся натомість командою `$\backslash$`. Вона дає «\».

```
Будь ласка, почніть новий
рядок прямо отут!\newline
Дякую!
```

Будь ласка, почніть новий рядок прямо отут!
Дякую!

1.3.4 Коментарі

Коли, обробляючи вхідний файл, LaTeX зустрічає символ %, він ігнорує залишок та кінець поточного рядка, і всі пробіли на початку наступного рядка.

Це корисно для додавання у вхідний файл коментарів, які не повинні виводитися на друк.

Символом % можна також скористатися, щоб розбити довгі рядки в тих місцях, де не дозволяються пробіли або кінець рядка.

```
Це "--- % безглуздий
% Краще: інструктивний <----
приклад: полісульфоніл%
         пиперідиніл%
         метиленгідроксид
```

Це — приклад: полісульфонілпиперідиніл-метиленгідроксид

Для довших коментарів слід скористатися оточенням `comment`. Це оточення визначене в пакеті verbatim, тому для використання `comment` вам необхідно додати команду `\usepackage{verbatim}` до преамбули документа.

```
Це "--- інший
\begin{comment}
можливо, також безглуздий,
але корисний
\end{comment}
приклад вбудовування
коментарів у ваш документ.
```

Це — інший приклад вбудовування коментарів у ваш документ.

Пам’ятайте, це не працюватиме у середині складних оточень, наприклад математичних.

1.4 Структура вхідного файлу

Коли LaTeX 2_ε обробляє вхідний файл, він очікує відповідності певній структурі. Зокрема, кожен документ повинен починатися командою

```
\documentclass{...}
```

Вона вказує, документ якого типу ви збираєтеся писати. Після цього можна включити команди, що впливають на стиль документа в цілому, чи завантажити пакети, що додають системі LaTeX нові якості. Для завантаження такого пакета використовується команда

```
\usepackage{...}
```

Коли все настроювання завершене[4], власне текст починається командою

```
\begin{document}
```

Тепер ви вводите текст із командами LaTeX. Наприкінці документа слід додати команду

```
\end{document}
```

Усе, що йде після цієї команди, LaTeX ігнорує.

Рис. 1.1 показує вміст мінімального файлу для LaTeX 2_ε. Трохи складніший вхідний файл наведено на рис. 1.2[5].

1.5 Типова сесія

Б'юся об заклад, ви не можете дочекатися моменту, коли зможете спробувати роботу LaTeX на прикладі маленького гарненького файлу, зображеного на сторінці 8. Ось кілька порад. LaTeX не має графічного середовища з вишуканими меню та кнопочками. Це просто комп'ютерна програма, що обробляє ваш вхідний файл. Деякі дистрибутиви LaTeX мають графічний інтерфейс, де можна, клацнувши мишкою, примусити LaTeX компілювати вхідний файл. Але Справжні Чоловіки не Клацають Мишкою, тому ось, як умовити LaTeX компілювати ваш вхідний файл у системі з алфавітно-цифровим терміналом. Зважте, цей опис виходить з припущення, що LaTeX уже встановлений на вашому комп'ютері[6]

1. Створіть/відредагуйте ваш вхідний файл LaTeX. Він повинен бути простим текстовим файлом `ASCII`. В Unix усі редактори створюють саме такі файли. Під Windows вам потрібно переконатися, що файл записано саме в форматі ASCII, або *Plain Text*. Вибираючи назву файла, переконайтеся, що він має розширення `.tex`.

---

[4] Область між `\documentclass` і `\begin{document}` називається *преамбулою*.

[5] Для того, щоб документи могли містити українські літери, необхідно підключити пакет українізації LaTeX. Пакет babel є стандартним засобом інтернаціоналізації і підтримує, серед інших, українську мову. (*Прим. пер.*)

[6] Так воно і є на більшості доглянутих Unix систем, і ... Справжні Чоловіки Використовують, так що ... `;-)`

```
\documentclass{article}
\usepackage[ukrainian]{babel}
\begin{document}
Мале є прекрасним.
\end{document}
```

Рис. 1.1: Мінімальний файл LaTeX

```
\documentclass[a4paper,11pt]{article}
\usepackage[T2A]{fontenc}
\usepackage[koi8-u]{inputenc}
\usepackage[ukrainian]{babel}
% Визначаємо заголовок
\author{Г.~Партл}
\title{Мінімалізм}
\begin{document}
% Ґенеруємо заголовок
\maketitle
% Створюємо зміст
\tableofcontents
\section{\cyr{}Деякі цікаві слова}
Ось тут і починається моя чудова стаття.
\section{\cyr{}Прощавай, світе}
\ldots{} а отут вона закінчується.
\end{document}
```

Рис. 1.2: Приклад реалістичної журнальної статті

2. Запустіть LaTeX з вашим вхідним файлом У випадку успішного завершення ви отримаєте `.dvi` файл. Вам можливо доведеться запістити LaTeX кілька разів для того, щоб отримати прасильні перехресні посилання. Якщо у вхідному файлі є помилки, LaTeX повідомить про це і припинить його обробку. Натисніть `ctrl-D` щоб повернутися до командного рядка.

   ```
   latex foo.tex
   ```

3. Тепер його можна продивитися Існує кілька способів це зробити. Можна відобразити файл на екрані за допомогою

   ```
   xdvi foo.dvi
   ```

   Це працює тільки в Unix з X11. Якщо ви під Windows, можете спробувати `yap` (інша програма для перегляду).

   Ви можете конвертувати `.dvi` файл в PostScript для друку і перегляду за допомогою GhostScript.

   ```
   dvips -Pcmz foo.dvi -o foo.ps
   ```

   Якщо вам пощастило мати систему LaTeX, укомплектовану `dvipdf`, ви можете конвертувати `.dvi` файл прямо в `.pdf` формат.

   ```
   dvipdf foo.dvi
   ```

1.6 Макет документа

1.6.1 Класи документів

Перше, що LaTeX повинен знати, обробляючи вхідний файл, це тип створюваного автором документа. Він задається командою

`\documentclass[`*опції*`]{`*клас*`}`

Тут *клас* визначає тип створюваного документа. Таблиця 1.1 перелічує класи документів, розглянуті у цьому *Вступі*. До складу LaTeX 2_ε входять додаткові класи для інших документів, у тому числі листи і слайди. Параметр *опції* змінює поведінку класу документа. Опції повинні бути розділені комами. Найбільш часто вживані опції стандартних класів документів перелічені у таблиці 1.2.

Приклад: вхідний файл для документа LaTeX може починатися рядком

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

Ця команда змушує LaTeX набирати документ як *статтю*, з базовим розміром шрифту *одинадцять пунктів* і форматувати документ для *двостороннього* друку на аркушах *формату A4*.

1.6.2 Пакети

У процесі створення деяких документів, вам, ймовірно, зустрінуться проблеми, які не можна вирішити за допомогою базового LaTeX. Якщо ви захочете включити у документ графіку, кольоровий текст чи вхідний код програми з зовнішнього файлу, можливості LaTeX потрібно буде розширити. Такі розширення називаються пакетами. Пакети активізуються командою

\usepackage[*опції*]{*пакет*}

Де *пакет* — це назва пакета, а *опції* — список ключових слів, що активізують спеціальні властивості пакета. Деякі пакети включені в основний дистрибутив LaTeX 2_ε (див. таблицю 1.3), інші розповсюджуються окремо. Додаткова інформація про встановлені у вас пакети може міститися в *Local Guide* [5]. Основне джерело інформації про пакети LaTeX — це *The LaTeX Companion* [4]. Ця книга містить описи сотень пакетів разом з інформацією про те, як писати ваші власні розширення для LaTeX 2_ε.

Табл. 1.1: Класи документів

`article` для статей у наукових журналах, презентацій, коротких звітів, програмної документації, запрошень...

`report` для довших звітів з кількома розділами, невеликих книжок, дисертацій...

`book` для справжніх книг

`slides` для слайдів. Використовує великі літери без зарубок. Замість нього можна використовувати FoilTeX[a].

[a] CTAN:/tex-archive/macros/latex/contrib/supported/foiltex

Табл. 1.2: Опції класів документів

`10pt`, `11pt`, `12pt` Встановлює розмір основного шрифту документа. Якщо жодна з цих опцій не зазначена, використовується `10pt`.

`a4paper`, `letterpaper`... Визначає розмір аркуша. Якщо не вказано, використовується `letterpaper`. Аналогічно можуть бути задані `a5paper`, `b5paper`, `executivepaper` і `legalpaper`.

`fleqn` Формули будуть не відцентровані, а вирівняні ліворуч.

`leqno` Формули нумеруються не праворуч, а ліворуч.

`titlepage`, `notitlepage` Вказують, розміщувати заголовок на окремому (титульному) аркуші, чи ні. Стандартно класи `report` і `book` формують титульну сторінку, починаючи наступний матеріал з нової, а `article` — ні.

`oneocolum`, `ntwocolumn` Форматування документа в один стовпчик чи у два стовпчики.

`twoside, oneside` Форматує сторінки для друку на двох, чи на одній стороні аркуша. Якщо не вказано, `article` і `report` використовують односторонній вивід, а клас `book` — двосторонній вивід. Зважте, опція `twoside` впливає на формат сторінок, а *не* вказує принтеру друкувати з двох сторін.

`openright, openany` Розділи починаються тільки з правої (непарної), чи з будь-якої сторінки. Для класу `article` ці опції не мають сенсу, оскільки там немає розділів. Клас `report` стандартно починає розділи з наступної сторінки, а клас `book` — з непарної.

Табл. 1.3: Деякі з пакетів, що розповсюджуються з LaTeX

doc	Дозволяє документувати програми на LaTeX. Описаний у `doc.dtx` [a] і в *The LaTeX Companion* [4].
exscale	Надає масштабовані версії додаткових математичних шрифтів. Описаний у `ltexscale.dtx`.
fontenc	Вказує кодування шрифту LaTeX. Описаний у `ltoutenc.dtx`.
ifthen	Надає команди виду «якщо ..., то виконувати ..., інакше виконувати ...». Описаний у `ifthen.dtx` і в *The LaTeX Companion* [4].
latexsym	Щоб підключити шрифт спеціальних символів LaTeX, потрібно використовувати пакет `latexsym`. Описаний у `latexsym.dtx` і в *The LaTeX Companion* [4].
makeidx	Надає команди для створення покажчиків. Описаний у розділі 4.3 і в *The LaTeX Companion* [4].
syntonly	Обробляє документ, не здійснюючи власне набору, що зручно для швидкої перевірки на помилки. Описаний у `syntonly.dtx` і в *The LaTeX Companion* [4].
inputenc	Дозволяє вказати вхідне кодування, наприклад, ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM code pages, Apple Macintosh, Next, ANSI-Windows чи визначене користувачем. Описаний у `inputenc.dtx`.

[a] Цей файл повинен бути встановлений у вашій системі, і ви можете отримати `dvi` файл, набравши `latex doc.dtx` у будь-якому каталозі, де ви маєте право на запис. Те саме стосується усіх інших файлів, згаданих у цій таблиці.

1.6.3 Стилі сторінки

LaTeX підтримує три стандартні комбінації верхнього та нижнього колонтитулів — так звані стилі сторінки. Параметр *стиль* команди

`\pagestyle{`*стиль*`}`

визначає, який із них використовувати. Стандартні стилі сторінки перелічені в таблиці 1.4.

Табл. 1.4: Стандартні стилі сторінки LaTeX

`plain` друкує номери в середині нижнього колонтитула. Цей стиль установлений за замовчуванням.

`headings` друкує назву поточного розділу і номер сторінки у верхньому колонтитулі, а нижній залишається порожнім. (Цей стиль використано у даному документі.)

`empty` робить верхні та нижні колонтитули порожніми. (Цей стиль автоматично застосовується до першої сторінки кожного розділу.)

Стиль поточної сторінки можна змінити командою

`\thispagestyle{`*стиль*`}`

Опис того, як створювати ваші власні стилі колонтитулів, дивіться в *The LaTeX Companion* [4], а також у розділі 4.4 на сторінці 67.

1.7 Файли, які вам можуть трапитися

Працюючи з LaTeX, ви скоро опинитеся в лабіринті файлів з різними розширеннями, і, можливо, без поняття. Нижче наведено список типів файлів, які ви можете зустріти, працюючи з LaTeX. Ця таблиця не претендує на виключну повноту, але якщо котресь із розширень, важливих, на вашу думку, тут відсутнє, дайте, будь ласка, знати.

`.tex` Вхідний файл LaTeX або TeX. Може бути компільований за допомогою `latex`.

`.sty` LaTeX макро пакет. Такий файл можна підключити до вашого LaTeX документа за допомогою команди `\usepackage`.

`.dtx` Документований TeX. Це основний формат розповсюдження стильових файлів LaTeX. Якщо обробити `.dtx` файл, то буде створено документований код макро пакета LaTeX, що міститься у файлі `.dtx`.

`.ins` Установник для файлів, що містяться у відповідному файлі `.dtx`. Завантаживши пакет LaTeX з мережі, ви звичайно отримаєте файли `.dtx` та `.ins`. Для того, щоб розпакувати `.dtx` файл, виконайте LaTeX з файлом `.ins`.

`.cls` Файли класів, які визначають вигляд вашого документа. Потрібний клас можна вибрати командою `\documentclass`.

`.fd` Файл опису шрифтів, інформує LaTeX про нові шрифти.

Під час обробки вашого документа LaTeX створює наступні файли:

`.dvi` Device Independent file (пристрій-незалежний файл). Це — головний результат компілювання LaTeX. ви можете продивитися його зміст програмою перегляду DVI, або надіслати на друк за допомогою `dvips` чи аналогічної програми.

`.log` Дає детальний звіт про те, що трапилося під час останнього проходу LaTeX.

`.toc` Містить назви розділів та інших структурних елементів. Він читається під час наступного проходу LaTeX, і використовується для побудови змісту.

`.lof` Те ж саме, що `.toc`, але для списку ілюстрацій.

`.lot` Те саме для списку таблиць.

`.aux` Ще один файл, що передає інформацію від одного проходу до наступного. Серед інших речей, `.aux` файл використовується для зберігання інформації, пов'язаної з перехресними посиланнями.

`.idx` Якщо ваш документ містить покажчик, у цьому файлі LaTeX тримає слова, які потрібно туди занести. Обробіть цей файл програмою `makeindex`. Дивіться розділ 4.3 на сторінці 66 для більш детальної інформації про створення покажчиків.

`.ind` Оброблений `.idx` файл, готовий для включення у ваш документ у наступному циклі компілювання.

`.ilg` Звіт про роботу `makeindex`.

1.8 Великі проекти

При роботі з великими документами буває зручно розділити вхідний файл на кілька частин. LaTeX має дві команди, які допоможуть це зробити. Команду

`\include{`*файл*`}`

можна використовувати в тілі документа, щоб включити у нього вміст файлу з назвою *файл.tex*. Пам'ятайте, що LaTeX почне нову сторінку, перш, ніж обробляти матеріал з *файл.tex*.

Для вибіркового читання файлів, включених командами `\include`, можна скористатися командою

`\includeonly{`*файл*`,`*файл*`,...}`

Після виконання в преамбулі документа цієї команди LaTeX буде включати тільки файли, перераховані в її аргументі. Зверніть увагу, між назвами файлів і комами не повинно бути пробілів.

Команда `\include` починає набір включеного тексту з нової сторінки. Це зручно при використанні `\includeonly`, тому що, навіть коли деякі включені файли пропущені, межі сторінок не будуть зміщуватися, і заголовки структурних елементів пропущених файлів залишаться у змісті. Проте іноді це небажано, і тоді ви можете використати команду

`\input{`*файл.tex*`}`

Вона просто включає вміст зазначеного файлу. Зверніть увагу, що у цьому випадку потрібно вказувати розширення файла.

Для того, щоб LaTeX швидко перевірив ваш документ, можна скористатися пакетом syntonly. Це примусить LaTeX продивитися документ, перевіряючи тільки синтаксис і правильність використання команд, без створення вихідного `.dvi` файла. У цьому режимі LaTeX працює набагато швидше і ви можете заощадити цінний час. Використання дуже просте:

```
\usepackage{syntonly}
\syntaxonly
```

Коли ви захочете отримати вивід, просто закоментуйте другий рядок (додавши знак процента).

Розділ 2

Набір тексту

Після прочитання попереднього розділу ви повинні мати уявлення, з чого складається документ LaTeX. У цьому розділі ви отримаєте знання, необхідні для створення реальних документів.

2.1 Структура тексту і мови

By Hanspeter Schmid <hanspi@schmid-werren.ch>

Основна мета написання тексту (за виключенням деякої сучасної UVA[1] літератури), — передати читачу ідеї, інформацію, чи знання. Читач краще зрозуміє текст, якщо ці ідеї структуровано, і набагато краще побачить і відчує цю структуру, якщо типографська форма відбиває логічну і семантичну структуру документа.

LaTeX відрізняється від інших систем комп'ютерного набору тим, що вам потрібно лише задавати йому логічну і семантичну структуру тексту. Він потім вибирає типографську форму відповідно до «правил», заданих у файлі класу документа та в різних стильових файлах.

Найважливіший елемент тексту в LaTeX (і в типографії взагалі) — абзац. Він називається «одиницею тексту», тому що абзац є тією типографською формою, яка повинна відображувати одну зв'язну думку або ідею. Далі ви довідаєтеся, як можна отримати розрив рядка, наприклад, за допомогою \\, або початок абзаца, наприклад, залишивши порожній рядок. Тому, якщо починається нова думка, повинен починатися новий абзац, а, якщо ні, — використовуйте розрив рядка. Якщо ви маєте сумнів щодо поділу на абзаци, подумайте про ваш текст, як про носія ідей і думок. Якщо ви почали новий абзац, продовжуючи стару думку, — заберіть початок нового абзаца. Якщо протягом абзаца починається зовсім нова думка — розбийте його на два.

Більшість людей недооцінюють важливість належної організації абзаців. Багато хто навіть не розуміє значення абзацних розривів, і, особливо

[1] Швейцарською німецькою «не схожий будь-якою ціною» (Um's Verrecken Anders).

в LaTeX, починає новий абзац, навіть не усвідомлюючи цього. Останню помилку особливо легко зробити, коли в тексті є формули. Подивіться на наступні приклади і спробуйте зрозуміти, чому іноді в них використовуються порожні рядки (розриви абзаців), а іноді — ні. (Якщо ви ще не знаєте усіх команд на рівні, достатньому для розуміння прикладів, прочитайте, будь ласка, цей і наступний розділи, і потім поверніться до прикладів знову.)

```
% Приклад 1
\ldots коли Ейнштейн вивів свою формулу
\begin{equation}
  e = m \cdot c^2 \; ,
\end{equation}
яка є одночасно найширше відомою
і найменш зрозумілою фізичною формулою.

% Приклад 2
\ldots з чого випливає закон струму Кірхгофа:
\begin{equation}
  \sum_{k=1}{n} I_k = 0 \; .
\end{equation}

Закон струму Кірхгофа можна вивести \ldots

% Приклад 3
\ldots який має кілька переваг.

\begin{equation}
  I_D = I_F - I_R
\end{equation}
є ядром зовсім іншої моделі транзистора.
\ldots
```

Наступна, менша, одиниця тексту — речення. У англійському тексті після крапки, що завершує речення, ставиться довший пробіл, ніж після крапки, яка стоїть після скорочення. LaTeX намагається визначити, яку з них ви мали на увазі. Якщо він помиляється, ви повинні йому підказати. Це пояснено далі у цьому розділі.

Належна організація тексту стосується і окремих частини речень. Більшість мов використовує дуже складні правила пунктуації, але в деяких (включаючи німецьку і англійську) ви розставите майже усі коми на місця, просто пам'ятаючи, що вони значать: коротку зупинку в потоці викладу. Якщо ви не впевнені у своїх ко́мах, прочитайте речення вголос, роблячи коротку паузу на кожній комі. Якщо десь це пролунає дивно, заберіть цю кому, якщо ви відчуєте необхідність зупинки в іншому місці, поставте кому там.

Нарешті, абзаци тексту повинні бути також логічно структуровані на більш високому рівні поєднанням у розділи, підрозділи, і так далі. Однак, типографський ефект написання, наприклад, `\section{Структура тексту і мови}`, настільки очевидний, що повинно бути одразу зрозуміло (для тих, хто знає англійську мову:-), як використовувати ці високорівневі структури.

2.2 Поділ на рядки і сторінки

2.2.1 Вирівнювання абзаців

Набір книг часто здійснюється так, щоб усі рядки мали однакову довжину. LaTeX вставляє необхідні розриви рядків і пробіли між словами, оптимізуючи форматування усього абзацу. При необхідності він також переносить слова, які не можна розмістити в рядку. Форматування абзаців залежить від класу документа. Звичайно кожен абзац починається з відступу, а між двома послідовними абзацами не робиться додаткового інтервалу. Подробиці дивіться у розділі 6.3.2.

У деяких випадках може бути необхідно вказати LaTeX, що потрібно розірвати рядок у певному місці:

`\\` чи `\newline`

починають новий рядок, не починаючи нового абзацу.

`\\*`

забороняє, крім того, розрив сторінки після вставленого розриву рядка.

`\newpage`

починає нову сторінку.

Команди

`\linebreak[`*n*`]`, `\nolinebreak[`*n*`]`, `\pagebreak[`*n*`]` і `\nopagebreak[`*n*`]`

призведуть, відповідно, до розриву рядка, заборони розриву рядка, розриву сторінки і заборони розриву сторінки. Необов'язковий аргумент n дозволяє автору впливати на їхні дії. Він може приймати значення від нуля до чотирьох. Значення n менше від 4 залишає LaTeX можливість ігнорувати вашу команду, якщо результат виглядав би дуже погано. Не плутайте команди «розриву» (break) з командами «початку» (new). Навіть якщо ви задасте команду «розриву», LaTeX спробує вирівняти правий край сторінки і загальну висоту сторінки, як описано в наступному підрозділі (секції). Якщо ви дійсно хочете почати «новий рядок» (new line), використовуйте відповідну команду. Вгадайте, як вона називається!

LaTeX завжди намагається оптимальним чином розбити абзац на рядки. Якщо він не може знайти спосіб розбити рядки у відповідності зі своїми стандартами, він дозволяє одному рядку виступати з абзацу вправо. Тоді LaTeX скаржиться на «overfull hbox». Найчастіше це трапляється, коли він не може знайти місця для переносу слова[2]. Можна примусити LaTeX трохи послабити свої стандарти, давши команду `\sloppy`. Це дозволить запобігти появі рядків, що виступають, збільшуючи інтервали між словами, навіть якщо кінцевий результат буде далеким від оптимального. У цьому випадку користувач отримає попередження про «underfull hbox». В багатьох випадках абзац матиме не дуже гарний вигляд. Команда `\fussy` дозволить повернути початкові високі стандарти поділу рядків.

2.2.2 Переноси

У разі потреби LaTeX розбиває слова для переносу на наступний рядок. Якщо алгоритм переносів не знаходить правильних місць переносу, ви можете виправити положення, повідомивши TeX про винятки за допомогою наступних команд.

Якщо задати команду

`\hyphenation{`*список слів*`}`

то слова, перераховані у її аргументі, будуть переноситися тільки у місцях, позначених «-». Ця команда повинна містити тільки слова, що складаються зі звичайних літер або, радше, знаків, які LaTeX вважає звичайними літерами. Винятки переносів зберігаються для мови, котра є активною в момент виконання команди `\hyphenation`. Це значить, що якщо ви використаєте цю команду в преамбулі документа, вона впливатиме на правила переносу для англійської мови. Якщо вона буде вжита після `\begin{document}` і ви використовуєте пакет для підтримки національних мов, наприклад, babel, винятки переносів стосуватимуться мови, активованої за допомогою babel.

Наступний приклад дозволяє переноси в слові «hyphenation», так само, як і в слові «Hyphenation», і забороняє переноси в словах «FORTRAN», «Fortran» і «fortran»:

[2]Хоча LaTeX і дає попередження, такі рядки не завжди легко знайти. Якщо в команді `\documentclass` використати опцію `draft`, такі рядки будуть відзначені жирною чорною рискою на правому полі.

```
\hyphenation{FORTRAN Hy-phen-a-tion}
```

Команда `\-` вставляє в слово можливі точки переносу. Ця команда особливо корисна для слів, що містять спеціальні символи (наприклад, символи з діакритичними акцентами), тому що LaTeX не переносить такі слова автоматично.

```
I think this is: su\-per\-cal\-%
i\-frag\-i\-lis\-tic\-ex\-pi\-%
al\-i\-do\-cious
```

I think this is: supercalifragilisticexpialido-cious

Кілька слів можна втримати разом на одному рядку командою

`\mbox{`*текст*`}`

Аргумент команди не буде перенесено за будь-яких обставин.

```
Номер мого телефону незабаром
зміниться на \mbox{0116 291 2319}.

Параметр
\mbox{\emph{назва файлу}} повинен
містити назву файлу.
```

Номер мого телефону незабаром зміниться на 0116 291 2319.

Параметр *назва файлу* повинен містити назву файлу.

Команда `\fbox` аналогічна до `\mbox`, але, на додачу, навколо вмісту буде зображено прямокутник.

2.3 Ready made Strings

В деяких прикладах на попередніх сторінках ви бачили кілька простих команд LaTeX для набору спеціальних рядків тексту:

Команда	Приклад	Опис
`\today`	5 травня 2003 р.	Поточна дата поточною мовою
`\TeX`	TeX	Ім'я улюбленого набірника
`\LaTeX`	LaTeX	The Tame of the Game
`\LaTeXe`	LaTeX 2_ε	Сучасна реінкарнація LaTeX

2.4 Спеціальні літери і символи

2.4.1 Знаки лапок

Для набору лапок *ніколи не використовуйте* знак `"`, як на друкарській машинці. В типографії існують спеціальні знаки відкриваючої і закриваючої лапок. У LaTeX використовують два знаки ` для відкриваючих лапок і два знаки ' (апострофи) для закриваючих[3].

```
``Будь ласка, натисніть
клавішу `x' ''
```

“Будь ласка, натисніть клавішу ‘x’ ”

2.4.2 Тире і дефіси

LaTeX знає про чотири види тире. Три з них ви можете одержати різною кількістю послідовних знаків -. Четверте насправді зовсім не тире, а математичний знак мінус:

```
X-промені\\
сторінки 13--67\\
{\Eng yes---or no?}\\
$0$, $1$ та $-1$
```

X-промені
сторінки 13–67
yes—or no?
0, 1 та -1

Ці тире називаються так: «-» дефіс, «–» коротке тире, «—» довге англійське тире[4] , і «$-$» знак мінуса.

2.4.3 Тильда (~)

Символ, котрий часто можна побачити як частину веб-адреси, називається тильдою. В LaTeX його можна отримати за допомогою \~, але результат: ˜ може бути це не зовсім тим, чого б ви хотіли. Спробуйте це:

```
http://www.rich.edu/\~{}bush \\
http://www.clever.edu/$\sim$demo
```

http://www.rich.edu/˜bush
http://www.clever.edu/$\sim$demo

2.4.4 Символ градуса (∘)

Як надрукувати символ градуса в LaTeX?

```
Уже $-30\,^{\circ}\mathrm{C}$,
скоро настане
надпровідність.
```

Уже $-30\,^{\circ}\mathrm{C}$, скоро настане надпровідність.

[3]В українській мові звичайно замість “таких” і ‘таких’ лапок використовують «такі» і „такі“. Вони задаються командами \flqq, \frqq (замість цієї пари часто використовують лігатури << і >>), і \glqq, \grqq. Див. також розділ 2.5.1. (*Прим. пер.*)

[4]В українській мові використовується довге кириличне тире « — », яке трошки коротше ніж англійське, і обмежується пробілами. Дивіться розділ 2.5.1 і таблицю 2.3. (*Прим. пер.*)

2.4.5 Символ Євро (€)

Пишучи про гроші, у наші дні не обійтися без символа євро. Він є у багатьох сучасних шрифтах. Завантаживши пакета textcomp у преамбулі документа

```
\usepackage{textcomp}
```

для отримання символа євро потрібно скористатися командою

```
\texteuro
```

Якщо у вашому шрифті немає власного символа євро, або якщо він вам не подобається, у вас є кілька альтернатив.

Можна скористатися пакетом eurosym, який надає офіційний символ євро:

```
\usepackage[official]{eurosym}
```

Якщо вам більш подобається символ євро, що відповідає вашому шрифту, скористайтеся опцією gen замість `official`.

Якщо на вашій системі встановлені шрифти Adobe Eurofonts (вони вільно доступні за адресою `ftp://ftp.adobe.com/pub/adobe/type/win/all`) ви можете скористатися пакетом europs і командою `\EUR` (для отримання символа євро, що відповідає поточному шрифту), або пакетом eurosans і командою `\euro` (для «офіційного євро»).

Пакет marvosym надає також багато різних символів, у тому числі євро (за допомогою команди `\EUR`). Його недоліком є те, що він не надає нахиленого і грубого варіантів символа євро.

Табл. 2.1: Повний мішок символів євро

Пакет	Команда	Прямий	Без зарубок	Машинопис
eurosym	`\euro`	€	€	€
[gen]eurosym	`\euro`	€	€	€
europs	`\EUR`	€	€	€
eurosans	`\euro`	E	E	E
marvosym	`\EUR`	€	€	€

2.4.6 Трикрапка (...)

На друкарській машинці крапка чи кома займають стільки ж місця, як і будь-яка інша літера. Якщо ці символи надруковані в книзі, то вони займають дуже мало місця і набираються дуже близько до попередньої літери. Ви не зможете отримати трикрапку, просто надрукувавши три крапки, тому що відстані між ними будуть неправильним. Тому для трикрапки є спеціальна команда, що називається

`\ldots`

```
Не так ..., а отак:\\
Нью-Йорк, Токіо, Будапешт, \ldots
```

Не так ..., а отак:
Нью-Йорк, Токіо, Будапешт, ...

2.4.7 Лігатури

Деякі комбінації літер виводяться на друк не просто як послідовності окремих літер, а з використанням спеціальних символів.

ﬀ ﬁ ﬂ ﬃ... замість ff fi fl ffi ...

Ці так звані лігатури можуть бути заборонені вставкою `\mbox{}` між двома відповідними літерами. Це може бути необхідно для слів, утворених із двох частин.

```
\Large Не ‘‘shelfful’’,\\
а ‘‘shelf\mbox{}ful’’
```

Не “shelﬀul”,
а “shelfful”

2.4.8 Акценти і спеціальні символи

LaTeX підтримує використання акцентів і спеціальних символів багатьох мов. Таблиця 2.2 показує всілякі акценти у застосуванні до літери o. Зрозуміло, що на її місці можуть бути й інші літери.

Щоб помістити знак акценту над літерами і чи j, крапки над ними повинні бути вилучені. Це досягається набором `\i` та `\j`.

```
H\^otel, na\"\i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\~norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße

Табл. 2.2: Акценти і спеціальні символи

ò	`\`o`	ó	`\'o`	ô	`\^o`	õ	`\~o`
ō	`\=o`	ȯ	`\.o`	ö	`\"o`	ç	`\c c`
ŏ	`\u o`	ǒ	`\v o`	ő	`\H o`	ǫ	`\c o`
ọ	`\d o`	o̲	`\b o`	o͡o	`\t oo`		
œ	`\oe`	Œ	`\OE`	æ	`\ae`	Æ	`\AE`
å	`\aa`	Å	`\AA`				
ø	`\o`	Ø	`\O`	ł	`\l`	Ł	`\L`
ı	`\i`	ȷ	`\j`	¡	`!``	¿	`?``

2.5 Підтримка багатомовності

Якщо вам потрібно створювати документи мовами, відмінними від англійської, LaTeX повинен бути відповідним чином сконфіґурованим:

1. Усі ґенеровані автоматично текстові послідовності[5] повинні бути перекладені потрібною мовою. Для багатьох мов ці зміни досягаються використанням пакета babel (автор Johannes Braams).

2. LaTeX повинен знати правила переносу для нової мови. Підключення правил переносу до LaTeX є більш складним. Воно включає перебудування форматного файлу з шаблонами переносів потрібних мов. Ваш *Local Guide* [5] повинен містити більше інформації про це.

3. Типографські правила, специфічні для мови. Наприклад, у французькій мові обов'язковим є пробіл перед кожною двокрапкою (:).

Якщо ваша система уже відповідним чином сконфіґурована, ви можете активізувати пакет babel додаванням команди

`\usepackage[`*мова*`]{babel}`

після команди `\documentclass`. Список *мов*, вбудованих у вашу LaTeX систему, буде відображено кожного разу, коли компілятор починає роботу. `Babel` автоматично активізує відповідні правила переносу для обраної вами мови. Якщо форматний файл вашого LaTeX не підтримує переносів для обраної мови, `babel` буде працювати, але заборонить переноси, що негативно позначиться на зовнішньому вигляді документа.

[5] Зміст, Список ілюстрацій, Бібліоґрафія ...

Для деяких мов babel уводить нові команди, що спрощують набір спеціальних символів. Наприклад, німецька мова містить безліч діакритичних знаків (äöü). З використанням babel можна вводити ö, друкуючи `"o` замість `\"o`.

Якщо ви використовуєте babel з кількома мовами

```
\usepackage[моваA,моваB]{babel}
```

остання з *мов* стає основною мовою документа, а команду

```
\selectlanguage{моваA}
```

можна використовувати для встановлення поточної мови всередині документа.

Більшість сучасних комп'ютерних систем дозволяє уводити символи національних алфавітів прямо з клавіатури. Пакет inputenc призначений для того, щоб LaTeX міг обробляти різноманіття кодувань, використовуваних різними мовами і на різних комп'ютерних платформах:

```
\usepackage[вхідне_кодування]{inputenc}
```

При використанні цього пакета ви повинні розуміти, що інші можуть неправильно бачити ваші вхідні файли на своєму комп'ютері через використання іншого кодування. Наприклад, німецька літера ä на IBM PC кодується як 132, на Unix системах, що використовують ISO-LATIN 1, — як 228, а в кодовій таблиці cp1251, використовуваній для кирилиці в MS Windows, такої літери взагалі немає. Тому використовуйте цю можливість з обережністю. В залежності від системи, на якій ви працюєте, вам може знадобитися одне з наступних кодувань[6]:

Операційна система	кодування: латиниця	кодування: кирилиця
Mac	`applemac`	`macukr`
Unix	`latin1`	`koi8-ru`
Windows	`ansinew`	`cp1251`
DOS, OS/2	`cp850`	`cp866nav`

Працюючи з багатомовними документами, що потребують несумісних вхідних кодувань, ви можливо захочете перейти на unicode за до-

[6] Про підтримувані в LaTeX вхідні кодування для латиниці та кирилиці можна дізнатися з `inputenc.dtx` та `cyinpenc.dtx` відповідно. Див. розділ 4.6, щоб дізнатися, як отримати документацію пакетів.

помогою пакета ucs.

```
\usepackage{ucs}
\usepackage[utf8]{inputenc}
```

дозволить створювати вхідні файли LaTeX в `utf8`, кодування, в якому літери закодовані за допомогою послідовностей від одного до чотирьох байтів.

Інша річ — кодування шрифту. Воно визначає, у якій позиції в межах шрифту TeX знаходиться кожна літера. Різні вхідні кодування можуть бути відображені в одне кодування шрифту, зменшуючи таким чином кількість необхідних наборів шрифтів. Кодування шрифтів LaTeX обробляє за допомогою пакета fontenc:

```
\usepackage[кодування]{fontenc}
```

де *кодування* — кодування шрифту, або вихідне кодування. Можливо завантажити кілька кодувань одночасно.

Якщо не задати вихідне кодування явно, LaTeX використовує `OT1` — кодування оригінальниого Computer Modern шрифту TeX, що містить тільки 128 символів з 7-бітного набору символів ASCII. Коли потрібні акцентовані літери, TeX створює їх, комбінуючи звичайну літеру з акцентом. Незважаючи на те, що результат виглядає прекрасно, цей підхід не дає автоматичним переносам працювати в словах, що використовують акцентовані літери. Крім того, деякі латинські літери не можна створити за допомогою звичайних літер і акцентів, не кажучи вже про літери не-латинських алфавітів, наприклад грецького чи кирилиці.

Для виправлення цих недоліків було створено ряд 8-бітних CM-подібних наборів шрифтів. EC (*Extended Cork*) шрифти у `T1` кодуванні містять літери і розділові знаки більшості європейських мов, що використовують латиницю. Набір шрифтів LH містить літери, необхідні для створення документів мовами, що використовують кирилицю. Через велику кількість літер вони згруповані в чотири кодування — `T2A`, `T2B`, та `T2C`, які містять кирилицю і базову латиницю, а також `X2`, яке містить тільки кирилицю[7]. Набір CB містить шрифти у кодуванні `LGR`, необхідні для створення документів грецькою мовою.

Використання цих шрифтів дозволяє поліпшити переноси в документах мовами, що використовують латиницю, та, власне, створення документів іншими мовами. Іншою перевагою є те, що вони надають шрифти родини CM в усіх комбінаціях типу, ваги, форми, та оптично масштабованих розмірів.

[7] Список мов, підтримуваних кожним з кодувань, можна знайти в [14].

2.5.1 Українська мова[8]

Декілька порад для тих, хто створює документи LaTeX українською мовою. Більшість сучасних дистрибутивів TeX/LaTeX містять пакет babel із стандартною підтримкою української мови, кириличні LH шрифти, і таблиці українських переносів.

Якщо ви збираєтеся використовувати кирилицю в математичних формулах, спочатку у преамбулі документа необхідно підключити пакет

```
\usepackage{mathtext}
```

Далі потрібно задати кодування шрифту[9] [10], воно повинно збігатися з кодуванням таблиці українських переносів[11], з якою зґенеровано файл формату (`latex.fmt`):

```
\usepackage[T2A]{fontenc}
```

Потім слід вибрати вхідне кодування. В залежності від платформи, на якій ви працюєте, це може бути `koi8-u` (UNIX, Linux), `cp866nav` (DOS), `cp1251` (Windows), або `macukr` (Mac):

```
\usepackage[koi8-u]{inputenc}
```

і, власне, завантажити підтримку української мови:

```
\usepackage[english,ukrainian]{babel}
```

Остання команда вмикає українські переноси (якщо ваш LaTeX відповідним чином налаштований) а також змінює всі автоматичні тексти українською, наприклад, «Chapter» стає «Розділ». Крім того, активізуються деякі установки, наприклад, `\frenchspacing`, та стають доступними ряд команд, що допомагають створювати документи українською мовою у відповідності з типографськими стандартами (див. табл. 2.3).

Українська опція пакета babel визначає команди `\Asbuk` та `\asbuk`, які працюють так же як і `\Alph` та `\alph`, але ґенерують, відповідно,

---

[8] Цей розділ додано під час перекладу на заміну розділу про підтримку німецької мови.

[9] Український алфавіт підтримують кодування T2A, LCY, X2, і, частково, OT2.

[10] Якщо ви використовуєте пакети AMS-LaTeX, завантажуйте їх перед пакетами fontenc і babel.

[11] Про те, як встановити українські переноси, читайте в `CTAN:/tex-archive/language/hyphenation/ukrhyph/README` та [18], де описані також інші аспекти українізації LaTeX.

Табл. 2.3: Деякі з команд, визначених українською опцією пакета babel

`"-`	вставляє в слово можливу точку переносу, але, на відміну від `\-`, не забороняє переноси в інших точках слова згідно до діючих правил.
`"---`	довге кириличне тире для тексту. Воно трошки коротше, ніж англійське, обмежується короткими жорсткими пробілами, і не відривається від попереднього слова.
`"=`	дефіс в складних словах, що дозволяє перенос в цьому місці і в решті слова (в LaTeX звичайний дефіс не дозволяє розрив складного слова для переносу).
`"` "'`	«німецькі» ліві і праві подвійні лапки („ “).
`"< ">`	ліві і праві подвійні лапки «ялинка» (« »).
`\dq`	прямі подвійні лапки (").

великі і малі літери української абетки, що використовуються для переліків. Наприклад, для того, щоб списки другого рівня нумерувались маленькими українськими літерами, потрібно

```
\renewcommand\theenumii{\asbuk{enumii}}
```

2.6 Пробіли між словами

Для отримання рівного правого краю LaTeX змінює ширину інтервалів між словами. Інтервал наприкінці речення він робить дещо більшим, роблячи текст більш читабельним[12]. LaTeX вважає, що речення закінчуються крапками, знаками питання, чи знаками оклику. Якщо крапка йде після великої літери, вона не вважається кінцем речення, тому що крапки після великих літер звичайно використовуються для скорочень.

Будь-які винятки з цих припущень повинні бути явно зазначені в тексті. Знак «\» перед пробілом дає в результаті пробіл, який не буде збільшений. Знак «~» дає пробіл, що не може збільшитися і який, крім того, забороняє розрив рядка. Команда `\@` перед крапкою вказує, що ця крапка закінчує речення, незважаючи на те, що вона йде після великої літери.

```
Mr.~Smith was happy to see her\\
cf.~Fig.~5\\
I like BASIC\@. What about you?
```

Mr. Smith was happy to see her
cf. Fig. 5
I like BASIC. What about you?

[12] У відповідності з традиціями набору, прийнятими в англійській мові. (*Прим. пер.*)

Додатковий пробіл після крапок можна заборонити командою

```
\frenchspacing
```

яка вказує LaTeX *не* робити пробіл після крапки більшим, ніж після звичайних символів. Це звичайно для мов, відмінних від англійської, за винятком бібліоґрафій. Якщо ви використовуєте `\frenchspacing`, команда `\@` не потрібна[13].

2.7 Заголовки, частини і розділи

Щоб допомогти читачу орієнтуватися у вашому документі, його потрібно розділяти на частини, розділи, та інші структурні одиниці. LaTeX підтримує такий поділ за допомогою спеціальних команд, аргументами яких є заголовки відповідних структурних одиниць. Ваша справа — використовувати їх у належному порядку.

В класі `article` є такі команди групування:

```
\section{...}
\subsection{...}
\subsubsection{...}
\paragraph{...}
\subparagraph{...}
```

Якщо потрібно розділити документ на частини, не впливаючи на нумерацію розділів та частин, можна використати команду

```
\part{...}
```

У класах `report` і `book` є команда для розділів вищого рівня

```
\chapter{...}
```

Оскільки в класі `article` немає розділів (chapters), статті досить легко об'єднати як розділи книги. Інтервали між розділами, нумерація і розмір шрифту заголовків LaTeX встановлює автоматично.

Існує дві особливі команди секціонування:

- Команда `\part` не впливає на послідовність нумерування розділів.
- Команда `\appendix` аргументу не має. Вона просто починає нумерувати розділи літерами замість цифр[14] і замінює «Розділ» на «Додаток».

---

[13] Якщо ви використовуєте пакет `babel`, українська мова автоматично активізує `\frenchspacing`. (*Прим. пер.*)

[14] У класі `article` міняється нумерація підрозділів (sections).

LaTeX створює зміст, беручи заголовки розділів і номери сторінок з попереднього проходу по документу. Команда

`\tableofcontents`

створює зміст у тому місці, де вона викликана. Щоб отримати правильний зміст, новий документ повинен бути оброблений LaTeX двічі. В особливих випадках може бути необхідний і третій прохід. LaTeX попередить вас, коли це буде потрібно.

Усі перераховані вище команди групування існують також у варіантах із зірочкою. Такі варіанти можна отримати додаванням `*` до назви команди. Вони генерують заголовки розділів, що не нумеруються і не включаються в зміст. Наприклад, замість команди `\section{Довідка}` слід використати `\section*{Довідка}`.

Як правило, заголовки розділів з'являються в змісті точно у тому ж вигляді, в якому вони набрані в тексті. Іноді це неможливо через те, що заголовок задовгий для змісту. Варіант заголовку розділу для змісту в таких випадках можна вказати як необов'язковий аргументом перед власне заголовком.

```
\chapter[Заголовок для змісту]{Довгий
 дуже нудний заголовок, показаний в тексті}
```

Титульний аркуш усього документа генерується за допомогою команди

`\maketitle`

Його вміст визначається командами

`\title{...}`, `\author{...}` і, при бажанні, `\date{...}`

перед тим, як буде дано команду `\maketitle`. Аргумент команди `\author` може містити кілька назв, розділених командами `\and`.

Приклад використання деяких із згаданих команд можна знайти на рис. 1.2.

Крім описаних вище команд групування, LaTeX 2_ε уводить три додаткові команди для використання з класом `book`.

`\frontmatter`, `\mainmatter` і `\backmatter`

Вони корисні для поділу вашої публікації. Команди змінюють заголовки розділів і нумерацію сторінок так, як це очікується від книги:

`\frontmatter` Повинна бути першою командою після `\begin{document}`. Вона активізує нумерування сторінок римськими цифрами. Звичайним є використання команд секціонування з зірочками (наприклад, `\chapter*{Передмова}`) для вступної частини книги, оскільки в такому випадку LaTeX не нумеруватиме їх.

`\mainmatter` іде якраз перед першим розділом книги. Вона обнуляє лічильник сторінок і починає нумерувати їх арабськими цифрами.

`\appendix` вказує на початок додатків вашої книги. Після цієї команди розділи будуть пронумеровані літерами.

`\backmatter` повинна стояти перед останніми елементами вашої книги, такими, як бібліоґрафія та предметний покажчик. В стандартних класах документів це не матиме візуального ефекту.

2.8 Перехресні посилання

У книгах, звітах і статтях часто зустрічаються перехресні посилання на ілюстрації, таблиці й окремі частини тексту. LaTeX надає наступні команди для таких посилань:

`\label{`*мітка*`}`, `\ref{`*мітка*`}` і `\pageref{`*мітка*`}`

де *мітка* — обраний користувачем ідентифікатор. LaTeX заміняє `\ref` номером розділу, підрозділу, ілюстрації, таблиці, чи теореми, де була використана відповідна команда `\label`. `\pageref` друкує номер сторінки, на якій знаходиться команда `\label`[15]. Так само, як і у випадку з заголовками розділів, тут використовуються номери з попереднього проходу.

```
Посилання на цей
розділ~\label{sec:this}
має такий вигляд: <<див.
розділ~\ref{sec:this} на
ст.~\pageref{sec:this}.>>
```

Посилання на цей розділ має такий вигляд: «див. розділ 2.8 на ст. 32.»

2.9 Виноски

Команда

`\footnote{`*текст виноски*`}`

друкує виноску внизу поточної сторінки. Виноски завжди повинні мі-

[15] Зверніть увагу, що ці команди не знають, на що саме вони посилаються. `\label` просто зберігає останній автоматично зґенерований номер.

ститися після слова, або речення, якого вони стосуються. Виноска, що стосується речення або його частини, повинна йти після коми чи крапки[16] [17].

```
Користувачі \LaTeX{}
часто вживають
виноски\footnote{%
  Це "--- виноска.}.
```

Користувачі LaTeX часто вживають виноски[a].

[a]Це — виноска.

2.10 Виділення

У машинописних документах `важливі слова виділяють` <u>`підкресленням`</u>.

`\underline{`*текст*`}`

У друкованих виданнях ці слова виділяються *курсивом*. В LaTeX для цього слугує команда

`\emph{`*текст*`}`

Її аргументом є текст для виділення. Що насправді робить ця команда, залежить від контексту:

```
Якщо ви використовуєте
\emph{виділення в уже
виділеному тексті,
\LaTeX{} використовує
\emph{прямий} шрифт.}
```

Якщо ви використовуєте *виділення в уже виділеному тексті, LaTeX використовує* прямий *шрифт.*

Зважте на відмінність між командами *виділення* і *зміни шрифту*:

```
\textit{Ви можете також
  \emph{виділити} текст,
  набравши його курсивом,}
\textsf{шрифтом без
  \emph{зарубок,}}
\texttt{чи в стилі
  \emph{друкарської машинки}.}
```

Ви можете також виділити *текст, набравши його курсивом,* шрифтом без *зарубок,* `чи в стилі` *`друкарської машинки`*.

[16]Це в англійській мові, а в українській виноски йдуть безпосередньо після слова, перед розділовими знаками. (*Прим. пер.*)

[17]Зважте, виноски відволікають читача від основної частини документа. В кінці кінців, усі читають виноски, ми допитливі істоти. То, чому не інтегрувати усе, що ви бажаєте сказати, в тіло документа?[18]

[18]Вказівник не обов'язково йде туди куди він вказує :-).

2.11 Оточення

Для набору спеціальних видів тексту LaTeX визначає безліч оточень для різних типів форматування:

`\begin{`*оточення*`}` *текст* `\end{`*оточення*`}`

де *оточення* — назва оточення. Оточення можна викликати усередині інших оточень, дотримуючись порядку виклику і повернення:

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

У наступних розділах розповідається про усі важливі оточення.

2.11.1 Список, перелік та опис

Оточення `itemize` підходить для простих списків, оточення `enumerate` — для нумерованих списків, а оточення `description` — для описів.

```
\flushleft
\begin{enumerate}
\item Ви можете на ваш
смак змішувати
оточення списків:
\begin{itemize}
\item Але це може виглядати
нерозумно.
\item[--] З тире.
\end{itemize}
\item Тому пам’ятайте:
\begin{description}
\item[Дурниці] не стануть
розумнішими якщо їх
розмістити списком.
\item[Розумні] речі, однак,
цілком можна представити
у вигляді списку.
\end{description}
\end{enumerate}
```

1. Ви можете на ваш смак змішувати оточення списків:
   - Але це може виглядати нерозумно.
   - – З тире.
2. Тому пам’ятайте:

   Дурниці не стануть розумнішими якщо їх розмістити списком.

   Розумні речі, однак, цілком можна представити у вигляді списку.

2.11.2 Вирівнювання ліворуч, праворуч і по центру

Оточення `flushleft` і `flushright` форматують абзаци, вирівняними ліворуч чи праворуч. Оточення `center` дає відцентрований текст. Якщо ви не використовуєте `\\` щоб вказати розриви рядків, LaTeX визначить їх автоматично.

```
\begin{flushleft}
  Цей текст\\ вирівняний ліворуч.
  \LaTeX{} не намагається зробити
  усі рядки однакової довжини.
\end{flushleft}
```

Цей текст
вирівняний ліворуч. LaTeX не намагається зробити усі рядки однакової довжини.

```
\begin{flushright}
  Цей текст\\ вирівняний праворуч.
  \LaTeX{} не намагається зробити
  усі рядки однакової довжини.
\end{flushright}
```

Цей текст
вирівняний праворуч. LaTeX не намагається зробити усі рядки однакової довжини.

```
\begin{center}
  У центрі\\Землі
\end{center}
```

У центрі
Землі

2.11.3 Цитати та вірші

Оточення `quote` корисно для цитат, важливих фраз і прикладів.

```
Типографське правило для
довжини рядка:
\begin{quote}
В середньому, довжина рядка
не повинна бути більшою
від 66~символів.
\end{quote}
Ось чому \LaTeX{} робить
поля сторінок такими
широкими, а в газетах
часто застосовують набір
у декілька шпальт.
```

Типографське правило для довжини рядка:

> В середньому, довжина рядка не повинна бути більшою від 66 символів.

Ось чому LaTeX робить поля сторінок такими широкими, а в газетах часто застосовують набір у декілька шпальт.

Існують ще два схожих оточення: `quotation` і `verse`. Для цитат, що охоплюють кілька абзаців, корисне оточення `quotation`, тому що воно робить абзацний відступ. Оточення `verse` використовують для віршів, де важливі розриви рядків. Рядки розділяються за допомогою \\ наприкінці рядка, і порожнього рядка після кожної строфи.

```
Я знаю тільки один англійський
віршик напам'ять:
\begin{flushleft}
\begin{verse}
Humpty Dumpty sat on a wall:\\
Humpty Dumpty had a great fall.\\
All the King's horses and all
the King's men\\
Couldn't put Humpty together
again.
\end{verse}
\end{flushleft}
```

Я знаю тільки один англійський віршик напам'ять:

> Humpty Dumpty sat on a wall:
> Humpty Dumpty had a great fall.
> All the King's horses and all the King's men
> Couldn't put Humpty together again.

2.11.4 Буквальне відтворення

Текст, розміщений між `\begin{verbatim}` і `\end{verbatim}` буде відтворений так, наче він надрукований друкарською машинкою, із усіма пробілами і розривами рядків, без виконання будь-яких команд LaTeX.

Усередині абзацу аналогічну функцію виконує команда

`\verb+`*текст*`+`

Тут «+» — це тільки приклад символа-обмежувача. Ви може використовувати будь-який символ, крім літер, «*» чи пробілу. Багато прикладів LaTeX у цьому буклеті набрані з використанням цієї команди.

```
Команда \verb|\ldots| \ldots

\begin{verbatim}
10 PRINT "HELLO WORLD ";
20 GOTO 10
\end{verbatim}
```

Команда `\ldots` ...

```
10 PRINT "HELLO WORLD ";
20 GOTO 10
```

```
\begin{verbatim*}
варіант   оточення
verbatim         з
зірочкою виділяє
пробіли   в  тексті
\end{verbatim*}
```

```
варіант␣␣␣оточення
verbatim␣␣␣␣␣␣␣␣␣з
зірочкою␣виділяє
пробіли␣␣␣в␣␣тексті
```

Команду `\verb` теж можна використовувати аналогічним чином із зірочкою:

```
\verb*|Ось   так :-) |
```

`Ось␣␣␣так␣:-)␣`

Оточення `verbatim` і команду `\verb` не можна використовувати усередині параметрів інших команд.

2.11.5 Таблиці

Оточення `tabular` використовують для набору таблиць з можливістю формування горизонтальних і вертикальних ліній. LaTeX автоматично визначає ширину стовпчиків.

Аргумент *специфікація* команди

`\begin{tabular}[`*поз*`]{`*специфікація*`}`

визначає формат таблиці. Використовуйте `l` для стовпчика тексту, вирівняного вліво, `r` — для тексту, вирівняного вправо і `c` — для центрованого тексту; `p{`*ширина*`}` — для стовпчика зазначеної ширини, що містить вирівняний текст із переносом рядків, і `|` — для вертикальної лінії.

поз визначає вертикальну позицію усього середовища `tabular`, `t`, `b` та `c` вказують, вирівнювати по верхній, нижній межі, чи по центру середовища.

Усередині оточення `tabular` стовпчики розділяються знаком «`&`», команда `\\` починає новий рядок, а `\hline` вставляє горизонтальну лінію. Ви можете вставити часткову горизонтальну лінію використовуючи `\cline{`*j*`-`*i*`}`, де j та i — номери стовпчиків, на які лінія повинна розповсюджуватися.

```
\begin{tabular}{|r|l|}
\hline
54 & шістнадцяткове \\
124 & вісімкове \\
1010100 & двійкове \\
\hline \hline
84 & десяткове \\
\hline
\end{tabular}
```

54	шістнадцяткове
124	вісімкове
1010100	двійкове
84	десяткове

```
\begin{tabular}{|p{4.7cm}|}
\hline
Ласкаво просимо до абзацу
у рамочці. Сподіваємося, вам усім
тут сподобається.\\
\hline
\end{tabular}
```

Ласкаво просимо до абзацу у рамочці. Сподіваємося, вам усім тут сподобається.

Роздільник стовпчиків можна задати конструкцією `@...`. Ця команда видаляє пробіл між стовпчиками і заміняє його на те, що включено у фігурні дужки. Одне з частих використань цієї команди буде показано далі, коли йтиметься про вирівнювання по десятковій крапці. Інше можливе використання — для заборони ведучого пробілу в таблиці за допомогою `@`:

```
\begin{tabular}{@{} l @{}}
\hline
немає ведучого пробілу\\
\hline
\end{tabular}
```

немає ведучого пробілу

```
\begin{tabular}{l}
\hline
ведучий пробіл ліворуч і праворуч\\
\hline
\end{tabular}
```

ведучий пробіл ліворуч і праворуч

Оскільки в LaTeX не існує стандартного способу вирівняти стовпчики чисел по десятковій крапці[19], ми можемо «обдурити» TeX і домогтися цього за допомогою двох стовпчиків: вирівняної вправо цілої частини і вирівняної вліво дробової. Команда `@{.}` у рядку `\begin{tabular}` замінює нормальний пробіл між стовпчиками просто на «.», даючи ефект одного стовпчика, вирівняного по десятковій крапці. Не забудьте замінити у ваших числах крапку на роздільник стовпчиків (`&`)! Мітку стовпчика можна помістити над нашим числовим «стовпчиком» командою `\multicolumn`:

```
\begin{tabular}{c r @{.} l}
Вираз із $\pi$       &
\multicolumn{2}{c}{Значення} \\
\hline
$\pi$               & 3&1416  \\
$\pi^{\pi}$         & 36&46   \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

Вираз із π	Значення
π	3.1416
π^{π}	36.46
$(\pi^{\pi})^{\pi}$	80662.7

```
\begin{tabular}{|c|c|}
\hline
\multicolumn{2}{|c|}{Ene} \\
\hline
Mene & Muh! \\
\hline
\end{tabular}
```

Ene	
Mene	Muh!

[19] Якщо на вашій системі встановлений комплект 'tools', зверніть увагу на пакет dcolumn.

Матеріал, набраний за допомогою середовища tabular, завжди залишається на одній сторінці. Якщо потрібно набирати довші таблиці, зверніть увагу на середовища supertabular і longtabular.

2.12 Плаваючі об'єкти

Більшість публікацій у наші дні містять безліч ілюстрацій і таблиць. Ці елементи потребують особливого поводження, оскільки їх не можна розбити між сторінками. Одним із рішень було б починати нову сторінку щоразу, коли ілюстрація чи таблиця занадто велика, щоб поміститися на поточній сторінці. Це залишало б сторінки частково порожніми, що виглядає не дуже гарно.

Вирішення цієї проблеми є «перепливання» будь-якої ілюстрації чи таблиці, що не вміщується на поточній сторінці, на наступну, і заповнення поточної текстом. LaTeX пропонує для плаваючих об'єктів два оточення, одне — для таблиць, і одне — для ілюстрацій. Щоб у повній мірі використати їхні переваги, важливо хоч приблизно уявляти, як LaTeX обробляє плаваючі об'єкти. Інакше вони можуть стати джерелом розчарування, коли LaTeX розміщуватиме їх не там, де ви хочете.

Давайте спочатку розглянемо команди LaTeX, призначені для плаваючих об'єктів.

Будь-який матеріал, включений в оточення `figure` чи `table`, трактується як плаваючий. Обидва оточення

`\begin{figure}[`*специфікація розміщення*`]` чи
`\begin{table}[`*специфікація розміщення*`]`

мають необов'язковий параметр, названий *специфікацією розміщення*. Цей параметр використовується для вказівки, куди LaTeX може переміщувати даний плаваючий об'єкт. *Специфікація розміщення* будується складанням у рядок *ключів розміщення плаваючого об'єкта*. Див. таблицю 2.4.

Наприклад, таблицю можна почати наступним рядком:

```
\begin{table}[!hbp]
```

Специфікація розміщення `[!hbp]` дозволяє LaTeX розмістити таблицю прямо тут (`h`), внизу тієї ж сторінки (`b`), чи на окремій сторінці (`p`), і навіть у тому випадку, якщо усе це не буде виглядати вельми добре (`!`). Якщо ніякої специфікації розміщення не задано, стандартні класи використовують `[tbp]`.

LaTeX розміщує кожен плаваючий об'єкт у відповідності із заданою автором специфікацією. Якщо об'єкт не можна помістити на поточній сторінці, він розміщується у чергу ілюстрацій чи в чергу таблиць[20]. Ко-

[20] Які є чергами типу *FIFO*: «першим увійшов — першим вийшов».

ли починається нова сторінка, LaTeX спершу перевіряє, чи можна заповнити спеціальну сторінку плаваючими об'єктами з черг. Якщо ні, то перший об'єкт з кожної черги вважається таким, що тільки що зустрівся у тексті: LaTeX знову намагається розмістити їх у відповідності із специфікаціями (за винятком 'h', що вже неможливо). Нові плаваючі об'єкти розміщуються у відповідні черги. LaTeX зберігає порядок, у якому зустрілися плаваючі об'єкти відповідного типу. Тому ілюстрація, яку не вдалося розмістити, відштовхує всі подальші ілюстрації до кінця документа. Отже:

> Якщо LaTeX не розміщує плаваючі об'єкти так, як ви цього очікуєте, то часто це тому, що якийсь об'єкт влаштував затримку в одній із двох черг.

Незважаючи на те, що специфікацію розміщення з одним варіантом задати можливо, робити цього не слід, оскільки це може призвести до проблем. Якщо плаваючий об'єкт не може бути розміщений в заданій позиції, він «застряє» і блокує наступні плаваючі об'єкти. Зокрема, ніколи не використовуйте опцію `[h]`: це настільки погано, що останні версії LaTeX навіть автоматично замінюють її на `[ht]`.

Після пояснення найскладнішої частини, залишається ще кілька заува-

Табл. 2.4: Ключі розміщення плаваючого об'єкта

Ключ	Дозволяє розміщувати об'єкт ...
`h`	*тут (here)*, у тому самому місці тексту, де він з'явився. Звичайно використовується для маленьких об'єктів.
`t`	*угорі (top)* сторінки
`b`	*унизу (bottom)* сторінки
`p`	на *спеціальній сторінці*, що містить тільки плаваючі об'єкти.
`!`	не звертати увагу на більшість внутрішніх параметрів[a], що можуть запобігти розміщенню цього об'єкта згідно з вашими специфікаціями.

Зверніть увагу, `pt` та `em` — одиниці виміру TeX units. Дивіться також таблицю 6.5 на сторінці 106.

[a]Таких, як максимальне число плаваючих об'єктів, дозволених на одній сторінці

жень стосовно оточень `table` і `figure`. Командою

`\caption{`*текст заголовка*`}`

ви можете задати заголовок для об'єкта. LaTeX додасть рядок «Таблиця» чи «Рис.» і поточний номер.

Дві команди

`\listoffigures` і `\listoftables`

працюють аналогічно команді `\tableofcontents`, виводячи список ілюстрацій чи таблиць, відповідно. У цих списках заголовки повторюються повністю. Якщо ви використовуєте довгі заголовки, то можна надати їхній короткий варіант для включення у списки, розмістивши його в квадратних дужках після команди `\caption`.

```
\caption[Короткий]{Дддддоооооввввввгггггггиииииииииййй}
```

Можна робити посилання на плаваючий об'єкт за допомогою команди `\label` після заголовка і команди `\ref` у тексті.

Наступний приклад зображує квадрат і вставляє його в документ. Подібну техніку можна використовувати, щоб залишити в документі місце під зображення, яке ви вставите пізніше.

```
Рис.~\ref{white} "--- приклад Поп-Арту.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{П'ять на п'ять сантиметрів} \label{white}
\end{figure}
```

У вище наведеному прикладі LaTeX буде *дуже сильно* (`!`) намагатися розмістити ілюстрацію прямо *тут* (`h`)[21]. Якщо це неможливо, він спробує розмістити її *унизу сторінки* (`b`). Якщо йому не вдасться помістити ілюстрацію на поточній сторінці, він спробує з'ясувати, чи можна створити сторінку плаваючих об'єктів, що містить цю ілюстрацію і, можливо, деякі таблиці з черги таблиць. Якщо для окремої сторінки матеріалу ще не нагромадилося, LaTeX починає нову сторінку і знову обробляє ілюстрацію так, наче вона тільки що з'явилася в тексті.

У певних випадках може бути потрібно використати команду

`\clearpage` чи навіть `\cleardoublepage`

Це примушує LaTeX негайно вивести усі плаваючі об'єкти, що залишалися в чергах, і потім почати нову сторінку. `\cleardoublepage`, крім цього, починає нову непарну сторінку.

[21] Припускаючи, що черга ілюстрацій порожня.

Пізніше ви довідаєтеся, як включати у ваші документи LaTeX малюнки в форматі PostScript.

2.13 Захист ламких команд

Текст, заданий як аргумент команд на зразок `\caption` чи `\section`, може з'явитися в документі кілька разів (наприклад, у змісті та в тілі документа). Деякі команди не спрацьовують, коли їх використовувати у аргументі `\section`-подібних команд. Такі команди називаються ламкими командами — наприклад, `\footnote` чи `\phantom`. Ці ламкі команди потребують захисту (як і всі ми). Ви можете захистити їх, поставивши попереду команду `\protect`.

`\protect` розповсюджується тільки на наступну команду, навіть не на її аргумент. У більшості випадків додаткові `\protect` не зашкодять.

```
\section{Я завбачливий
      \protect\footnote{і захищаю мої виноски}}
```

Розділ 3

Набір математичних формул

От тепер ви готові! У цьому розділі ми ознайомимося з найсильнішим місцем TEX: набором математики. Але майте на увазі, що цей розділ дає тільки поверхневий огляд. Хоча для багатьох з вас викладених тут речей буде досить, не впадайте у відчай, якщо ви не зможете знайти рішення, яке б задовольнило потреби набору вашої математики. Дуже вірогідно, що ваша проблема може бути вирішена за допомогою AMS-LATEX[1] [2] чи якогось іншого пакета.

3.1 Загальні відомості

LATEX має спеціальний режим для набору математики. Формули можуть набиратися всередині абзацу, або винесені як окремий абзац. Математичний текст усередині абзацу вводиться між `\(` і `\)`, між `$` і `$`, або між `\begin{math}` і `\end{math}`.

```
Додайте $a$ у квадраті і
$b$ у квадраті, щоб отримати
$c$ у квадраті. Або,
мовою математики:
$c^{2}=a^{2}+b^{2}$
```

Додайте a у квадраті і b у квадраті, щоб отримати c у квадраті. Або, мовою математики: $c^2 = a^2 + b^2$

[1] *Американським Математичним Товариством* створено потужне розширення LATEX. Велика кількість прикладів цього розділу використовує це розширення. Воно входить у всі останні дистрибутиви TEX. Якщо у вас його не вистачає загляніть на `CTAN:/tex-archive/macros/latex/required/amslatex`

[2] Якщо ви використовуєте пакети AMS-LATEX в документах українською мовою, завантажуйте їх перед пакетами fontenc і babel. (*Прим. пер.*)

```
\TeX{} вимовляється як
\(\tau\epsilon\chi\).\\[6pt]
100~м$^{3}$ води.\\[6pt]
Від усього
\begin{math}\heartsuit\end{math}.
```

TEX вимовляється як $\tau\epsilon\chi$.

100 м^{3} води.

Від усього $\heartsuit$.

Якщо ви бажаєте, щоб великі математичні рівняння чи формули бажано були набрані окремо від решти абзацу, їх бажано «експонувати», а не просто набирати як окремий абзац. Для цього їх потрібно розмістити між `\[` і `\]` або всередині оточення `displaymath`.

```
Додавши $a$ у квадраті і
$b$ у квадраті, отримаємо
$c$ у квадраті. Або,
мовою математики:
\begin{displaymath}
  c^{2}=a^{2}+b^{2}
\end{displaymath}
або, щоб менше набирати,
\[a+b=c\]
```

Додавши a у квадраті і b у квадраті, отримаємо c у квадраті. Або, мовою математики:

$$c^2 = a^2 + b^2$$

або, щоб менше набирати,

$$a + b = c$$

Якщо ви хочете, щоб LATEX нумерував ваші формули, користуйтеся оточенням `equation`. За допомогою `\label` можна помітити формулу, а потім послатися на нього в іншому місці тексту за допомогою `\ref` або `\eqref`:

```
\begin{equation}
  \label{eq:eps}
  \epsilon > 0
\end{equation}
З (\ref{eq:eps})
випливає \ldots
З (\eqref{eq:eps})
випливає те ж саме.
```

$$\epsilon > 0 \qquad (3.1)$$

З (3.1) випливає … З ((3.1)) випливає те ж саме.

Зверніть увагу, що вирази в експонованих формулах будуть набрані в іншому стилі, ніж у тексті:

```
$\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}$
```

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

```
\begin{displaymath}
  \lim_{n \to \infty}
  \sum_{k=1}^n \frac{1}{k^2}
  = \frac{\pi^2}{6}
\end{displaymath}
```

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

Математичний режим відрізняється від *текстового режиму*. Наприклад, у *математичному режимі*:

1. Більшість пробілів і кінців рядків не приймаються до уваги, тому що всі пробіли або залежать від логіки математичних виразів, або повинні бути задані явно такими командами, як `\,`, `\quad`, чи `\qquad`.

2. Порожні рядки неприпустимі. Кожна формула займає тільки один абзац.

3. Кожна літера вважається назвою змінної, і відповідно набирається. Якщо потрібно ввести у формулу нормальний текст (нормальний прямий шрифт із нормальними пробілами), то слід користуватися командами `\textrm{...}` (дивіться також розділ 3.7 на ст. 53).

```
\begin{equation}
  \forall x \in \mathbf{R}:
  \qquad x^{2} \geq 0
\end{equation}
```

$$\forall x \in \mathbf{R}: \qquad x^2 \geq 0 \tag{3.2}$$

```
\begin{equation}
  x^{2} \geq 0\qquad
  \textrm{для усіх }x\in
  \mathbf{R}
\end{equation}
```

$$x^2 \geq 0 \qquad \textrm{для усіх } x \in \mathbf{R} \tag{3.3}$$

Математики бувають дуже вибагливі щодо використовуваних символів: у цьому випадку було б традиційним використання «blackboard bold» символів, для отримання яких служить команда `\mathbb` з пакетів amsfonts чи amssymb. Останній приклад тепер матиме такий вигляд:

```
\begin{displaymath}
  x^{2} \geq 0\qquad
  \textrm{для всіх }x\in
  \mathbb{R}
\end{displaymath}
```

$$x^2 \geq 0 \qquad \textrm{для всіх } x \in \mathbb{R}$$

3.2 Групування в математичному режимі

Більшість команд математичного режиму діє тільки на наступний символ, тому, якщо ви хочете, щоб команда стосувалася кількох символів, вам потрібно згрупувати їх разом за допомогою фігурних дужок: `{...}`.

```
\begin{equation}
  a^x+y \neq a^{x+y}
\end{equation}
```

$$a^x + y \neq a^{x+y} \tag{3.4}$$

3.3 Складові математичної формули

У цьому розділі будуть описані найбільш важливі команди, що використовуються в математичному наборі. Детальний перелік команд для набору математичних символів дивіться у розділі 3.10 на сторінці 56.

Малі грецькі літери вводяться як `\alpha`, `\beta`, `\gamma`, ..., великі літери — як `\Gamma`, `\Delta`, ...[3]

```
$\lambda,\xi,\pi,\mu,%
\Phi,\Omega$
```

$\lambda, \xi, \pi, \mu, \Phi, \Omega$

Верхні і нижні індекси вводяться за допомогою символів «`^`» та «`_`».

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

a_{1} $\qquad$ x^{2} $\qquad$ $e^{-\alpha t}$ $\qquad$ a^{3}_{ij}
$e^{x^2} \neq {e^x}^2$

Квадратний корінь вводиться як `\sqrt`; корінь n-ного ступеня — за допомогою `\sqrt[`n`]`. Розмір знака кореня вибирається LaTeX автоматично. Якщо потрібен один тільки знак, використовуйте `\surd`.

```
$\sqrt{x}$ \qquad
$\sqrt{ x^{2}+\sqrt{y} }$
\qquad $\sqrt[3]{2}$\\[3pt]
$\surd[x^2 + y^2]$
```

$\sqrt{x}$ $\qquad$ $\sqrt{ x^{2}+\sqrt{y} }$ $\qquad$ $\sqrt[3]{2}$
$\surd[x^2 + y^2]$

Команди `\overline` і `\underline` створюють **горизонтальні лінії** відразу над чи під виразом.

```
$\overline{m+n}$
```

$\overline{m+n}$

Команди `\overbrace` і `\underbrace` створюють довгі **горизонтальні фігурні дужки** відразу над чи під виразом.

[3] У LaTeX 2_ε велика «альфа» не визначена, тому що вона виглядає так само, як латинська «A». Коли буде завершено розробку нового кодування математичних символів, це буде змінено.

```
$\underbrace{ a+b+\cdots%
  +z }_{26}$
```

$\underbrace{a + b + \cdots + z}_{26}$

Для додавання знаків математичних акцентів, таких, як маленька стрілка чи знак тильда, до змінних, можна скористатися командами, перерахованим у таблиці 3.1. Широкі «капелюшки» і тильди, що охоплюють кілька символів, генеруються командами `\widetilde` і `\widehat`. Символ «`'`» дає знак похідної.

```
\begin{displaymath}
  y=x^{2}\qquad y'=2x
  \qquad y''=2
\end{displaymath}
```

$$y = x^2 \qquad y' = 2x \qquad y'' = 2$$

Вектори часто вказуються додаванням маленьких стрілок над змінною. Це робиться командою `\vec`. Для позначення вектора від A до B корисні дві команди `\overrightarrow` і `\overleftarrow`.

```
\begin{displaymath}
  \vec a\quad
  \overrightarrow{AB}
\end{displaymath}
```

$$\vec a \quad \overrightarrow{AB}$$

Для позначення оператора множення крапка звичайно не набирається. Проте, іноді її корисно використовувати, щоб допомогти оку читача згрупувати формулу. В такому випадку слід вживати `\cdot`:

```
\begin{displaymath}
v = {\sigma}_1 \cdot {\sigma}_2
    {\tau}_1 \cdot {\tau}_2
\end{displaymath}
```

$$v = {\sigma}_1 \cdot {\sigma}_2 {\tau}_1 \cdot {\tau}_2$$

Назви log-подібних функцій набираються прямим шрифтом, а не курсивом, як змінні, тому LaTeX має наступні команди для набору назв найбільш важливих функцій:

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

```
\[\lim_{x \rightarrow 0}
\frac{\sin x}{x}=1\]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1$$

Для функції модуля існує дві команди: `\bmod` для бінарного оператора «$a \bmod b$» і `\pmod` для таких виразів як «$x \equiv a \pmod{b}$».

```
$a\bmod b$\\
$x\equiv a \pmod{b}$
```

$a \bmod b$
$x \equiv a \pmod{b}$

Дроби набираються командою `\frac{...}{...}`. Іноді перевагу надають дробам з похилою рискою $1/2$, оскільки вона краще виглядає при невеликій кількості «дробового матеріалу».

```
$1\frac{1}{2}$~години
\begin{displaymath}
  \frac{ x^{2} }{ k+1 }\qquad
  x^{ \frac{2}{k+1} }\qquad
  x^{ 1/2 }
\end{displaymath}
```

$1\frac{1}{2}$ години

$$\frac{x^2}{k+1} \qquad x^{\frac{2}{k+1}} \qquad x^{1/2}$$

Для набору біноміальних коефіцієнтів чи аналогічних структур можна користуватися командою `\binom` з пакета `amsmath`.

```
\begin{displaymath}
\binom{n}{k}\qquad\mathrm{C}_n^k
\end{displaymath}
```

$$\binom{n}{k} \qquad \mathrm{C}_n^k$$

Для відображення бінарних відносин може бути корисним розміщення символів один над другим. `\stackrel` розміщує символ, даний у першому аргументі, з використанням шрифту розміру індексу, над другим, що розміщений на звичайній позиції.

```
\begin{displaymath}
\int f_N(x) \stackrel{!}{=} 1
\end{displaymath}
```

$$\int f_N(x) \stackrel{!}{=} 1$$

Оператор інтеграла вводиться командою `\int`, а **оператор суми** — командою `\sum`. Верхні і нижні границі вказуються за допомогою знаків «^» та «_», так само, як верхні і нижні індекси[4].

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\end{displaymath}
```

$$\sum_{i=1}^{n} \qquad \int_{0}^{\frac{\pi}{2}}$$

Для кращого управління розміщенням індексів в складних виразах пакет `amsmath` надає два додаткові інструменти: команду `\substack` та середовище `subarray`:

[4] $\mathcal{AMS}$-LaTeX, крім того, дає можливість використовувати багаторядкові верхні і нижні індекси.

```
\begin{displaymath}
\sum_{\substack{0<i<n \\ 1<j<m}}
P(i,j) =
\sum_{\begin{subarray}{l} i\in I\\
      1<j<m
   \end{subarray}}     Q(i,j)
\end{displaymath}
```

$$\sum_{\substack{0<i<n \\ 1<j<m}} P(i,j) = \sum_{\begin{subarray}{l} i\in I\\ 1<j<m \end{subarray}} Q(i,j)$$

TEX надає безліч різних символів для **дужок** та інших обмежувачів у TEX існує безліч символів (скажімо, $[\ \langle\ \|\ \ \updownarrow$). Круглі і квадратні дужки можна уводити за допомогою відповідних клавіш, фігурні — `\{`, інші обмежувачі — спеціальними командами (наприклад, `\updownarrow`). Список доступних обмежувачів дивіться в таблиці 3.8 на сторінці 58.

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

$$a,b,c \neq \{a,b,c\}$$

Якщо помістити команду `\left` перед відкриваючим обмежувачем, або команду `\right` — перед закриваючим, то TEX автоматично вибере правильний розмір обмежувача. Зважте, що Ви повинні кожен `\left` закривати відповідним `\right`, і що розмір визначається коректно, тільки якщо обидва набрані в одному рядку. Якщо вам не потрібний правий обмежувач, використовуйте невидимий — «`\right.`»!

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
    \right) ^3
\end{displaymath}
```

$$1 + \left(\frac{1}{1-x^{2}} \right)^3$$

У деяких випадках необхідно вказати коректний розмір математичного обмежувача вручну, це можна зробити, використовуючи `\big`, `\Big`, `\bigg`, і `\Bigg`, як префікси до більшості команд обмежувачів[5].

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

$$\Big((x+1)(x-1) \Big)^{2}$$
$$\big(\Big(\bigg(\Bigg(\quad \big\}\Big\}\bigg\}\Bigg\} \quad \big\|\Big\|\bigg\|\Bigg\|$$

Щоб ввести у формулу **трикрапки**, існує кілька команд. `\ldots` набирає крапки на базовій лінії, `\cdots` — центровані. Крім того, існують команди `\vdots` для вертикальних і `\ddots` для діагональних трикрапок. У розділі 3.5 наведено інший приклад.

[5] Ці команди не працюють як слід, якщо використовуються команди зміни розміру шрифту, чи якщо зазначена опція `11pt` чи `12pt`. Для виправлення цієї поведінки скористуйтеся пакетами exscale чи amsmath.

```
\begin{displaymath}
x_{1},\ldots,x_{n} \qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

$$x_1, \ldots, x_n \qquad x_1 + \cdots + x_n$$

3.4 Математичні пробіли

Якщо обрані TEX пробіли усередині формул незадовільні, ви можете їх налаштувати з використанням спеціальних команд. Команди для маленьких пробілів: `\,` для $\frac{3}{18}$ quad ( ), `\:` для $\frac{4}{18}$ quad ( ) і `\;` для $\frac{5}{18}$ quad ( ). Екранований символ пробілу `\␣` дає пробіл середніх размірів, а `\quad` ( ) і `\qquad` ( ) дають великі пробіли. Розмір `\quad` приблизно відповідає ширині літери «M» у поточному шрифті. Команда `\!` робить від'ємний пробіл розміром $-\frac{3}{18}$ quad ( ).

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int_{D} g(x,y)
  \, \ud x\, \ud y
\end{displaymath}
замість
\begin{displaymath}
\int\int_{D} g(x,y)\ud x \ud y
\end{displaymath}
```

$$\int\!\!\!\int_{D} g(x,y)\, \mathrm{d}x\, \mathrm{d}y$$

замість

$$\int\int_{D} g(x,y)\mathrm{d}x\mathrm{d}y$$

Зверніть увагу, «d» у диференціалі звичайно набирається прямим шрифтом.

AMS-LATEX пропонує інший спосіб точного настроювання пробілів між кількома знаками інтегралів а саме, команди `\iint`, `\iiint`, `\iiiint` і `\idotsint`. Якщо пакет `amsmath` завантажений, попередній приклад можна набирати так:

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\iint_{D} \, \ud x \, \ud y
\end{displaymath}
```

$$\iint_{D} \, \mathrm{d}x \, \mathrm{d}y$$

Деталі дивіться в електронному документі `textmath.tex` (розповсюджує з AMS-LATEX) або в розділі 8 *The LATEX Companion* [4].

3.5 Вертикально розташований матеріал

Для набору **матриць** користуйтесь оточенням `array`. Воно працює подібно до оточення `tabular`. Для розриву рядка використовується команда `\\`.

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

Оточення `array` можна також використовувати для набору виразів, що мають один великий обмежувач, підставивши «`.`» замість невидимого правого обмежувача:

```
\begin{displaymath}
y = \left\{ \begin{array}{ll}
 a & \textrm{якщо $d>c$}\\
 b+x & \textrm{вранці}\\
 l & \textrm{решту дня}
  \end{array} \right.
\end{displaymath}
```

$$y = \left\{ \begin{array}{ll} a & \textrm{якщо $d>c$}\\ b+x & \textrm{вранці}\\ l & \textrm{решту дня} \end{array} \right.$$

Як і у середовищі `tabular`, в середовищі `array` можна також креслити лінії, наприклад для відокремлення елементів матриці:

```
\begin{displaymath}
\left(\begin{array}{c|c}
 1 & 2 \\
\hline
3 & 4
\end{array}\right)
\end{displaymath}
```

$$\left(\begin{array}{c|c} 1 & 2 \\ \hline 3 & 4 \end{array}\right)$$

Для формул, що займають кілька рядків, або для систем рівнянь, замість `equation` користуйтеся оточеннями `eqnarray` і `eqnarray*`. Кожен рядок `eqnarray` отримує окремий номер рівняння. У `eqnarray*` номери не ставляться.

Оточення `eqnarray` і `eqnarray*` працюють на зразок таблиці з трьох стовпчиків формату `{rcl}`, де середній стовпчик використовується для знака рівності, знака нерівності, чи іншого придатного знака. Команда `\\` розбиває рядки.

```
\begin{eqnarray}
f(x) & = & \cos x     \\
f'(x) & = & -\sin x   \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

$$f(x) = \cos x \qquad (3.5)$$

$$f'(x) = -\sin x \qquad (3.6)$$

$$\int_{0}^{x} f(y)dy = \sin x \qquad (3.7)$$

Зверніть увагу, що по обидва боки знаків рівності забагато вільного місця. Його можна зменшити установкою `\setlength\arraycolsep{2pt}`, як у наступному прикладі.

Довгі рівняння не будуть автоматично розбиватися на правильні частини. Автор повинен вказати, де їх розбивати і скільки відступати. Найчастіше для цього використовують наступні методи:

```
{\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!}-{}
                    \nonumber\\
 & & {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}}
```

$$\begin{aligned}\sin x &= x - \frac{x^3}{3!} + \frac{x^5}{5!} - {} \\ &\quad - \frac{x^7}{7!} + \cdots \qquad (3.8)\end{aligned}$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
     -\frac{x^{2}}{2!} +{} }
                    \nonumber\\
 & & {}+\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!}+{}\cdots
\end{eqnarray}
```

$$\begin{aligned}&\cos x = 1 - \frac{x^2}{2!} + {} \\ &\qquad + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots \qquad (3.9)\end{aligned}$$

Команда `\nonumber` змушує LaTeX не генерувати номер для цього рівняння.

Такими методами може бути складно отримати вертикально вирівняні рівняння, що виглядали б правильно; більш потужні альтернативи надає пакет amsmath. (Див. середовища `align`, `flalign`, `gather`, `multline` та `split`).

3.6 Привиди

Ми не можемо бачити привидів, але вони займають певне місце в думках багатьох людей. LaTeX нічим не відрізняється. Ми можемо скористатися цим для деяких цікавих трюків.

Вирівнюючи текст вертикально за допомогою `^` та `_`, LaTeX іноді намагається бути занадто запобігливим. Використовуючи команду `\phantom`, можна зарезервувати місце для символів, які не з'являються в кінцевому виводі. Щоб краще зрозуміти, подивитися наступні приклади.

```
\begin{displaymath}
{}^{12}_{\phantom{1}6}\textrm{C}
\qquad \textrm{замість} \qquad
{}^{12}_{6}\textrm{C}
\end{displaymath}
```

$${}^{12}_{6}\textrm{C} \qquad \textrm{замість} \qquad {}^{12}_{6}\textrm{C}$$

```
\begin{displaymath}
\Gamma_{ij}^{\phantom{ij}k}
\qquad \textrm{замість} \qquad
\Gamma_{ij}^{k}
\end{displaymath}
```

$$\Gamma_{ij}^{k} \qquad \textrm{замість} \qquad \Gamma_{ij}^{k}$$

3.7 Розмір математичного шрифту

У математичному режимі TEX вибирає розмір шрифту в залежності від контексту. Наприклад, індекси набираються меншим шрифтом. Якщо потрібно додати до рівняння звичайний текст, не користуйтеся командою `\textrm`, тому, що механізм переключення розміру працювати не буде, оскільки `\textrm` тимчасово виходить у текстовий режим. Щоб механізм переключення розміру працював як належить, використовуйте команду `\mathrm`. Але майте на увазі, `\mathrm` буде добре працювати тільки з короткими елементами. Пробіли, як і раніше, не діють, і акцентовані символи[6] не працюють[7].

```
\begin{equation}
2^{\textrm{-й}} \quad
2^{\mathrm{-й}}
\end{equation}
```

$$2^{\textrm{-й}} \quad 2^{-} \qquad (3.10)$$

Проте, іноді вам може бути потрібно вказати LATEX точний розмір шрифту. У математичному режимі розмір встановлюється чотирма командами:

`\displaystyle` (123), `\textstyle` (123), `\scriptstyle` ($\scriptstyle 123$) і `\scriptscriptstyle` ($\scriptscriptstyle 123$).

Зміна стилів впливає також на спосіб зображення границь.

```
\begin{displaymath}
\mathop{\mathrm{corr}}(X,Y)=
 \frac{\displaystyle
   \sum_{i=1}^n(x_i-\overline x)
   (y_i-\overline y)}
  {\displaystyle\biggl[
 \sum_{i=1}^n(x_i-\overline x)^2
\sum_{i=1}^n(y_i-\overline y)^2
\biggr]^{1/2}}
\end{displaymath}
```

$$\mathop{\mathrm{corr}}(X,Y)= \frac{\displaystyle \sum_{i=1}^n(x_i-\overline x) (y_i-\overline y)} {\displaystyle\biggl[\sum_{i=1}^n(x_i-\overline x)^2 \sum_{i=1}^n(y_i-\overline y)^2 \biggr]^{1/2}}$$

[6] А також кирилиця. (*Прим. пер.*)

[7] При підключенні $\mathcal{AMS}$-LATEX (пакет amssym) команда `\textrm` починає працювати зі зміною розміру шрифту.

Це один із прикладів, коли нам потрібні більші дужки, ніж надані стандартними `\left[ \right]`.

3.8 Теореми, закони, ...

При написанні математичних документів, вам, імовірно, потрібний спосіб набору «лем», «визначень», «аксіом» і аналогічних структур. LaTeX підтримує це за допомогою команд

`\newtheorem{`*назва*`}[`*лічильник*`]{`*текст*`}[`*розділ*`]`

Аргумент *назва* — це коротке ключове слово, використовуване для ідентифікації «теореми». Аргументом *текст* ви визначаєте дійсну назву «теореми», під якою вона буде друкуватися в документі.

Аргументи в квадратних дужках необов'язкові. Вони використовуються для визначення того, як нумерувати «теорему». Аргументом *лічильник* ви може вказати *назву* попередньо визначеної «теореми». Нова «теорема» буде тоді нумеруватися в тій же послідовності. Аргумент *розділ* дозволяє вам указати розділ, в межах якого ви хочете нумерувати вашу «теорему».

Після використання в преамбулі документа команди `\newtheorem`, ви можете користуватися наступними командами:

`\begin{`*назва*`}[`*текст*`]`
Це — моя цікава теорема.
`\end{`*назва*`}`

На цьому теорії повинно бути досить. Подальші приклади повинні розвіяти останні сумніви, і остаточно переконати вас, що оточення `\newtheorem` занадто складне, щоб його можна було зрозуміти:

```
 % визначення для преамбули
 % документа
\newtheorem{law}{Law}
\newtheorem{jury}[law]{Jury}
 % в документі
\begin{law} \label{law:box}
Don't hide in the witness box
\end{law}
\begin{jury}[The Twelve]
It could be you! So beware and
see law \ref{law:box}\end{jury}
\begin{law}No, No, No\end{law}
```

Law 1 *Don't hide in the witness box*

Jury 2 (The Twelve) *It could be you! So beware and see law 1*

Law 3 *No, No, No*

Теорема «Jury» використовує той же лічильник, що і теорема «Law». Отже, вона отримає номер у послідовності з іншими теоремами «Law».

За допомогою аргумента у квадратних дужках можна вказати назву теореми, чи щось аналогічне.

```
\flushleft
\newtheorem{mur}{Мерфі}[section]
\begin{mur}
Якщо існує два і
більше способів зробити
щось, і один їх цих
способів може призвести
до катастрофи, то хтось
обов’язково це зробить.
\end{mur}
```

Мерфі 3.8.1 *Якщо існує два і більше способів зробити щось, і один їх цих способів може призвести до катастрофи, то хтось обов’язково це зробить.*

Теорема «Мерфі» отримує номер, пов’язаний з номером поточного розділу. Ви можете також використовувати іншу структурну одиницю, наприклад, частину чи підрозділ.

3.9 Жирні символи

У LaTeX досить непросто одержати жирні символи; це, ймовірно, зроблено навмисно, тому що непрофесіонали занадто часто зловживають ними. Команда зміни шрифту `\mathbf` дає напівжирні символи, але вони звичайні (прямі), тоді як математичні символи звичайно курсивні. Існує команда `\boldmath`, але *вона може використовуватися тільки поза математичним режимом.* Це ж стосується і символів.

```
\begin{displaymath}
\mu, M \qquad \mathbf{M} \qquad
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

$$\mu, M \qquad \mathbf{M} \qquad \boldsymbol{\mu, M}$$

Зверніть увагу, що кома теж жирна, що може бути небажаним.

Пакет amsbsy (включається пакетом amsmath), а також bm з набору tools спрощує це, надаючи команду `\boldsymbol`.

```
\begin{displaymath}
\mu, M \qquad
\boldsymbol{\mu}, \boldsymbol{M}
\end{displaymath}
```

$$\mu, M \qquad \boldsymbol{\mu}, \boldsymbol{M}$$

3.10 Список математичних символів

В наступних таблицях ви зможете знайти усі символи, доступні в *математичному режимі.*

Щоб використовувати символи, перелічені в таблицях 3.12–3.16[8], необхідно завантажений пакет amssymb в преамбулі документа, а також мати встановлені математичні шрифти AMS[9]. Більш вичерпний список символів можна знайти в `CTAN:/tex-archive/info/symbols/comprehensive`.

Табл. 3.1: Акценти в математичному режимі

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

$\hat{a}$

Табл. 3.2: Маленькі грецькі літери

α	`\alpha`	θ	`\theta`	o	`o`	υ	`\upsilon`
β	`\beta`	ϑ	`\vartheta`	π	`\pi`	ϕ	`\phi`
γ	`\gamma`	ι	`\iota`	ϖ	`\varpi`	φ	`\varphi`
δ	`\delta`	κ	`\kappa`	ρ	`\rho`	χ	`\chi`
ϵ	`\epsilon`	λ	`\lambda`	ϱ	`\varrho`	ψ	`\psi`
ε	`\varepsilon`	μ	`\mu`	σ	`\sigma`	ω	`\omega`
ζ	`\zeta`	ν	`\nu`	ς	`\varsigma`		
η	`\eta`	ξ	`\xi`	τ	`\tau`		

Табл. 3.3: Великі грецькі літери

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

[8]Ці таблиці були отримані з файла `symbols.tex` (автор — David Carlisle) і суттєво модифіковані, як порекомендував Josef Tkadlec.

[9]Доступні за адресою `CTAN:/tex-archive/fonts/amsfonts`.

Табл. 3.4: Бінарні відношення

Ви можете відтворити відповідні заперечення додаючи команду `\not` як префікс перед наступними символами.

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` or `\le`	$\geq$	`\geq` or `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$	`\sqsubset` [a]	$\sqsupset$	`\sqsupset` [a]	$\Join$	`\Join` [a]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` or `\ne`

[a] Для отримання цього символа скористайтеся пакетом latexsym

Табл. 3.5: Бінарні оператори

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

Табл. 3.6: Великі оператори

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

Табл. 3.7: Стрілки

$\leftarrow$	`\leftarrow` or `\gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\rightarrow$	`\rightarrow` or `\to`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\updownarrow$	`\updownarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\iff$	`\iff` (bigger spaces)	$\leadsto$	`\leadsto` [a]

[a]Для отримання цього символа скористайтеся пакетом latexsym

Табл. 3.8: Розділювачі

$($	`(`	$)$	`)`	$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`
$[$	`[` or `\lbrack`	$]$	`]` or `\rbrack`	$\downarrow$	`\downarrow`	$\Downarrow$	`\Downarrow`
$\{$	`\{` or `\lbrace`	$\}$	`\}` or `\rbrace`	$\updownarrow$	`\updownarrow`	$\Updownarrow$	`\Updownarrow`
$\langle$	`\langle`	$\rangle$	`\rangle`	$\vert$	`|` or `\vert`	$\Vert$	`|` or `\Vert`
$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`	$\lceil$	`\lceil`	$\rceil$	`\rceil`
$/$	`/`	$\backslash$	`\backslash`		. (dual. empty)		

Табл. 3.9: Великі розділювачі

$\lgroup$	`\lgroup`	$\rgroup$	`\rgroup`	$\lmoustache$	`\lmoustache`	$\rmoustache$	`\rmoustache`
$\arrowvert$	`\arrowvert`	$\Arrowvert$	`\Arrowvert`	$\bracevert$	`\bracevert`		

Табл. 3.10: Різні символи

$\dots$	\dots	$\cdots$	\cdots	$\vdots$	\vdots	$\ddots$	\ddots
$\hbar$	\hbar	$\imath$	\imath	$\jmath$	\jmath	ℓ	\ell
$\Re$	\Re	$\Im$	\Im	$\aleph$	\aleph	$\wp$	\wp
$\forall$	\forall	$\exists$	\exists	$\mho$	\mho [a]	∂	\partial
$'$	'	$\prime$	\prime	$\emptyset$	\emptyset	∞	\infty
∇	\nabla	$\triangle$	\triangle	$\Box$	\Box [a]	$\Diamond$	\Diamond [a]
$\bot$	\bot	$\top$	\top	$\angle$	\angle	$\surd$	\surd
$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\clubsuit$	\clubsuit	$\spadesuit$	\spadesuit
$\neg$	\neg or \lnot	$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp

[a] Для отримання цього символа скористайтеся пакетом latexsym

Табл. 3.11: Не-математичні символи

Ці символи можна використовувати також у текстовому режимі.

†	\dag	§	\S	©	\copyright	®	\textregistered
‡	\ddag	¶	\P	£	\pounds	%	\%

Табл. 3.12: Розділювачі $\mathcal{AMS}$

$\ulcorner$	\ulcorner	$\urcorner$	\urcorner	$\llcorner$	\llcorner	$\lrcorner$	\lrcorner
$\lvert$	\lvert	$\rvert$	\rvert	$\lVert$	\lVert	$\rVert$	\rVert

Табл. 3.13: Грецькі та іврит літери $\mathcal{AMS}$

$\digamma$	\digamma	$\varkappa$	\varkappa	$\beth$	\beth	$\gimel$	\gimel	$\daleth$	\daleth

Табл. 3.14: Бінарні відношення $\mathcal{AMS}$

$\lessdot$	\lessdot	$\gtrdot$	\gtrdot	$\doteqdot$	\doteqdot or \Doteq
$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\risingdotseq$	\risingdotseq
$\eqslantless$	\eqslantless	$\eqslantgtr$	\eqslantgtr	$\fallingdotseq$	\fallingdotseq
$\leqq$	\leqq	$\geqq$	\geqq	$\eqcirc$	\eqcirc
$\lll$	\lll або \llless	$\ggg$	\ggg або \gggtr	$\circeq$	\circeq
$\lesssim$	\lesssim	$\gtrsim$	\gtrsim	$\triangleq$	\triangleq
$\lessapprox$	\lessapprox	$\gtrapprox$	\gtrapprox	$\bumpeq$	\bumpeq
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\Bumpeq$	\Bumpeq
$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless	$\thicksim$	\thicksim
$\lesseqqgtr$	\lesseqqgtr	$\gtreqqless$	\gtreqqless	$\thickapprox$	\thickapprox
$\preccurlyeq$	\preccurlyeq	$\succcurlyeq$	\succcurlyeq	$\approxeq$	\approxeq
$\curlyeqprec$	\curlyeqprec	$\curlyeqsucc$	\curlyeqsucc	$\backsim$	\backsim
$\precsim$	\precsim	$\succsim$	\succsim	$\backsimeq$	\backsimeq
$\precapprox$	\precapprox	$\succapprox$	\succapprox	$\vDash$	\vDash
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\Vdash$	\Vdash
$\Subset$	\Subset	$\Supset$	\Supset	$\Vvdash$	\Vvdash
$\sqsubset$	\sqsubset	$\sqsupset$	\sqsupset	$\backepsilon$	\backepsilon
$\therefore$	\therefore	$\because$	\because	$\varpropto$	\varpropto
$\shortmid$	\shortmid	$\shortparallel$	\shortparallel	$\between$	\between
$\smallsmile$	\smallsmile	$\smallfrown$	\smallfrown	$\pitchfork$	\pitchfork
$\vartriangleleft$	\vartriangleleft	$\vartriangleright$	\vartriangleright	$\blacktriangleleft$	\blacktriangleleft
$\trianglelefteq$	\trianglelefteq	$\trianglerighteq$	\trianglerighteq	$\blacktriangleright$	\blacktriangleright

Табл. 3.15: Стрілки $\mathcal{AMS}$

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

Табл. 3.16: Заперечення бінарних відношень і стрілок $\mathcal{AMS}$

$\nless$	`\nless`	$\ngtr$	`\ngtr`	$\varsubsetneqq$	`\varsubsetneqq`
$\lneq$	`\lneq`	$\gneq$	`\gneq`	$\varsupsetneqq$	`\varsupsetneqq`
$\nleq$	`\nleq`	$\ngeq$	`\ngeq`	$\nsubseteqq$	`\nsubseteqq`
$\nleqslant$	`\nleqslant`	$\ngeqslant$	`\ngeqslant`	$\nsupseteqq$	`\nsupseteqq`
$\lneqq$	`\lneqq`	$\gneqq$	`\gneqq`	$\nmid$	`\nmid`
$\lvertneqq$	`\lvertneqq`	$\gvertneqq$	`\gvertneqq`	$\nparallel$	`\nparallel`
$\nleqq$	`\nleqq`	$\ngeqq$	`\ngeqq`	$\nshortmid$	`\nshortmid`
$\lnsim$	`\lnsim`	$\gnsim$	`\gnsim`	$\nshortparallel$	`\nshortparallel`
$\lnapprox$	`\lnapprox`	$\gnapprox$	`\gnapprox`	$\nsim$	`\nsim`
$\nprec$	`\nprec`	$\nsucc$	`\nsucc`	$\ncong$	`\ncong`
$\npreceq$	`\npreceq`	$\nsucceq$	`\nsucceq`	$\nvdash$	`\nvdash`
$\precneqq$	`\precneqq`	$\succneqq$	`\succneqq`	$\nvDash$	`\nvDash`
$\precnsim$	`\precnsim`	$\succnsim$	`\succnsim`	$\nVdash$	`\nVdash`
$\precnapprox$	`\precnapprox`	$\succnapprox$	`\succnapprox`	$\nVDash$	`\nVDash`
$\subsetneq$	`\subsetneq`	$\supsetneq$	`\supsetneq`	$\ntriangleleft$	`\ntriangleleft`
$\varsubsetneq$	`\varsubsetneq`	$\varsupsetneq$	`\varsupsetneq`	$\ntriangleright$	`\ntriangleright`
$\nsubseteq$	`\nsubseteq`	$\nsupseteq$	`\nsupseteq`	$\ntrianglelefteq$	`\ntrianglelefteq`
$\subsetneqq$	`\subsetneqq`	$\supsetneqq$	`\supsetneqq`	$\ntrianglerighteq$	`\ntrianglerighteq`
$\nleftarrow$	`\nleftarrow`	$\nrightarrow$	`\nrightarrow`	$\nleftrightarrow$	`\nleftrightarrow`
$\nLeftarrow$	`\nLeftarrow`	$\nRightarrow$	`\nRightarrow`	$\nLeftrightarrow$	`\nLeftrightarrow`

Табл. 3.17: Бінарні оператори $\mathcal{AMS}$

$\dotplus$	`\dotplus`	$\centerdot$	`\centerdot`	$\intercal$	`\intercal`
$\ltimes$	`\ltimes`	$\rtimes$	`\rtimes`	$\divideontimes$	`\divideontimes`
$\Cup$	`\Cup` or `\doublecup`	$\Cap$	`\Cap` or `\doublecap`	$\smallsetminus$	`\smallsetminus`
$\veebar$	`\veebar`	$\barwedge$	`\barwedge`	$\doublebarwedge$	`\doublebarwedge`
$\boxplus$	`\boxplus`	$\boxminus$	`\boxminus`	$\circleddash$	`\circleddash`
$\boxtimes$	`\boxtimes`	$\boxdot$	`\boxdot`	$\circledcirc$	`\circledcirc`
$\leftthreetimes$	`\leftthreetimes`	$\rightthreetimes$	`\rightthreetimes`	$\circledast$	`\circledast`
$\curlyvee$	`\curlyvee`	$\curlywedge$	`\curlywedge`		

Табл. 3.18: $\mathcal{AMS}$ різне

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

Табл. 3.19: Математичний алфавіт

Приклад	Команда	Необхідний пакет
ABCdef	`\mathrm{ABCdef}`	
ABCdef	`\mathit{ABCdef}`	
$\mathnormal{ABCdef}$	`\mathnormal{ABCdef}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	euscript з опцією `mathcal`
$\mathscr{ABC}$	`\mathscr{ABC}`	mathrsfs
$\mathfrak{ABCdef}$	`\mathfrak{ABCdef}`	eufrak
$\mathbb{ABC}$	`\mathbb{ABC}`	amsfonts або amssymb

Розділ 4

Корисні можливості

При створенні великих документів, ви можете скористатися деякими корисними можливостями LaTeX, такими, як генерація покажчиків, робота з бібліоґрафією, та іншими. Набагато детальніший опис можливостей та розширень LaTeX можна знайти в *LaTeX Manual* [2] та *The LaTeX Companion* [4].

4.1 Включення EPS графіки

LaTeX надає базові засоби для роботи з плаваючими об'єктами, такими, як зображення чи графіки, за допомогою оточень `figure` та `table`.

Існує також ряд можливостей створювати власне графіку за допомогою базового LaTeX або спеціальних пакетів. На жаль, більшість користувачів вважають ці способи занадто складним, і тому про них не йтиметься далі в цьому *Вступі*. Ви зможете знайти потрібну інформацію з цього приводу, звернувшись до *The LaTeX Companion* [4] або до *LaTeX Manual* [2].

Набагато простіший спосіб отримання графіки в документі — створити її за допомогою спеціалізованих програмних пакетів[1] і включити готові зображення в документ. Пакети LaTeX пропонують багато способів це зробити. У цьому *Вступі* ми висвітлимо тільки використання графіки у форматі Encapsulated PostScript (EPS), оскільки він досить простий і широко вживаний. Для того, щоб отримати зображення в форматі EPS, вам потрібен доступ до PostScript принтера[2].

Пакет graphicx (автор — D. P. Carlisle) надає гарний набір команд для включення графіки. Він є частиною цілого зібрання пакетів, що назива-

[1] Таких, як XFig, CorelDraw!, Freehand, Gnuplot, ...

[2] Інша можливість отримати PostScript — скористатися програмою GhostScript, що доступна за адресою `CTAN:/tex-archive/support/ghostscript`. Користувачів Windows та OS/2 може зацікавити GSview.

ється «graphics»[3].

Працюючи в системі з доступним для виводу PostScript принтером та встановленим пакетом graphicx, скористайтеся наступною інструкцією для включення зображення в документ:

1. Експортуйте зображення з вашої графічної програми в формат EPS[4].

2. Завантажте пакет graphicx у преамбулі вхідного файлу за допомогою команди

   `\usepackage[`*драйвер*`]{graphicx}`

   де *драйвер* — назва вашого конвертора «DVI в PostScript». Найбільш широко використовувана програма називається `dvips`. Назва драйвера потрібна, оскільки не існує стандарту на те, як включати графіку в TeX. Знаючи назву *драйвера*, пакет graphicx може вибрати правильний метод включення інформації про графіку в файл `.dvi`, так, щоб драйвер зрозумів це і міг правильно включити файл `.eps`.

3. Скористайтеся командою

   `\includegraphics[`*ключ*=*значення*, …`]{`*файл*`}`

   для включення *файлу* до вашого документа. Необов'язковим параметром може бути список *ключів* і їхніх *значень*, які повинні іти через кому. *Ключі* можуть слугувати для зміни ширини, висоти, та орієнтації включеної графіки. Таблиця 4.1 перелічує найбільш важливі ключі.

Табл. 4.1: Назви ключів пакета graphicx.

`width`	масштабує графіку до заданої ширини
`height`	масштабує графіку до заданої висоти
`angle`	повертає графіку проти годинникової стрілки
`scale`	масштабує графіку

[3] `CTAN:/tex-archive/macros/latex/required/graphics`

[4] Якщо програма не може експортувати в формат EPS, спробуйте встановити драйвер PostScript принтера (наприклад, Apple LaserWriter) і надрукуйте його у файл через цей драйвер. Якщо пощастить, ви отримаєте EPS файл. Пам'ятайте, EPS повинен містити не більше однієї сторінки. Драйвери деяких принтерів можна явно настроїти для створення файлів у форматі EPS.

Наступний приклад, сподіваюся, зробить це зрозумілішим:

```
\begin{figure}
\begin{center}
\includegraphics[angle=90, width=0.5\textwidth]{test}
\end{center}
\end{figure}
```

Зображення, що міститься у файлі `test.eps`, *спершу* повернуте на 90 градусів, а *потім* масштабоване до ширини, що дорівнює 0.5 ширини стандартного абзацу. Співвідношення сторін зберігається, оскільки висоту не задано. Ширина і висота можуть бути також задані в абсолютних одиницях. Дивіться таблицю 6.5 на сторінці 106 для більш детальної інформації. Якщо ви бажаєте знати про це більше, читайте [9] і [12].

4.2 Бібліоґрафія

За допомогою оточення `thebibliography` ви можете створити бібліоґрафію. Кожен елемент починається з

`\bibitem[`*label*`]{`*marker*`}` `\bibitem[`*мітка*`]{`*маркер*`}`

Маркер потім використовується для посилань на книгу чи статтю:

`\cite{`*маркер*`}`

Якщо не задати параметр *мітка*, елементи бібліоґрафії будуть пронумеровані автоматично. Параметр після команди `\begin{thebibliography}` встановлює максимальну ширину цього номера. В наступному прикладі `{99}` повідомляє LaTeX, що номер жодного з елементів бібліоґрафії не буде ширшим, ніж число «99».

```
Партл~\cite{pa} запропонував,
щоб \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Volume~9, Issue~1 (1988)

\end{thebibliography}
```

Партл [1] запропонував, щоб ...

Бібліоґрафія

[1] H. Partl: *German TEX*, TUGboat Volume 9, Issue 1 (1988) 65

Для роботи з бібліоґрафією більших проектів варто скористатися програмою BibTEX, що входить до більшості дистрибутивів TEX[5]. Він дозволяє підтримувати бібліоґрафічну базу даних і вибирати з неї описи документів, посилання на які є у вашому документі. Візуальна презентація бібліоґрафій, зґенерованих BibTEX'ом, базується на концепції стилів, що дозволяє створити бібліоґрафію у відповідності з певним стандартом чи дизайном.

4.3 Покажчики

Корисною особливістю багатьох книг є їхні предметні покажчики. Покажчики можуть бути створені автоматично за допомогою LaTEX і супутньої програми `makeindex`[6]. У цьому *Вступі* будуть викладені тільки базові команди для створення покажчика. Більш детальний опис можна знайти в *The LaTEX Companion* [4].

Для того, щоб LaTEX міг створювати покажчик, необхідно завантажити пакет makeidx:

```
\usepackage{makeidx}
```

та активізувати спеціальні команди індексування, помістивши

```
\makeindex
```

у преамбулу документа.

Вміст покажчика створюється командами

```
\index{ключ}
```

де *ключ* — елемент покажчика. Ви вводите команди створення покажчика у ті місця тексту, куди ці елементи повинні вказувати. Таблиця 4.2 пояснює синтаксис аргументу *ключ* на кількох прикладах.

Коли LaTEX обробляє вхідний файл, кожна команда `\index` записує відповідний елемент покажчика разом з номером поточної сторінки в спеціальний файл. Файл має ту ж назву, що і вхідний файл LaTEX, але інше розширення (`.idx`). Цей `.idx` файл потім може бути оброблений

[5] Для роботи з україномовною бібліоґрафією слід скористатися 8-бітним BibTEX'ом `bibtex8`, який також входить до більшості дистрибутивів TEX. (*Прим. пер.*)

[6] На системах, що не підтримують довгі назви файлів, програма може називатися `makeidx`.

Табл. 4.2: Приклади синтаксису ключів покажчика

Приклад	Елемент	Пояснення
`\index {hello}`	hello, 1	Звичайний елемент
`\index {hello!Peter}`	Peter, 3	Підпорядкований елемент
`\index {Sam@\textsl {Sam}}`	*Sam*, 2	Форматований ключ
`\index {Lin@\textbf {Lin}}`	**Lin**, 7	Те ж
`\index {Jenny|textbf}`	Jenny, **3**	Форматований номер сторінки
`\index {Joe|textit}`	Joe, *5*	Те ж
`\index {eolienne@\'eolienne}`	éolienne, 4	Обробка акцентів

програмою `makeindex`.

`makeindex` *назвафайлу*

Програма `makeindex`[7] генерує відсортований покажчик з тією ж назвою, але цього разу з розширенням `.ind`. Якщо тепер повторно обробити вхідний файл, цей відсортований покажчик включається в документ у те місце, де LaTeX знаходить команду

`\printindex`

Пакет showidx, що входить у LaTeX 2_ε, друкує всі елементи покажчика на лівому полі тексту. Це дуже корисно при перевірці тексту і звірці покажчика.

Пам'ятайте, що команда `\index` може негативно вплинути на вигляд документа, так що користуйтеся нею обережно.

```
Моє Слово \index{слово}. А тепер
Слово\index{слово}. Зверніть увагу
на розміщерння крапки.
```

Моє Слово . А тепер Слово. Зверніть увагу на розміщерння крапки.

4.4 Вибагливі колонтитули

Пакет fancyhdr[8] (автор — Piet van Oostrum) надає кілька простих команд, що дозволяють вам налаштовувати верхні і нижні колонтитули доку-

[7] Оскільки українські літери пишуться в файл у вигляді команд типу `\IeC {\cyrp }`, для створення покажчиків україномовних документів потрібно користуватися програмою `uamakeindex`. (*Прим. пер.*)

[8] Доступний з `CTAN:/tex-archive/macros/latex/contrib/supported/fancyhdr`

мента. Якщо ви зараз глянете на верх цієї сторінки, то побачите одне з можливих застосувань цього пакета.

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
 % цим ми забезпечуємо вивід заголовків розділів
 % і підрозділів маленькими літерами
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}  % вилучаємо поточні установки для колонтитулів
\fancyhead[LE,RO]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\addtolength{\headheight}{0.5pt} % залишаємо місце для лінійки
\fancypagestyle{plain}{%
   \fancyhead{} % вилучаємо колонтитули на звичайних сторінках
   \renewcommand{\headrulewidth}{0pt} % і лінійку
}
```

Рис. 4.1: Приклад налаштування fancyhdr

При налаштуванні колонтитулів хитрість полягає у тім, щоб включити туди такі речі, як заголовки поточних розділу чи підрозділу. LaTeX досягає цього в два етапи. У визначеннях колонтитулів можна використовувати команди `\rightmark` і `\leftmark`, щоб відобразити заголовки поточного розділу чи підрозділу, відповідно. Значення цих двох команд міняються при обробці команд `\chapter` чи `\section`.

Для максимальної гнучкості команди `\chapter` і подібні не перевизначають `\rightmark` і `\leftmark` самі, а викликають іншу команду, що називається `\chaptermark`, `\sectionmark` чи `\subsectionmark`, відповідальну за перевизначення `\rightmark` і `\leftmark`.

Якщо потрібно змінити вигляд назви розділу у верхньому колонтитулі, просто «обновіть» команду `\chaptermark`.

Рисунок 4.1 показує можливий варіант настройки пакета fancyhdr, що створює майже такі колонтитули, як у цьому буклеті. У будь-якому випадку я рекомендую вам ознайомитися з документацією пакета за адресою, згаданою у виносці.

4.5 Пакет verbatim

Раніше в цьому *Вступі* ви ознайомилися з *оточенням* `verbatim`. У цьому підрозділі ви довідаєтеся про *пакет* verbatim. Пакет verbatim являє собою повторну реалізацію оточення `verbatim`, що обходить деякі з його обмежень. Саме по собі це не так вже й захоплююче, але, крім того, у нього була додана деяка нова функціональність, і саме з цієї причини я згадую тут цей пакет. Пакет verbatim надає команду

`\verbatiminput{`*назвафайлу*`}`

яка дозволяє вам включати текстовий файл у ваш документ так, наче його вміст знаходиться усередині оточення `verbatim`.

Оскільки пакет verbatim входить у комплект «tools», ви знайдете його установленим на більшості систем. Якщо ви хочете довідатися про нього більше, обов'язково прочитайте [10].

4.6 Звантаження і встановлення пакетів LaTeX

Більшість дистрибутивів LaTeX містить велику кількість стильових пакетів, але ще більше доступно в мережі. Основним місцем, де їх потрібно шукати в Інтернеті, є CTAN (`http://www.ctan.org/`).

Дистрибутиви таких пакетів, як geometry, hyphenat, та багатьох інших, типово складаються з двох файлів: файлу з розширенням `.ins`, та іншого — з розширенням `.dtx`. Часто там буде ще й `readme.txt` з коротким описом пакета. Звичайно, його слід читати першим.

У будь-якому випадку, коли ви маєте копію файлів пакета на комп'ютері, вам ще потрібно обробити їх таким чином, щоб (а) повідомити ваш дистрибутив TeX, де знаходиться новий стильовий пакет, і (б) отримати документацію. Ось як виконати першу частину:

1. Виконайте LaTeX з файлом `.ins`. Це згенерує файл `.sty`.
2. Перемістіть файл `.sty` у місце, де ваш дистрибутив зможе його знайти. Як правило, це у підкаталозі `.../`*`localtexmf`*`/tex/latex` (користувачі Windows чи OS/2 можуть змінити напрям похилих рисок).
3. Поновити базу даних назв файлів вашого дистрибутива. Команда залежить від дистрибутива LaTeX, яким ви користуєтесь:
   teTeX, fpTeX — `maktexlsr` або `texhash`;
   web2c — `maktexlsr`;
   MikTeX — `initexmf -update-fndb` або скористайтеся GUI.

Тепер з файла `.dtx` можна згенерувати документацію:

1. Виконайте LaTeX з файлом `.dtx`. Це згенерує файл `.dvi`. Пам'ятайте, можливо доведеться виконувати LaTeX кілька разів, щоб отримати правильні перехресні посилання.
2. Перевірте, чи з'явився серед інших файлів файл `.idx`. Якщо ні — можете перейти до кроку 5.
3. Щоб отримати покажчик, виконайте наступне:

   ```
   makeindex -s gind.ist назва
   ```

   де *назва* — назва основного файла (без розширення).
4. Виконайте LaTeX з файлом `.dtx` ще раз.
5. Нарешті, створіть файл `.ps` або `.pdf` для отримання більшого задоволення від читання.

Може трапитися, що ви виявите файл `.glo`. Виконайте наступну команду між кроками 4 і 5:

```
makeindex -s gglo.ist -o назва.gls назва.glo
```

Не забудьте виконати LaTeX з файлом `.dtx` ще один раз перед переходом до кроку 5.

4.7 Робота з pdfLaTeX

By Daniel Flipo <Daniel.Flipo@univ-lille1.fr>

Документи у форматі PDF є гіпертекстовим. Як і на веб сторінці, деякі слова в документі є гіперпосиланнями на інші місця документа, або навіть на інші документи. Якщо клацнути мишкою на таке гіперпосилання, ви потрапите туди, куди воно вказує. У контексті LaTeX це значить, що усі `\ref` та `\pageref` стають гіперпосиланнями. Крім того, зміст, предметний покажчик, та інші аналогічні структури стають зібраннями гіперпосилань.

Більшість веб сторінок зараз створені у HTML *(HyperText Markup Language)*. Цей формат має кілька суттєвих недоліків при створенні наукових документів.

1. Включення математичних формул у документи HTML не підтримується. Хоча для цього існує стандарт, більшість переглядачів, що використовуються нині, не підтримують його або не мають відповідних шрифтів.

2. Вивід на друк HTML документів можливий, але результат відрізняється для різних платформ і переглядачів, та й якість далека від тієї, до якої звикли у світі LaTeX.

Було багато спроб створити транслятори з LaTeX в HTML. Деякі були досить успішними у тому сенсі, що вони можуть створювати правильні веб сторінки із стандартних вхідних файлів LaTeX. Але усі вони, щоб виконати роботу, рубають кути наліво і направо. Як тільки ви почнете використовувати більш складні властивості LaTeX та зовнішні пакети, документ перестає триматися купи. Автори, що бажають зберегти унікальні типографські властивості їхніх документів навіть після публікації в тенетах, звертаються до PDF *(Portable Document Format)*, який зберігає макет документа і дозволяє навігацію за допомогою гіперпосилань. Більшість сучасних переглядачів мають модуль, що дозволяє безпосередній перегляд документів PDF.

Незважаючи на те, що переглядачі для DVI та PS файлів існують практично для усіх платформ, програми Adobe Acrobat Reader та xpdf, призначені для перегляду документів у форматі PDF, є набагато більш поширені. Тому надання PDF версії документа зробить його більш доступним для потенційного читача.

4.7.1 Документи PDF для Тенет

Створити файл PDF з вихідного тексту LaTeX дуже просто завдяки програмі pdfTeX, автором якої є Hàn Thế Thành. PdfTeX створює файл у форматі PDF так само, як TeX створює файл у форматі DVI. Існує також pdfLaTeX, який створює PDF з вихідного тексту LaTeX.

Обидва, pdfTeX і pdfLaTeX, входить у більшість сучасних дистрибутивів TeX, таких, як teTeX, fpTeX, MikTeX, TeXLive та CMacTeX.

Для того, щоб створити PDF замість DVI, достатньо замість команди `latex file.tex` виконати `pdflatex file.tex`. В системах, де LaTeX не викликається з командного рядка, потрібно знайти спеціальну кнопку в TeXControlCenter.

В LaTeX параметри сторінки можна задавати за допомогою необов'язкового аргументу команди `\documentclass`, наприклад `a4paper` або `letterpaper`. Це працює і в pdfLaTeX, але крім цього pdfTeX потребує знати фізичні розміри аркуша паперу. Якщо ви користуєтесь пакетом hyperref (див. стор. 74), розміри аркуша будуть встановлені автоматично. В іншому випадку це можна зробити вручну, помістивши в преамбулу документа наступні рядки:

```
\pdfpagewidth=\paperwidth
\pdfpageheight=\paperheight
```

Наступні розділи розглянуть особливості pdfLaTeX у порівнянні з LaTeX детальніше. Основні розбіжності стосуються трьох областей: використання шрифтів, формат зображень для включення у документ, і ручну конфігурацію гіперпосилань.

4.7.2 Шрифти

PdfLaTeX може працювати з багатьма типами шрифтів (растровими PK, векторними TrueType та PostScript type 1…), але основний тип шрифтів TeX, растрові PK шрифти, виглядають непривабливо, коли документ переглядати за допомогою Acrobat Reader'a. Для того, щоб створити документ, який би гарно відображувався на екрані, краще використовувати виключно шрифти PostScript Type 1.

Шрифти Computer Modern та AMSFonts були конвертовані у формат PostScript Type 1 фірмами Blue Sky Research та Y&Y, Inc., які передали права інтелектуальної власності Американському Математичному Товариству. Ці шрифти передані у вільне користування на початку 1997 року і зараз розповсюджуються з більшістю дистрибутивів TeX.

Проте, якщо використовувати pdfLaTeX для створення документів мовами, іншими ніж англійська, потрібні шрифти EC, LH, або CB (дивіться обговорення шрифтів `OT1` на ст. 27). Володимир Волович створив набір Type 1 шрифтів `cm-super`, який охоплює шрифти EC/TC, EC Concrete, EC Bright та LH. Він доступний на `CTAN:/tex-archive//fonts/ps-type1/cm-super` і входить в TeXLive7 та MikTeX. Подібний набір type 1 шрифтів для CB Greek (створений Апостолосом Сирополосом) доступний на `CTAN:/tex-archive/fonts/gree` Нажаль, обидва ці набори дещо гіршої типографської якості ніж шрифти Type 1 CM створені Blue Sky/Y&Y. Вони хінтовані автоматично, тому документ може виглядати на екрані не так ідеально, як із шрифтами від Blue Sky/Y&Y. Проте, на пристроях високої розподільчої здатності (наприклад, лазернизх принтерах) вони дають результат ідентичний до результату оригінальних растрових шрифтів EC/LH/CB.

Якщо ви створюєте документи однією з мов, що використовує латиницю, є ще декілька рішень.

- Можна скористатися пакетом aeguill, або *Almost European Computer Modern with Guillemets*. Додайте рядок `\usepackage{aeguill}` в преамбулу вашого документа, це дозволить використання віртуальних шрифтів AE замість шрифтів EC.

- Можна скористатися пакетом mltex, але це працюватиме тільки тоді, коли pdfTeX був компільований з опцією mltex.

Набір віртуальних шрифтів AE, як і система MlTeX, переконує TeX, що він працює з повним набором літер у кодуванні `T1`, створюючи більшість літер, яких бракує в CM шрифтах, за допомогою комбінації існуючих літер і акцентів. Це дозволяє використовувати високоякісні CM шрифти у форматі type 1, що доступні на більшості систем. Це надає велику перевагу для переносів у європейських мовах, що використовують латиницю. Єдиним недоліком є те що функція пошуку Acrobat Reader'а не працюють з штучними символами AE шрифту, тому слова з акцентованими літерами неможливо знайти у PDF файлі.

Для російської мови аналогічним рішенням є використання віртуальних шрифтів C1, доступних за адресою `ftp://ftp.vsu.ru/pub/tex/font-packs/c1fonts`. Вони комбінують стандартні CM шрифти з колекції Bluesky та шрифти CMCYR з наборів Paradissa і BaKoMa, усі вони доступні на CTAN у форматі type 1. Оскільки шрифти Paradissa містять тільки російські літери, шрифти C1 не мають інших літер кирилиці.

Альтернативним рішенням може бути використання інших родин шрифтів у форматі type 1. Деякі з них навіть розповсюджуються з кожною копією Acrobat Reader'а.

Два готових набори шрифтів доступні за допомогою пакетів pxfonts, де основним шрифтом є *Palatino*, та txfonts, де основним шрифтом є *Times*. Для того, щоб задіяти їх, додайте наступні рядки до преамбули документа:

```
\usepackage[T1]{fontenc}
\usepackage{pxfonts}
```

Зверніть увагу, ви можете зустріти повідомлення на зразок

```
Warning: pdftex (file eurmo10): Font eurmo10 at ... not found
```

у `log` файлі. Це значить, що pdfTeX не знайшов деякі шрифти. Якщо не вирішити цю проблему, сторінки PDF документа, де використовуються не знайдені шрифти, не будуть відображені.

Вся ця справа з шрифтами, особливо брак високоякісних шрифтів EC, бентежить думки багатьох людей, так що весь час знаходяться нові рішення.

4.7.3 Використання графіки

Найкраще включати графіку в LaTeX документ за допомогою пакета graphicx (див. ст. 63). Якщо вказати опцію `pdftex`, пакет буде працювати з pdfLaTeX:

```
\usepackage[pdftex]{color,graphicx}
```

У наведеному вище прикладі завантажено також пакет color, оскільки використання кольорів є природнім у документах, призначених для Тенет.

Поки що все добре. Поганою новиною є те, що EPS (Encapsulated PostScript) не підходить для pdfLaTeX. Якщо команді `\includegraphics` не вказати розширення файла, пакет graphicx буде шукати файли придатних форматів в залежності від опції *драйвер*. Для `pdftex` такими форматами є `.png`, `.pdf`, `.jpg`, `.mps` (METAPOST), та `.tif` — але *не* `.eps`.

Найпростішим способом вирішення цієї проблеми є конвертування EPS файлів у PDF формат за допомогою програми `epstopdf`, яка доступна в багатьох операційних системах. Для векторної графіки це прекрасне рішення. Для растрових зображень (фотографії, скани) це не найкращий вихід, оскільки формат PDF підтримує безпосереднє включення зображень у форматах PNG та JPEG. PNG підходить для зображень екрана (screenshots) та інших зображень з невеликою кількістю кольорів. JPEG зручний для фотографій, оскільки він дуже економічний.

Іноді буває доцільним не просто малювати певні геометричні фігури, а скоріше описати їх за допомогою спеціалізованої командної мови. Зверніть увагу на METAPOST, який можна знайти у більшості дистрибутивів TeX, разом з власним детальним посібником.

4.7.4 Гіпертекстові посилання

Пакет hyperref подбає про перетворення усіх внутрішніх посилань вашого документа у гіперпосилання. Для того, щоб це запрацювало, потрібно трішки почаклувати: цей пакет тому слід додавати *останнім* в преамбулі вашого документа.

Існує два способи налаштування поведінки пакета hyperref:

- за допомогою опцій, вказаних через кому в квадратних дужках
  `\usepackage[`*опція1,опція2,. . .*`]{hyperref}`

- або за допомогою спеціальної команди
  `\hypersetup{`*опція1,опція2,. . .*`}`
  яку можна використовувати кілька разів.

Єдиною обов'язковою опцією є `pdftex`; інші дозволяють змінити стандартну поведінку hyperref[9]. В наступному списку опцій стандартні значення вказані прямим шрифтом.

`bookmarks (=true,`*`false`*`)` вказує, показати чи сховати закладки при відкритті документа

`pdftoolbar (=true,`*`false`*`)` показати чи сховати лінійку інструментів Acrobat'a

`pdfmenubar (=true,`*`false`*`)` показати чи сховати меню Acrobat'a

[9] Пакет hyperref працює не тільки з pdfTeX. Його можна налаштувати так, що в DVI вивід звичайного LaTeX буде вбудовано PDF-специфічну інформацію, яку буде передано в PS файл за допомогою `dvips`, а потім використано Adobe Distiller'ом за допомогою якого PS файл може бути перетворено в PDF.

`pdffitwindow` (=true,*false*) чи масштабувати сторінку до розміру вікна при відкритті документа

`pdftitle` (={текст}) визначає назву документа, яка буде відображена у вікні `Document Info` Acrobat'a

`pdfauthor` (={текст}) Автор PDF документа

`pdfnewwindow` (=true,*false*) визначає, чи відкривати нове вікно, коли активізувати гіперпосилання, що веде за межі поточного документа

`colorlinks` (=false,*true*) показувати «зони» посилань кольором (`true`) чи рамками (`false`). Кольори можуть бути налаштовані за допомогою наступних опцій:

- `linkcolor` (=red) колір внутрішніх посилань (розділи, сторінки і т. ін.)
- `citecolor` (=green) колір посилань на бібліографію
- `filecolor` (=magenta) колір посилань на файли
- `urlcolor` (=cyan) колір url посилань (e-mail, web)

Якщо вас задовольняють стандартні значення, використовуйте

```
\usepackage[pdftex]{hyperref}
```

Для того, щоб отримати відкритий список закладок та кольорові посилання (вказувати значення `=true` не обов'язково):

```
\usepackage[pdftex,bookmarks,colorlinks]{hyperref}
```

Якщо документ створюється для друку, кольорові посилання — не дуже гарна ідея, оскільки вони будуть сірими і погано читаються (якщо ви користуєтеся чорно-білим принтером), тому можна скористатися кольоровими рамками, які не друкуються:

```
\usepackage{hyperref}
\hypersetup{colorlinks=false,pdftex}
```

або зробити посилання чорними:

```
\usepackage{hyperref}
\hypersetup{colorlinks,%
            citecolor=black,%
            filecolor=black,%
            linkcolor=black,%
            urlcolor=black,%
            pdftex}
```

Якщо ви хочете надати інформацію для секції `Document Info` PDF файла:

```
\usepackage[pdfauthor={Pierre Desproges}%
            pdftitle={Des femmes qui tombent},%
            pdftex]{hyperref}
```

Крім гіперпосилань, що створюються автоматично для перехресних посилань документа, можна створити додаткові гіперпосилання, явно вказавши

`\href{`*url*`}{`*text*`}`

Наприклад,

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

The CTAN website.

Клацніть мишкою на слові «CTAN» і ви потрапите на CTAN.

Якщо місце призначення посилання не URL а локальний файл, скористайтеся командою `\href`:

```
Повний документ знаходиться
\href{manual.pdf}{тут}
```

Повний документ знаходиться тут

Якщо клацнути мишкою на слові «тут», повинен відкритися файл `manual.pdf`. (Ім'я файла відносно розташування поточного документа).

Автор статті можливо захоче, щоб читачі могли легко надіслати електронного листа за допомогою команди `\href` всередині команди `\author` на титульній сторінці документа:

```
\author{Mary Oetiker $<$\href{mailto:mary@oetiker.ch}%
              {mary@oetiker.ch}$>$
```

Зверніть увагу, посилання розміщено так, що моя електронна адреса з'являється не тільки в гіперпосиланні, а і власне на сторінці. Це зроблено тому, що посилання `\href{mailto:mary@oetiker.ch}{Mary Oetiker}` буде гарно працювати в Acrobat, але коли сторінку роздрукувати, електронна адреса вже не буде доступна.

4.7.5 Проблеми з посиланнями

Іноді ви можете отримати повідомлення на зразок:

```
! pdfTeX warning (ext4): destination with the same identifier
  (name{page.1}) has been already used, duplicate ignored
```

коли рахівник обнуляється, наприклад, команда `\mainmatter` класу документів `book` обнуляє рахівник номерів сторінок перед першим розділом книги. Але вступ також має сторінку номер 1, тому усі посилання на «сторінку 1» більше не будуть унікальними, звідси і повідомлення, що «`дуплікати` були `проігноровані`».

Ця проблема «лікується» додаванням `plainpages=false` в опції пакета hyperref. Це допоможе з лічильником сторінок. Більш радикальною мірою є використання опції `hypertexnames=false`, але тоді посилання на сторінки перестануть працювати.

4.7.6 Проблеми з закладками

Текст, відображений в закладках, не завжди виглядає як очікувалось. Закладки — це «просто текст», тому там доступно набагато менше символів, ніж у нормальному тексті LaTeX. Hyperref як правило реагує на такі проблеми попередженнями:

```
Package hyperref Warning:
Token not allowed in a PDFDocEncoded string:
```

Обійти цю проблему можна надавши текст для закладки, на заміну тексту-«порушника»:

`\texorpdfstring{`*Текст TEX*`}{`*Текст для закладки*`}`

Найчастіше джерелом такого роду проблем є математичні вирази:

```
\section{\texorpdfstring{$E=mc^2$}%
        {E\ =\ mc\texttwosuperior}}
```

що перетворить `\section{$E=mc^2$}` на «E=mc2» в тексті закладок.

Зміна кольорів також не переносить подорожі в закладки:

```
\section{\textcolor{red}{Red !}}
```

дасть рядок «redRed!». Команда `\textcolor` ігнорується, а її аргумент (red) потрапить у закладку.

Якщо використати

```
\section{\texorpdfstring{\textcolor{red}{Red !}}{Red\ !}}
```

результат буде набагато більш прийнятним.

4.7.7 Сумісність джерельних текстів між LaTeX та pdfLaTeX

В ідеалі ваш документ повинен успішно компілюватися як LaTeX'ом, так і pdfLaTeX'ом. Головною проблемою є включення графіки. Простим рішенням є *систематично пропускати* розширення файлів у аргументі команд `\includegraphics`. Вони будуть шукати файли придатних форматів у поточному каталозі. Усе що потрібно, це створити відповідні версії файлів зображень. LaTeX буде шукати `.eps`, а pdfLaTeX буде намагатися включити файли з розширеннями `.png`, `.pdf`, `.jpg`, `.mps` або `.tif` (у цьому порядку).

Для випадків, коли бажано використовувати інший код для PDF версії документа, можна додати:

```
\newif\ifPDF
\ifx\pdfoutput\undefined\PDFfalse
\else\ifnum\pdfoutput > 0\PDFtrue
     \else\PDFfalse
     \fi
\fi
```

на самому початку документа. Це визначить спеціальну команду, яка дозволить легко писати умовний код:

```
\ifPDF
  \usepackage[T1]{fontenc}
  \usepackage{aeguill}
  \usepackage[pdftex]{graphicx,color}
  \usepackage[pdftex]{hyperref}
\else
  \usepackage[T1]{fontenc}
  \usepackage[dvips]{graphicx}
  \usepackage[dvips]{hyperref}
\fi
```

У наведеному вище прикладі пакет hyperref включено навіть для не-PDF версії. Ефектом цього є здатність команди `\href` працювати у всіх випадках, що дозволить уникнути потреби писати умовні оператори в кожному випадку появи цієї команди.

Зверніть увагу, що в останніх дистрибутивах TeX, наприклад, TeXLive, вибір між `pdftex` і `dvips` при виклику пакетів graphicx та color здійснюється автоматично у файлах конфігурації `graphics.cfg` та `color.cfg`.

4.8 Створення презентацій за допомогою pdfscreen

By Daniel Flipo <Daniel.Flipo@univ-lille1.fr>

Результати науковою роботи можна презентувати на дошці, за допомогою проектора і прозорих плівок, або прямо з лептопа за допомогою програми для презентацій.

PdfLaTeX у поєднанні з пакетом pdfscreen дозволяє створювати презентації у PDF, такі ж різнокольорові та яскраві, як і в *PowerPoint*, але набагато менш залежні від платформи, оскільки Acrobat Reader доступний на більшій кількості платформ та операційних систем.

Клас pdfscreen використовує пакети graphicx, color та hyperref з опціями, адаптованими для презентацій на екрані.

```
\documentclass[pdftex,12pt]{article}
%%% misc extensions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\usepackage[koi8u]{inputenc}
\usepackage[ukrainian]{babel}
\usepackage[T2A]{fontenc}
%%% pdfscreen %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\usepackage[screen,panelleft,chocolate]{pdfscreen}
% Формат екрана
\panelwidth=25mm
%%          висота, ширина
\screensize{150mm}{200mm}
%%          право, ліво, верх, низ
\marginsize{42mm}{8mm}{10mm}{10mm}
% Колір або зображення для тла
\overlayempty
\definecolor{mybg}{rgb}{1,0.9,0.7}
\backgroundcolor{mybg}
% Лого
\emblema{mylogo}
%%% для пост-процесора PPower4 %%%%%%%%%%%%%%%%%%%%%%%
\usepackage{pause}
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{document}
\begin{slide}
\begin{itemize}
\item Добра новина \ldots \pause
\item Погана новина
\end{itemize}
\end{slide}
\end{document}
```

Рис. 4.2: Приклад вхідного файла для pdfscreen

Для створення документів такого типу зазвичай використовують клас `article`. На рис. 4.2 показано приклад вхідного файла. Спершу потрібно завантажити пакет pdfscreen з відповідними опціями:

`screen` презентація для екрану. Скористайтесь опцією `print` для створення версії для друку.

`panelright` створює панель навігації з правого боку екрану. Щоб панель була з лівого боку, скористайтесь опцією `panelleft`, а якщо ви не бажаєте навігаційної панелі — `nopanel`.

`french` французька, або інша мова[10], підтримувана пакетом, для виводу текстів на кнопках навігаційної панелі відповідним чином. Ця опція незалежна від опцій пакета babel.

`chocolate` схема кольорів навігаційної панелі. Іншими опціями є `gray`, `orange`, `palegreen`, `bluelace`, та `blue`, яка є опцією за замовчуванням.

Потім потрібно налаштувати формат для показу на екрані. Оскільки презентація завжди буде масштабована до реального розміру екрана, формат сторінки можна використати для конфігурації відносного розміру шрифтів:

`\panelwidth` визначає ширину панелі навігації.

`\screensize{`*width*`}{`*height*`}` визначає висоту і ширину екрана включаючи панель навігації.

`\marginsize{`*left*`}{`*right*`}{`*top*`}{`*bottom*`}` визначає поля документа. У прикладі документ не центрований, оскільки номери розділів розміщені на лівому полі.

Використовуючи команду

`\overlay{`*зображення*`}`

можна задати зображення для тла у будь-якому підтримуваному pdfTEX'ом форматі зображень, або, якщо вам більше подобається просте тло, його колір можна задати командою

`\background{`*колір*`}`

Нарешті, якщо ви бажаєте розмістити логотип вашої організації на панелі навігації, скористайтеся командою

`\emblema{`*лого*`}`

де *лого* — назва файла з логотипом.

[10] На даний момент pdfscreen не підтримує української мови, українізація може бути зроблена за допомогою файла `pdfscreen.cfg` (*Прим. пер.*)

Якщо ви вважаєте, що ваша презентація виграє від послідовного показу елементів, скористайтеся пакетом pause. Він надає команду `\pause`, яку можна використовувати в тих місцях документа, де ви бажаєте, щоб Acrobat зробив паузу при показі сторінки. Пакет pause є частиною системи `ppower4` (P^4: *Pdf Presentation Post-Processor*), яка призначена для обробки `.pdf` виводу pdfTEX, щоб примусити його танцювати, співати, і просити їсти. Вивід pdfTEX можна обробити пропустивши його через постпроцесор:

```
ppower4 xy.pdf xyz.pdf
```

де xy.pdf — вхідний файл, а xyz.pdf — вихідний файл.

Для управління вмістом окремого слайду слід користуватися оточенням `slide`. Вміст кожного слайду буде центровано вертикально на сторінці.

Якщо ви спробуєте відкомпілювати попередній приклад, ви отримаєте повідомлення про помилку:

```
! pdfTeX warning (dest): name{contents} has been
  referenced but does not exist, replaced by a fixed one
```

Це тому, що на панелі є кнопка, що вказує на зміст, а оскільки приклад не містить команди `\tableofcontents`, і, відповідно, змісту, pdfTEX не може вирішити перехресних посилань.

Якщо ви бажаєте мати зміст прямо на панелі навігації, при завантаженні pdfscreen можна використати опцію `paneltoc`. Це дасть задовільний результат якщо у змісті презентації кілька коротких елементів. Можна задати скорочені варіанти заголовків розділів у квадратних дужках.

Цей короткий вступ тільки торкається поверхні можливостей pdfscreen та `PPower4`. Обидва мають власну детальну документацію.

Розділ 5

Створення графіки для математики

Більшість людей використовують LaTeX для набору текстів. Але оскільки структурно орієнтований підхід до створення документів є дуже зручним для авторів, LaTeX надає також певні (дещо обмежені) можливості створювати зображення за допомогою текстового опису. Крім того, було створено ряд розширень LaTeX для того щоб обійти ці обмеження. Ви дізнаєтеся про деякі з них у цьому розділі.

5.1 Огляд

Оточення `picture` дозволяє програмувати малюнки прямо в LaTeX. Детальний опис можна знайти в *LaTeX Manual* [2]. Це оточення має серйозні обмеження щодо кутів нахилу відрізків а також радіусів кіл, які можуть приймати тільки певні значення. З іншого боку, оточення `picture` надає команду `\qbezier` для створення квадратичних кривих Безьє. Багато які з часто використовуваних кривих, наприклад кола, еліпси, або ланцюгові криві можна з прийнятним наближенням описати за допомогою квадратичних кривих Безьє, хоча для цього можуть знадобитися певні математичні зусилля. Крім того, коли для генерування блоків `\qbezier` у вхідних файлах LaTeX користуватися мовами програмування, наприклад, Java, це оточення стає досить потужним інструментом.

Хоча програмування зображень прямо в LaTeX має суттєві обмеження і може бути навіть набридливим, існує ряд причин для використання цього методу. Документи створені таким чином мають менший розмір, крім того не потрібні додаткові графічні файли.

Такі пакети, як epic та eepic (описані, наприклад, в *The LaTeX Companion* [4]), або pstricks допомагають усунути обмеження оригінального оточення `picture`, і суттєво розширюють графічні можливості LaTeX.

В той час, як перші два пакети просто розширюють можливості оточення `picture`, пакет pstricks має своє власне оточення для малювання, `pspicture`. Потужність pstricks криється у широкому використанні можливостей PostScript. Крім того, написано ряд пакетів для специфічних цілей. Один з них — XY-pic, описаний в кінці цього розділу. Ряд таких пакетів описано детально в *The LaTeX Graphics Companion* [1] (не плутати з *The LaTeX Companion* [4]).

Напевне найпотужнішим з пов'язаних з LaTeX графічних інструментів є METAPOST, близнюк Кнутівського METAFONT'а. METAPOST використовує дуже потужну і математично витончену мову програмування METAFONT. На відміну від METAFONT, який генерує растрові зображення, METAPOST створює файли у форматі Encapsulated PostScript, які можна імпортувати в LaTeX. Для більш детального знайомства дивіться *A User's Manual for MetaPost* [15] або підручник на [17].

Дуже детальне обговорення стратегій використання графіки (і шрифтів) в LaTeX і TeX можна знайти в *TeX Unbound* [16].

5.2 Оточення `picture`

By Urs Oswald <osurs@bluewin.ch>

5.2.1 Базові команди

Оточення `picture` можна створити за допомогою однієї з двох команд

`\begin{picture}(`x, y`)...\end{picture}`

або

`\begin{picture}(`x, y`)(`x_0, y_0`)...\end{picture}`

Розміри x, y, початкові координати x_0, y_0, а також всі координати і розміри всередині оточення даються в одиницях `\unitlength`, які можуть бути перевизначені (але не всередині оточення `picture`) за допомогою команди

`\setlength{\unitlength}{1.2cm}`

Початковим значенням `\unitlength` є `1pt`. Перша пара, (x, y), резервує всередині документа прямокутний простір для малюнка. Необов'язкова друга пара, (x_0, y_0), задає довільні початкові координати (координати лівого нижнього кута прямокутника).

Більшість команд малювання мають одну або дві форми

$\boxed{\texttt{\textbackslash put(}x,y\texttt{)\{}\textit{об'єкт}\texttt{\}}}$

або

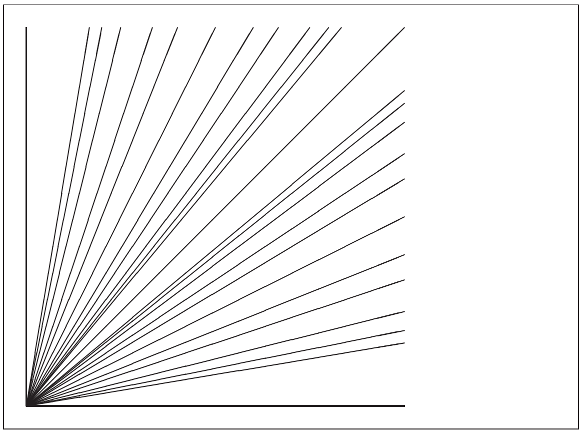
$\boxed{\texttt{\textbackslash multiput(}x,y\texttt{)(}\Delta x,\Delta y\texttt{)\{}n\texttt{\}\{}\textit{об'єкт}\texttt{\}}}$

Виключенням є криві Безьє. Їх малюють за допомогою команди

$\boxed{\texttt{\textbackslash qbezier(}x_1,y_1\texttt{)(}x_2,y_2\texttt{)(}x_3,y_3\texttt{)}}$

5.2.2 Відрізки

```
\setlength{\unitlength}{5cm}
\begin{picture}(1,1)
  \put(0,0){\line(0,1){1}}
  \put(0,0){\line(1,0){1}}
  \put(0,0){\line(1,1){1}}
  \put(0,0){\line(1,2){.5}}
  \put(0,0){\line(1,3){.3333}}
  \put(0,0){\line(1,4){.25}}
  \put(0,0){\line(1,5){.2}}
  \put(0,0){\line(1,6){.1667}}
  \put(0,0){\line(2,1){1}}
  \put(0,0){\line(2,3){.6667}}
  \put(0,0){\line(2,5){.4}}
  \put(0,0){\line(3,1){1}}
  \put(0,0){\line(3,2){1}}
  \put(0,0){\line(3,4){.75}}
  \put(0,0){\line(3,5){.6}}
  \put(0,0){\line(4,1){1}}
  \put(0,0){\line(4,3){1}}
  \put(0,0){\line(4,5){.8}}
  \put(0,0){\line(5,1){1}}
  \put(0,0){\line(5,2){1}}
  \put(0,0){\line(5,3){1}}
  \put(0,0){\line(5,4){1}}
  \put(0,0){\line(5,6){.8333}}
  \put(0,0){\line(6,1){1}}
  \put(0,0){\line(6,5){1}}
\end{picture}
```



Відрізки малюють командою

$\boxed{\texttt{\textbackslash put}(x,y)\texttt{\{\textbackslash line}(x_1,y_1)\texttt{\{}length\texttt{\}\}}}$

Команда `\line` має два аргументи:

1. вектор напряму,
2. довжина.

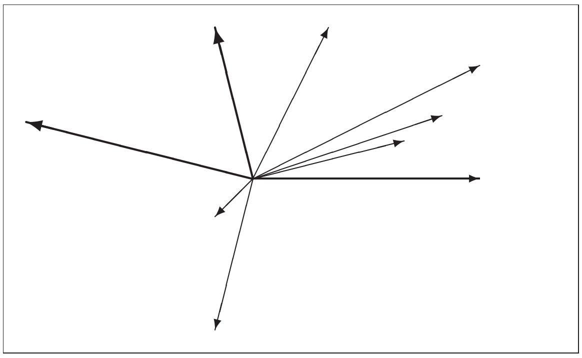
Значення компонент вектору напряму обмежені цілими числами

$$-6,\ -5,\ \ldots,\ 5,\ 6,$$

і вони не повинні мати спільних дільників (крім одиниці) Рисунок ілюструє всі двадцять п'ять можливих значень кутів нахилу у першому квадранті. Довжина задана в одиницях `\unitlength`. Аргумент «довжина» задає вертикальну координату у випадку вертикального відрізка і горизонтальну координату у всіх інших випадках.

5.2.3 Стрілки

```
\setlength{\unitlength}{1mm}
\begin{picture}(60,40)
  \put(30,20){\vector(1,0){30}}
  \put(30,20){\vector(4,1){20}}
  \put(30,20){\vector(3,1){25}}
  \put(30,20){\vector(2,1){30}}
  \put(30,20){\vector(1,2){10}}
  \thicklines
  \put(30,20){\vector(-4,1){30}}
  \put(30,20){\vector(-1,4){5}}
  \thinlines
  \put(30,20){\vector(-1,-1){5}}
  \put(30,20){\vector(-1,-4){5}}
\end{picture}
```



Стрілки малюють командою

$\boxed{\texttt{\textbackslash put}(x,y)\texttt{\{\textbackslash vector}(x_1,y_1)\texttt{\{}length\texttt{\}\}}}$

Для стрілок значення компонентів вектора напряму обмежені ще вужче ніж для відрізків, а саме, цілими числами

$$-4,\ -3,\ \ldots,\ 3,\ 4.$$

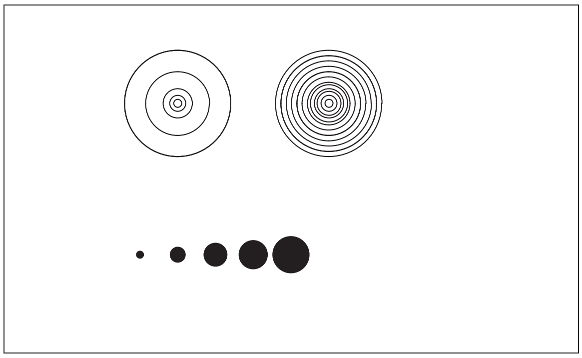
Компоненти також не можуть мати спільних дільників (крім одиниці). Зверніть увагу на вплив команди `\thicklines` (товщина) на дві стрілки, що вказують вгору ліворуч.

5.2.4 Кола

```
\setlength{\unitlength}{1mm}
\begin{picture}(60, 40)
  \put(20,30){\circle{1}}
  \put(20,30){\circle{2}}
  \put(20,30){\circle{4}}
  \put(20,30){\circle{8}}
  \put(20,30){\circle{16}}
  \put(20,30){\circle{32}}

  \put(40,30){\circle{1}}
  \put(40,30){\circle{2}}
  \put(40,30){\circle{3}}
  \put(40,30){\circle{4}}
  \put(40,30){\circle{5}}
  \put(40,30){\circle{6}}
  \put(40,30){\circle{7}}
  \put(40,30){\circle{8}}
  \put(40,30){\circle{9}}
  \put(40,30){\circle{10}}
  \put(40,30){\circle{11}}
  \put(40,30){\circle{12}}
  \put(40,30){\circle{13}}
  \put(40,30){\circle{14}}

  \put(15,10){\circle*{1}}
  \put(20,10){\circle*{2}}
  \put(25,10){\circle*{3}}
  \put(30,10){\circle*{4}}
  \put(35,10){\circle*{5}}
\end{picture}
```



Команда

`\put(`x, y`){\circle{`*diameter*`}}`

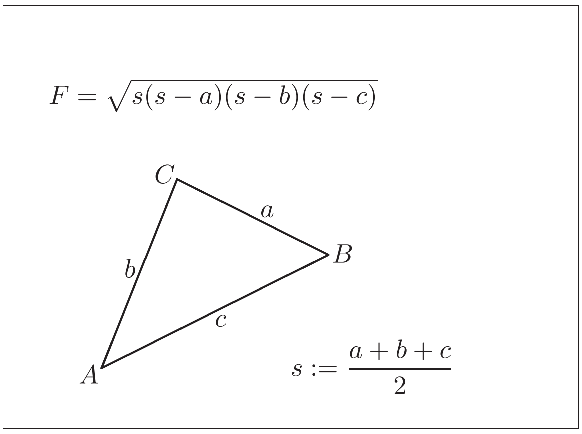
малює кола з центром в (x, y) і діаметром (не радіусом) *diameter*. Оточення `picture` дозволяє діаметри тільки приблизно до 14 мм, але навіть до цієї межі не всі діаметри можливі. Команда `\circle*` зображує диски (суцільні кола).

Як і у випадку з відрізками, для подолання цих обмежень можна звернутися до додаткових пакетів, наприклад, eepic або pstricks. Детальний опис цих пакетів наведено в *The LaTeX Graphics Companion* [1].

В середовищі `picture` є ще інші можливості. Якщо ви не боїтеся додаткових розрахунків (які можна зробити за допомогою програми), довільні кола та еліпси можуть бути зліплені з кривих Безьє. Дивіться приклади і джерельні коди програм на Java в *Graphics in LaTeX 2ε* [17].

5.2.5 Текст і формули

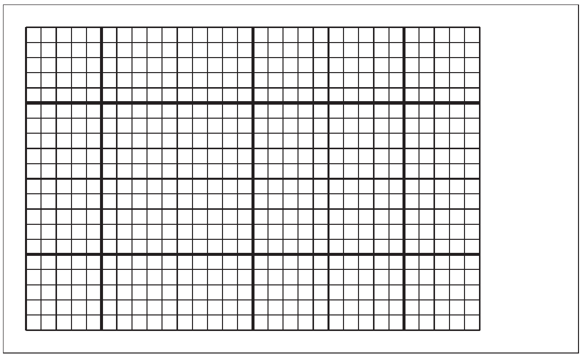
```
\setlength{\unitlength}{1cm}
\begin{picture}(6,5)
  \thicklines
  \put(1,0.5){\line(2,1){3}}
  \put(4,2){\line(-2,1){2}}
  \put(2,3){\line(-2,-5){1}}
  \put(0.7,0.3){$A$}
  \put(4.05,1.9){$B$}
  \put(1.7,2.95){$C$}
  \put(3.1,2.5){$a$}
  \put(1.3,1.7){$b$}
  \put(2.5,1.05){$c$}
  \put(0.3,4){$F=
    \sqrt{s(s-a)(s-b)(s-c)}$}
  \put(3.5,0.4){$\displaystyle
    s:=\frac{a+b+c}{2}$}
\end{picture}
```



Як показано в цьому прикладі, текст і формули можна вписувати в оточення `picture` за допомогою команди `\put` як звичайно.

5.2.6 Команди `\multiput` та `\linethickness`

```
\setlength{\unitlength}{2mm}
\begin{picture}(30,20)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){31}%
    {\line(0,1){20}}
  \multiput(0,0)(0,1){21}%
    {\line(1,0){30}}
  \linethickness{0.15mm}
  \multiput(0,0)(5,0){7}%
    {\line(0,1){20}}
  \multiput(0,0)(0,5){5}%
    {\line(1,0){30}}
  \linethickness{0.3mm}
  \multiput(5,0)(10,0){3}%
    {\line(0,1){20}}
  \multiput(0,5)(0,10){2}%
    {\line(1,0){30}}
\end{picture}
```



Команда

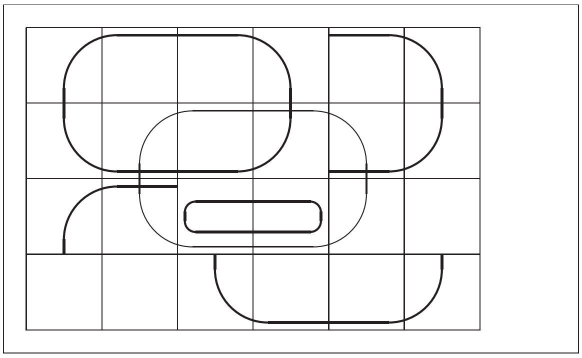
$\boxed{\texttt{\textbackslash multiput}(x,y)(\Delta x,\Delta y)\{n\}\{object\}}$

має чотири аргументи: початкову точку, вектор зміщення від одного об’-

єкта до наступного, кількість об'єктів, і об'єкт, що потрібно вивести. Команда `\linethickness` стосується горизонтальних і вертикальних відрізків і квадратичних кривих Безьє, але не стосується ні похилих відрізків ні елементів кіл.

5.2.7 Овали. Команди \thinlines та \thicklines

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}%
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}%
    {\line(1,0){6}}
  \thicklines
  \put(2,3){\oval(3,1.8)}
  \thinlines
  \put(3,2){\oval(3,1.8)}
  \thicklines
  \put(2,1){\oval(3,1.8)[tl]}
  \put(4,1){\oval(3,1.8)[b]}
  \put(4,3){\oval(3,1.8)[r]}
  \put(3,1.5){\oval(1.8,0.4)}
\end{picture}
```



Команда

`\put(`x, y`){\oval(`w, h`)}`

або

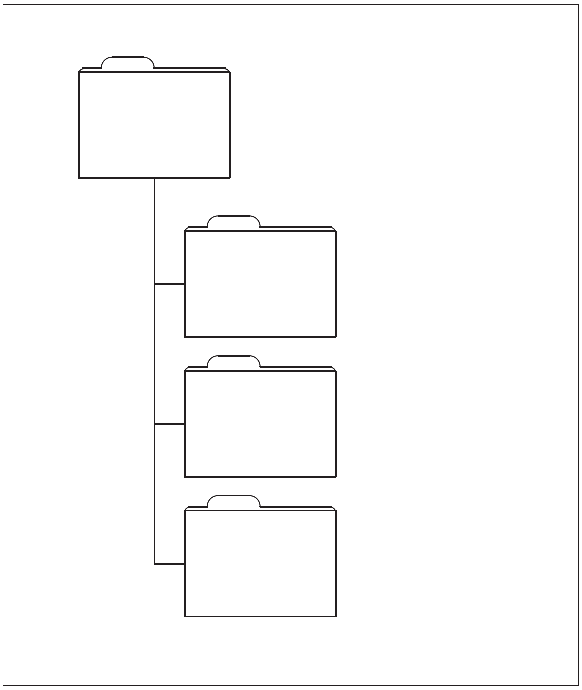
`\put(`x, y`){\oval(`w, h`)[`*position*`]}`

виводить овал з центром в (x, y) шириною w і висотою h. Необов'язкові аргументи *позиції* `b`, `t`, `l`, `r` означають, який фрагмент овала («верхній», «нижній», «лівий», чи «правий») слід намалювати, їх можна використовувати разом, як показано на ілюстрації.

Товщину ліній похилих відрізків та кіл і овалів можна задавати командами `\thinlines` та `\thicklines` (тонкі та грубі лінії, відповідно).

5.2.8 Багаторазове використання заздалегідь визначених елементів зображення

```
\setlength{\unitlength}{0.5mm}
\begin{picture}(120,168)
\newsavebox{\foldera}% декларація
\savebox{\foldera}
  (40,32)[bl]{%        визначення
  \multiput(0,0)(0,28){2}
    {\line(1,0){40}}
  \multiput(0,0)(40,0){2}
    {\line(0,1){28}}
  \put(1,28){\oval(2,2)[tl]}
  \put(1,29){\line(1,0){5}}
  \put(9,29){\oval(6,6)[tl]}
  \put(9,32){\line(1,0){8}}
  \put(17,29){\oval(6,6)[tr]}
  \put(20,29){\line(1,0){19}}
  \put(39,28){\oval(2,2)[tr]}
}
\newsavebox{\folderb}% декларація
\savebox{\folderb}
  (40,32)[l]{%        визначення
  \put(0,14){\line(1,0){8}}
  \put(8,0){\usebox{\foldera}}
}
\put(34,26){\line(0,1){102}}
\put(14,128){\usebox{\foldera}}
\multiput(34,86)(0,-37){3}
  {\usebox{\folderb}}
\end{picture}
```



Малюнок можна *задекларувати* командою

\newsavebox{*назва*}

а потім *визначити* командою

\savebox{*назва*}(*ширина,висота*)[*позиція*]{*зміст*}

і, накінець, (багаторазово) *вивести* командою

\put(x, y)\usebox{*назва*}

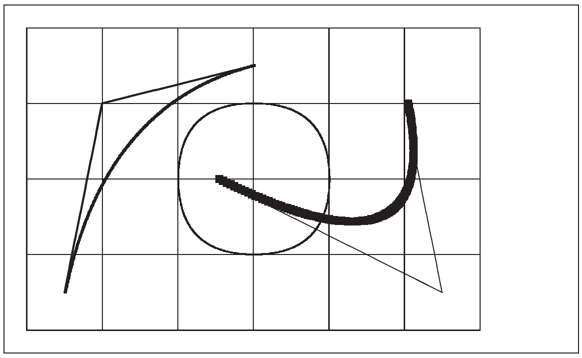
Необов'язковий параметр *позиція* вказує, де розмістити «точку відліку» зображення всередині прямокутника \savebox. У даному випадку йому задано значення `bl`, таким чином розміщує «точку відліку» зображення в нижньому лівому куті. Іншими можливими значеннями специфікації позиції є `t` (верх) and `r` (праворуч).

Аргумент *назва* вказує на буфер в пам'яті, де LaTeX зберігає зображення, і має властивості команди (повиннен починатися із зворотної похилої). Такі зображення можуть бути розміщені одне в одному: у попередньому прикладі `\foldera` використаний всередині визначення `\folderb`.

В даному прикладі використовується команда `\oval`, оскільки `\line` не працює, якщо довжина відрізка менша, ніж 3 мм.

5.2.9 Квадратичні криві Безьє

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)
  \linethickness{0.075mm}
  \multiput(0,0)(1,0){7}
    {\line(0,1){4}}
  \multiput(0,0)(0,1){5}
    {\line(1,0){6}}
  \thicklines
  \put(0.5,0.5){\line(1,5){0.5}}
  \put(1,3){\line(4,1){2}}
  \qbezier(0.5,0.5)(1,3)(3,3.5)
  \thinlines
  \put(2.5,2){\line(2,-1){3}}
  \put(5.5,0.5){\line(-1,5){0.5}}
  \linethickness{1mm}
  \qbezier(2.5,2)(5.5,0.5)(5,3)
  \thinlines
  \qbezier(4,2)(4,3)(3,3)
  \qbezier(3,3)(2,3)(2,2)
  \qbezier(2,2)(2,1)(3,1)
  \qbezier(3,1)(4,1)(4,2)
\end{picture}
```



Як показано в цьому прикладі, поділ кола на чотири квадратичні криві Безьє не дає задовільного результату. Потрібно не менш ніж вісім сегментів. Приклад показує також ефект команди `\linethickness` на горизонтальні та вертикальні відрізки, а також команд `\thinlines` та `\thicklines` на похилі відрізки. Він також показує, що обидві групи команд впливають на товщину квадратичних кривих Безьє, кожна анулюючи дію попередніх.

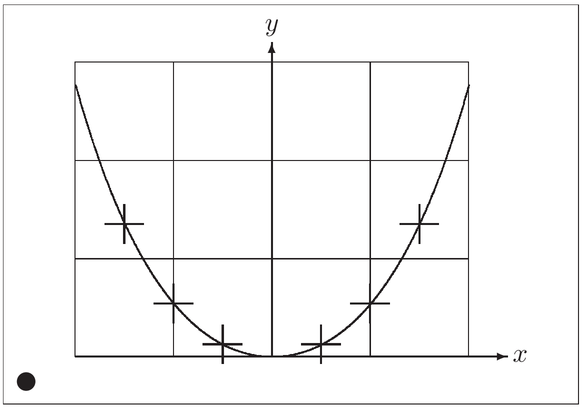
Нехай $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$ позначає кінцеві точки, а m_1, m_2 — відповідні кути нахилу квадратичної кривої Безьє. Проміжна контроль-на точки $S = (x, y)$ визначається за допомогою рівняння

$$\begin{cases} x = \dfrac{m_2 x_2 - m_1 x_1 - (y_2 - y_1)}{m_2 - m_1}, \\ y = y_i + m_i(x - x_i) \qquad (i = 1,\, 2). \end{cases} \tag{5.1}$$

В *Graphics in LaTeX 2ε* [17] описана програма на Java, яка генерує необхідні команди `\qbezier`.

5.2.10 Ланцюгова крива

```
\setlength{\unitlength}{1.3cm}
\begin{picture}(4.3,3.6)(-2.5,-0.25)
  \put(-2,0){\vector(1,0){4.4}}
  \put(2.45,-.05){$x$}
  \put(0,0){\vector(0,1){3.2}}
  \put(0,3.35){\makebox(0,0){$y$}}
  \qbezier(0.0,0.0)(1.2384,0.0)
    (2.0,2.7622)
  \qbezier(0.0,0.0)(-1.2384,0.0)
    (-2.0,2.7622)
  \linethickness{.075mm}
  \multiput(-2,0)(1,0){5}
    {\line(0,1){3}}
  \multiput(-2,0)(0,1){4}
    {\line(1,0){4}}
  \linethickness{.2mm}
  \put( .3,.12763){\line(1,0){.4}}
  \put(.5,-.07237){\line(0,1){.4}}
  \put(-.7,.12763){\line(1,0){.4}}
  \put(-.5,-.07237){\line(0,1){.4}}
  \put(.8,.54308){\line(1,0){.4}}
  \put(1,.34308){\line(0,1){.4}}
  \put(-1.2,.54308){\line(1,0){.4}}
  \put(-1,.34308){\line(0,1){.4}}
  \put(1.3,1.35241){\line(1,0){.4}}
  \put(1.5,1.15241){\line(0,1){.4}}
  \put(-1.7,1.35241){\line(1,0){.4}}
  \put(-1.5,1.15241){\line(0,1){.4}}
  \put(-2.5,-0.25){\circle*{0.2}}
\end{picture}
```



На цьому малюнку симетричні половини кривої $y = \cosh x - 1$ апроксимовані квадратичними кривими Безьє. Права половина кривої починається в точці (2, 2.7622), кут нахилу має значення $m = 3.6269$. Користуючись рівнянням (5.1) можна обчислити проміжні контрольні точки. В даному прикладі це (1.2384, 0) і (−1.2384, 0). Хрестики вказують на точки *справжньої* кривої. Помилка (менше одного відсотка) майже не помітна.

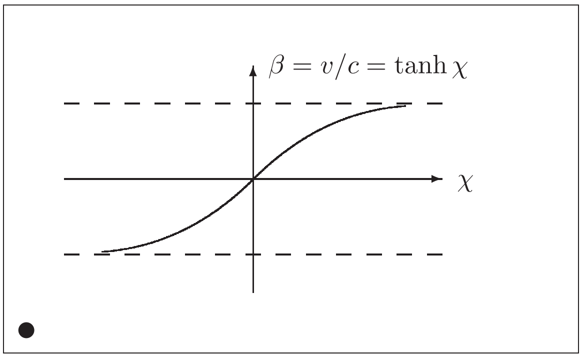
Цей приклад також показує використання необов'язкового аргументу команди `\begin{picture}`. Зображення задане в зручних «математичних» координатах, а за допомогою команди

```
\begin{picture}(4.3,3.6)(-2.5,-0.25)
```

його нижньому лівому куту (поміченому жирною крапкою) присвоєні координати $(-2.5, -0.25)$.

5.2.11 Швидкість в спеціальній теорії відносності

```
\setlength{\unitlength}{1cm}
\begin{picture}(6,4)(-3,-2)
  \put(-2.5,0){\vector(1,0){5}}
  \put(2.7,-0.1){$\chi$}
  \put(0,-1.5){\vector(0,1){3}}
  \multiput(-2.5,1)(0.4,0){13}
    {\line(1,0){0.2}}
  \multiput(-2.5,-1)(0.4,0){13}
    {\line(1,0){0.2}}
  \put(0.2,1.4)
    {$\beta=v/c=\tanh\chi$}
  \qbezier(0,0)(0.8853,0.8853)
    (2,0.9640)
  \qbezier(0,0)(-0.8853,-0.8853)
    (-2,-0.9640)
  \put(-3,-2){\circle*{0.2}}
\end{picture}
```



Контрольні точки двох кривих Безьє розраховані за допомогою формули (5.1). Позитивна гілка визначена рівняннями $P_1 = (0, 0)$, $m_1 = 1$ та $P_2 = (2, \tanh 2)$, $m_2 = 1/\cosh^2 2$. Знову, зображення задане в математично зручних координатах, і лівий нижній кут має координати $(-3, -2)$ (жирна крапка).

5.3 XY-pic

By Alberto Manuel Brandão Simões <albie@alfarrabio.di.uminho.pt>

xy — спеціальний пакет для малювання діаграм. Для того, щоб скористуватися ним, додайте наступні рядки у преамбулу документа:

```
\usepackage[опції]{xy}
```

де *опції* — це список функцій XY-pic, які потрібно завантажити. Ці опції корисні в першу чергу для відладки пакета. Я рекомендую скористатися опцією `all`, примусивши LaTeX завантажити усі команди XY.

Діаграми XY-pic малюються на матрице-подібному «полотні», де кожен з елементів розміщений в окремій клітині:

```
\begin{displaymath}
\xymatrix{A & B \\
          C & D }
\end{displaymath}
```

$$\begin{matrix} A & B \\ C & D \end{matrix}$$

Команду `\xymatrix` слід використовувати в математичному режимі. У даному прикладі ми визначили два стовпчики і два рядки. Щоб зробити цю матрицю діаграмою, слід додати стрілки за допомогою команди `\ar`:

```
\begin{displaymath}
\xymatrix{ A \ar[r] & B \ar[d] \\
           D \ar[u] & C \ar[l] }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[r] & B \arrow[d] \\ D \arrow[u] & C \arrow[l] \end{tikzcd}$$

Команди стрілок розміщені в клітинках, з яких вони виходять. Аргументами цих команд є напрям, куди стрілки мають вказувати (`u` — догори, `d` — донизу, `r` — праворуч, та `l` — ліворуч).

```
\begin{displaymath}
\xymatrix{
  A \ar[d] \ar[dr] \ar[r] & B \\
  D                       & C }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[d] \arrow[dr] \arrow[r] & B \\ D & C \end{tikzcd}$$

Щоб зробити діагоналі, слід вказати більш ніж один напрямок. Також, можна вказати той самий напрямок кілька разів, щоб зробити довшу стрілку.

```
\begin{displaymath}
\xymatrix{
  A \ar[d] \ar[dr] \ar[drr] & & \\
  B                       & C & D }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[d] \arrow[dr] \arrow[drr] & & \\ B & C & D \end{tikzcd}$$

Можна зробити діаграми цікавішими, додавши до стрілок мітки. Для цього скористайтеся звичайними операторами верхнього та нижнього індексів.

```
\begin{displaymath}
\xymatrix{
  A \ar[r]^f \ar[d]_g &
            B \ar[d]^{g’} \\
  D \ar[r]_{f’}       & C }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ D \arrow[r, "f'"'] & C \end{tikzcd}$$

Як показано, ці оператори використовуються так само, як і в математичному режимі. Єдиною різницею є те, що верхній індекс значить «над стрілкою», а нижній індекс — «під стрілкою». Є ще третій оператор, вертикальна паличка: `|`. Він розміщує мітку посередині стрілки.

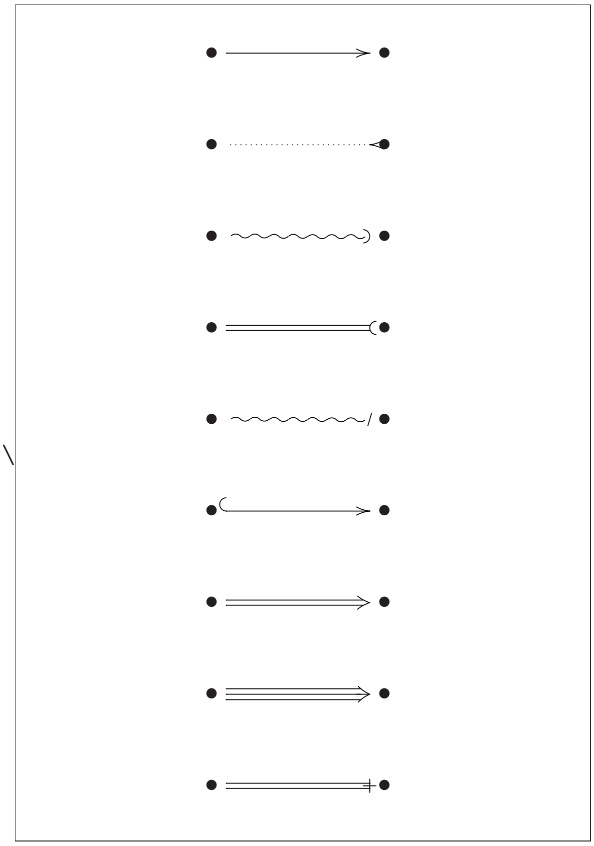
```
\begin{displaymath}
\xymatrix{
  A \ar[r]|f \ar[d]|g &
            B \ar[d]|{g’} \\
  D \ar[r]|{f’}       & C }
\end{displaymath}
```

$$\begin{tikzcd} A \arrow[r, "f" description] \arrow[d, "g" description] & B \arrow[d, "g'" description] \\ D \arrow[r, "f'" description] & C \end{tikzcd}$$

Щоб намалювати стрілку з отвором, скористайтеся `\ar[...]|\hole`.

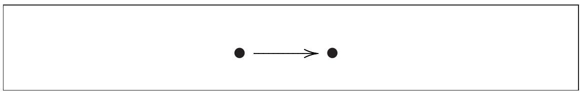
У деяких випадках важливо розрізняти стрілки різних типів. Це можна зробити за допомогою міток, або змінивши зовнішній вигляд стрілки:

```
\shorthandoff{"}
\begin{displaymath}
\xymatrix{
 \bullet\ar@{->}[rr] && \bullet\\
 \bullet\ar@{.<}[rr] && \bullet\\
 \bullet\ar@{~)}[rr] && \bullet\\
 \bullet\ar@{=(}[rr] && \bullet\\
 \bullet\ar@{~/}[rr]  && \bullet\\
 \bullet\ar@{^{(}->}[rr]  && \bullet\\
 \bullet\ar@2{->}[rr]  && \bullet\\
 \bullet\ar@3{->}[rr]  && \bullet\\
 \bullet\ar@{=+}[rr]   && \bullet
}
\end{displaymath}
\shorthandon{"}
```

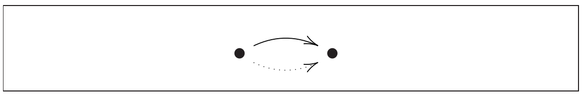


Зверніть увагу на різницю між двома наступними діаграмами:

```
\begin{displaymath}
\xymatrix{
 \bullet \ar[r]
        \ar@{.>}[r] &
 \bullet
}
\end{displaymath}
```



```
\begin{displaymath}
\xymatrix{
 \bullet \ar@/^/[r]
        \ar@/_/@{.>}[r] &
 \bullet
}
\end{displaymath}
```



Модифікатори між похилими рисками визначають, як малювати криві. XY-pic пропонує багато способів зображення кривих, щоб отримати більше інформації, зверніться до документації XY-pic.

Розділ 6

Налаштування LaTeX

Документи, створені з використанням вивчених дотепер команд, будуть виглядати прийнятно для широкої аудиторії. Не виглядаючи занадто вибагливо, вони відповідають усім правилам гарного тону в наборі, тому їх легко читати, і на них приємно глянути.

Проте, бувають ситуації, у яких LaTeX не надає команду чи оточення, які б задовольняли вашим потребам, або вивід, вироблений певною існуючою командою, не відповідає вашим вимогам.

У цьому розділі я спробую дати деякі поради з приводу того, як навчити LaTeX новим речам, і як примусити його давати вивід, що відрізняється від стандартного.

6.1 Нові команди, оточення і пакети

Ви певне помітили, що усі команди, представлені в цій книзі, оточені рамкою і включені в покажчик наприкінці книги. Замість того, щоб прямо використовувати необхідні для цього команди LaTeX, я створив пакет, у якому визначив для цього нові команди й оточення. Тепер можна просто писати:

```
\begin{lscommand}
\ci{dum}
\end{lscommand}
```

`\dum`

У цьому прикладі я використовую нове оточення `lscommand`, що малює рамку навколо команди, одночасно з новою командою `\ci`, що виводить назву команди і заносить відповідний елемент у покажчик. Ви можете у цьому переконатися, відшукавши команду `\dum` у покажчику наприкінці книги, де ви знайдете елемент для `\dum`, що вказує на кожну сторінку, де я згадую команду `\dum`.

Якщо я колись вирішу, що мені більше не подобаються команди, набрані в рамці, я просто зміню визначення оточення `lscommand`. Це набагато простіше, ніж проглядати весь документ, вишукуючи усі місця, де я використав загальні команди LaTeX для зображення рамки навколо слів.

6.1.1 Нові команди

Щоб створити ваші власні команди, скористайтеся командою

`\newcommand{`*назва*`}[`*число*`]{`*визначення*`}`

Звичайно ця команда приймає два аргументи: *назву* команди, яку ви створюєте, і *визначення* команди. Необов'язковий аргумент *число* у квадратних дужках визначає кількість аргументів нової команди (може бути не більше 9). Якщо цей аргумент відсутній, вважається 0, тобто нова команда не має аргументів.

Наступні два приклади повинні допомогти вам отримати уявлення про те, як працює `\newcommand`. Перший приклад визначає нову команду, що називається `\tnss`, що є скороченням від «The Not So Short Introduction to LaTeX 2_ε». Така команда знадобиться, якщо вам доведеться багато разів писати назву цієї книги.

```
\newcommand{\tnss}{The not
    so Short Introduction to
    \LaTeXe}
This is ‘‘\tnss’’ \ldots{}
‘‘\tnss’’
```

This is “The not so Short Introduction to LaTeX 2_ε” … “The not so Short Introduction to LaTeX 2_ε”

Наступний приклад ілюструє, як визначити команду з одним аргументом. Мітка `#1` замінюється на заданий аргумент. Якщо ви хочете використовувати більш одного аргумента, використовуйте `#2`, і так далі.

```
\newcommand{\txsit}[1]
 {This is the \emph{#1} Short
      Introduction to \LaTeXe}
% в тілі документа:
\begin{itemize}
\item \txsit{not so}
\item \txsit{very}
\end{itemize}
```

- This is the *not so* Short Introduction to LaTeX 2_ε
- This is the *very* Short Introduction to LaTeX 2_ε

LaTeX не дозволить вам створити нову команду, яка б замінила вже існуючу. Але є спеціальна команда для випадку, коли ви явно хочете це зробити: `\renewcommand`. Вона має той же синтаксис, що і команда `\newcommand`.

В певних випадках може знадобитися команда `\providecommand`. Вона працює так само, як `\newcommand`, але, якщо команда уже визначена, LaTeX 2_ε її просто проігнорує.

Потрібно звернути увагу на кілька моментів стосовно пробілів після команд. Деталі дивіться на сторінці 5.

6.1.2 Нові оточення

Аналогічно до команди `\newcommand`, існує команда для створення ваших власних оточень. Команда `\newenvironment` має наступний синтаксис:

`\newenvironment{`*назва*`}[`*номер*`]{`*перед*`}{`*після*`}`

Як і команду `\newcommand`, `\newenvironment` можна використовувати з необов'язковим аргументом або без нього. Матеріал, вказаний в аргументі *перед*, обробляється до обробки тексту усередині оточення. Матеріал, вказаний в аргументі *після*, обробляється, коли зустрічається команда `\end{`*назва*`}`.

Використання команди `\newenvironment` проілюстровано наступним прикладом.

```
\newenvironment{king}
 {\rule{1ex}{1ex}%
      \hspace{\stretch{1}}}
 {\hspace{\stretch{1}}%
      \rule{1ex}{1ex}}

\begin{king}
Мої покірні піддані\ldots
\end{king}
```

■ Мої покірні піддані… ■

Аргумент *номер* використовують так же, як і в команді `\newcommand`. LaTeX контролює, щоб ви не визначили вже існуюче оточення. Якщо потрібно явно перевизначити існуюче оточення, скористайтеся командою `\renewenvironment`, що має такий же синтаксис, як і `\newenvironment`.

Команди, використані у цьому прикладі, будуть пояснені пізніше. Опис команди `\rule` дивіться на сторінці 112, команда `\stretch` описана на сторінці 106, інформацію про команду `\hspace` можна знайти на сторінці 105.

6.1.3 Ваш власний пакет

Коли ви визначаєте безліч нових оточень і команд, преамбули ваших документів стають дуже довгими. У цій ситуації хорошою ідеєю може бути створення пакету LaTeX, що містить визначення усіх ваших команд і оточень. Потім можна командою `\usepackage` використовувати пакет у ваших документах.

```
% Demo Package by Tobias Oetiker
\ProvidesPackage{demopack}
\newcommand{\tnss}{Не надто короткий вступ до  \LaTeXe}
\newcommand{\txsit}[1]{\emph{#1}
                              короткий вступ до \LaTeXe}
\newenvironment{king}{\begin{quote}}{\end{quote}}
```

Рис. 6.1: Приклад пакета

Створення пакета в основному складається з переносу вмісту вашої преамбули в окремий файл з назвою, що закінчується на `.sty`. Є тільки одна спеціальна команда, яку ви повинні використовувати

`\ProvidesPackage{`*назва пакета*`}`

на самому початку файлу з вашим пакетом. `\ProvidesPackage` указує LaTeX назву пакета, що дозволяє йому видавати зрозуміле повідомлення про помилку, коли ви намагаєтеся підключити пакет двічі. Рис. 6.1 показує маленький приклад пакета, що містить визначені в наведених вище прикладах команди.

6.2 Шрифти та розміри

6.2.1 Команди зміни шрифту

LaTeX вибирає відповідний шрифт і розмір шрифту, ґрунтуючись на логічній структурі документа (розділи, виноски, ...). Іноді буває потрібно змінити шрифт вручну. Для цього ви можете скористатися командами, переліченими в таблицях 6.1 та 6.2. Справжній розмір кожного шрифту — це питання дизайну, і залежить від класу і опцій документа. Таблиця 6.3 показує абсолютні розміри, що відповідають цим командам у стандартних класах документів.

```
{\small The small and
\textbf{bold} Romans ruled}
{\Large all of great big
\textit{Italy}.}
```

The small and **bold** Romans ruled all of great big *Italy*.

Важливою особливістю LaTeX 2_ε є те, що атрибути шрифту незалежні. Це значить, що ви можете давати команди зміни розміру чи навіть шрифту, зберігаючи при цьому атрибути насиченості чи нахилу, встановлені раніше.

У *математичному режимі* можна використовувати *команди* зміни шрифту, щоб тимчасово вийти з *математичного режиму* і увести нормальний текст. Якщо потрібно переключитися на інший шрифт для набору математики, для цього існує спеціальний набір команд. Дивіться таблицю 6.4.

У зв'язку з командами зміни розміру шрифту помітну роль грають фігурні дужки. Вони використовуються для побудови *груп*. Групи обмежують область дії більшості команд LaTeX.

```
Йому подобаються
{\LARGE великі і
{\small маленькі} літери}.
```

Йому подобаються великі і маленькі літери.

Команди, що впливають на розмір шрифту, впливають також на відстань між рядками, але тільки якщо відповідний абзац закінчується усередині області дії команди. Тому закриваюча фігурна дужка `}` не повинна стояти занадто рано. Зверніть увагу на положення команди `\par` у

Табл. 6.1: Шрифти

`\textrm{...}`	з зарубками	`\textsf{...}`	без зарубок
`\texttt{...}`	`машинопис`		
`\textmd{...}`	нормальний	`\textbf{...}`	**жирний**
`\textup{...}`	прямий	`\textit{...}`	*курсив*
`\textsl{...}`	*похилий*	`\textsc{...}`	КАПІТЕЛЬ
`\emph{...}`	*виділений*	`\textnormal{...}`	звичайний

Табл. 6.2: Розміри шрифтів

`\tiny`	малесенький	`\Large`	більший
`\scriptsize`	дуже маленький	`\LARGE`	дуже великий
`\footnotesize`	досить маленький	`\huge`	величезний
`\small`	маленький	`\Huge`	гігантський
`\normalsize`	нормальний		
`\large`	великий		

Табл. 6.3: Абсолютні розміри шрифтів у стандартних класах

розмір	10pt (за замовч.)	опція 11pt	опція 12pt
`\tiny`	5pt	6pt	6pt
`\scriptsize`	7pt	8pt	8pt
`\footnotesize`	8pt	9pt	10pt
`\small`	9pt	10pt	11pt
`\normalsize`	10pt	11pt	12pt
`\large`	12pt	12pt	14pt
`\Large`	14pt	14pt	17pt
`\LARGE`	17pt	17pt	20pt
`\huge`	20pt	20pt	25pt
`\Huge`	25pt	25pt	25pt

Табл. 6.4: Математичні шрифти

Команда	*Приклад*	*Вивід*
`\mathcal{...}`	`$\mathcal{B}=c$`	$\mathcal{B}=c$
`\mathrm{...}`	`$\mathrm{K}_2$`	K_2
`\mathbf{...}`	`$\sum x=\mathbf{v}$`	$\sum x=\mathbf{v}$
`\mathsf{...}`	`$\mathsf{G\times R}$`	$\mathsf{G\times R}$
`\mathtt{...}`	`$\mathtt{L}(b,c)$`	$\mathtt{L}(b,c)$
`\mathnormal{...}`	`$\mathnormal{R_{19}}\neq R_{19}$`	$\mathnormal{R_{19}}\neq R_{19}$
`\mathit{...}`	`$\mathit{ffi}\neq ffi$`	$\mathit{ffi}\neq ffi$

наступних двох прикладах[1]:

```
{\Large Не читайте цього! Це
неправда. Вірте мені!\par}
```

Не читайте цього! Це неправда. Вірте мені!

```
{\Large Це також неправда.
Але пам’ятайте, я брехун.}\par
```

Це також неправда. Але пам’ятайте, я брехун.

Якщо потрібно застосувати команду зміни розміру до цілого абзацу тексту чи навіть більше, варто використовувати синтаксис оточення для команди зміни шрифту.

```
\begin{Large}
Це неправда. Але,
знову ж таки,
що в наші дні\ldots
\end{Large}
```

Це неправда. Але, знову ж таки, що в наші дні. . .

Це дозволить вам уникнути підрахунку безлічі фігурних дужок.

6.2.2 Увага, небезпека

Як відзначено на початку цього розділу, небезпечно загромаджувати ваші документи явними командами, на зразок тільки що описаних, тому що це працює всупереч основній ідеї LaTeX, а саме, відокремлення логічної і візуальної розмітки вашого документа. Це значить, що якщо ви користуєтеся тими самими командами зміни шрифту в різних місцях для набору певного виду інформації, слід використовувати `\newcommand` і визначити команду, що буде «логічною обгорткою» для команди зміни шрифту.

```
\newcommand{\oops}[1]{\textbf{#1}}
Не \oops{заходьте} до цієї кімнати,
в ній знаходиться \oops{машина}
невідомого походження
і призначення.
```

Не **заходьте** до цієї кімнати, в ній знаходиться **машина** невідомого походження і призначення.

Цей підхід має ту перевагу, що ви пізніше можете вирішити, що хочете використовувати інше візуальне представлення небезпеки, ніж `\textbf`, без необхідності пробиратися через весь документ, знаходити усі випадки `\textbf` і визначати у кожному з них, використано `\textbf`, щоб вказати на небезпеку, чи з якогось іншого приводу.

[1] Команда `\par` еквівалентна порожньому рядку

6.2.3 Порада

Для завершення нашої подорожі у світ шрифтів та їхніх розмірів, дозвольте дати вам одну пораду:

> **Пам'ятайте!** *Чим* БІЛЬ*ШЕ* шрифтів ВИ використовуєте **у** вашому документі, *тим* простіше ЙОГО ЧИТАТИ і тим ***гарнішим*** він буде.

6.3 Інтервали

6.3.1 Інтервали між рядками

Якщо вам потрібні більші інтервали між рядками, їхнє значення можна змінити, помістивши команду

`\linespread{коефіцієнт}`

у преамбулу документа. Для друку «у півтора інтервали» значення *коефіцієнта* повинно бути 1.3, а для друку «у два інтервали» — 1.6. За замовчуванням цей коефіцієнт дорівнює 1.

Майте на увазі, що команда `\linespread` діє грубо і не підходить для отримання документів видавничої якості. Якщо для зміни міжрядкового інтервалу є вагома причина, краще скористайтеся командою

`\setlength{\baselineskip}{1.5\baselineskip}`

```
{\setlength{\baselineskip}%
           {1.5\baselineskip}
Цей абзац набраний з міжрядковим
інтервалом в 1.5 рази більшим від
попереднього. Зверніть увагу
на команду par в кіці абзаца.\par}

Метою цього абзаца є показати, що
після закриття фігурної дужки все
повертається до нормального стану.
```

> Цей абзац набраний з міжрядковим інтервалом в 1.5 рази більшим від попереднього. Зверніть увагу на команду par в кіці абзаца.
>
> Метою цього абзаца є показати, що після закриття фігурної дужки все повертається до нормального стану.

6.3.2 Форматування абзаців

Два параметри LaTeX впливають на формат абзаців. Помістивши визначення типу

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

в преамбулу, ви зміните зовнішній вигляд абзаців. Ці два рядки збільшують відстань між абзацами і встановлюють абзацний відступ рівним нулю.

Елементи `plus` та `minus` повідомляють, що TEX може зменшити або збільшити відстань між абзацами на задану величину у випадку, коли не вдається вмістити абзаци на сторінці.

У континентальній Європі абзаци часто відокремлюють вертикальним відступом і не роблять абзацного відступу. Але майте на увазі, це також впливає на зміст. Відстані між його елементами також збільшуються. Щоб запобігти цьому, ці команди можна перенести з преамбули документа куди-небудь після `\tableofcontents`, або не використовувати їх зовсім, оскільки в професійно виготовлених книгах для відокремлення абзаців використовується абзацний відступ, а не вертикальний відступ.

Якщо потрібно зробити абзацний відступ у абзаці, який його не має, вставте команду

```
\indent
```

на початку абзацу[2]. Зрозуміло, що вона матиме ефект тільки якщо `\parindent` не встановлений рівним нулю.

Для створення абзацу без відступу, використайте

```
\noindent
```

на початку абзацу. Це може бути зручно, коли документ починається текстом, а не заголовком розділу.

6.3.3 Горизонтальні інтервали

LATEX автоматично визначає розміри пробілів між словами і реченнями. Щоб додати горизонтальний інтервал, користуйтеся

```
\hspace{довжина}
```

Якщо такий інтервал повинен бути витриманий, навіть якщо він потрапить на початок чи кінець рядка, використовуйте `\hspace*` замість `\hspace`. У найпростішому випадку *довжина* — це просто число та одиниця виміру. Найважливіші одиниці перелічені в таблиці 6.5.

[2] Для того, щоб перший абзац розділу був з відступом, користуйтеся пакетом indentfirst з комплекту «tools».

Табл. 6.5: Одиниці виміру TeX

`mm`	міліметр ≈ 1/25 дюйма
`cm`	сантиметр = 10 мм
`in`	дюйм = 25.4 мм
`pt`	пункт ≈ 1/72 дюйма ≈ $\frac{1}{3}$ мм
`em`	приблизно ширина літери «M» поточного шрифту
`ex`	приблизно висота літери «x» поточного шрифту

```
Цей\hspace{1.5cm}пробіл
1,5 см.
```

Цей пробіл 1,5 см.

Команда

\stretch{*n*}

де *n* — «фактор пружності», генерує спеціальний «гумовий» пробіл. Він розтягується, заповнюючи все місце, що залишилося в рядку. Якщо в одному рядку трапляться дві команди `\hspace{\stretch{`*n*`}}`, вони розтягуються пропорційно своїм факторам пружності.

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

x x x

Коли горизонтальний інтервал використовується разом з текстом, доцільно встановлювати його розмір відносно розміру поточного шрифту. Це можна зробити, використовуючи одиниці виміру, залежні від шрифту — `em` та `ex`:

```
{\Large{}ведикий\hspace{1em}y}\\
{\tiny{}крихітний\hspace{1em}y}
```

ведикий y
крихітний y

6.3.4 Вертикальні інтервали

Інтервали між абзацами, розділами, підрозділами, ... LaTeX визначає автоматично. При необхідності, додатковий пробіл *між двома абзацами* можна додати командою:

\vspace{*довжина*}

Звичайно ця команда вставляється між двома порожніми рядками.

Якщо цей простір повинен бути збережений навіть угорі чи внизу сторінки, використовуйте варіант команди з зірочкою: `\vspace*` замість `\vspace`.

Команду `\stretch` разом з `\pagebreak` можна застосовувати для набору тексту на останньому рядку сторінки чи для вертикального центрування тексту на сторінці.

```
Деякий текст\ldots

\vspace{\stretch{1}}
А іде на останньому рядку сторінки.\pagebreak
```

Додатковий пробіл між двома рядками *одного* абзацу чи всередині таблиці вказується командою

`\\[`*довжина*`]`

Командами `\bigskip` та `\smallskip` можна зробити вертикальні відступи заздалегідь визначених розмірів без потреби турбуватися про числа.

6.4 Макетування сторінки

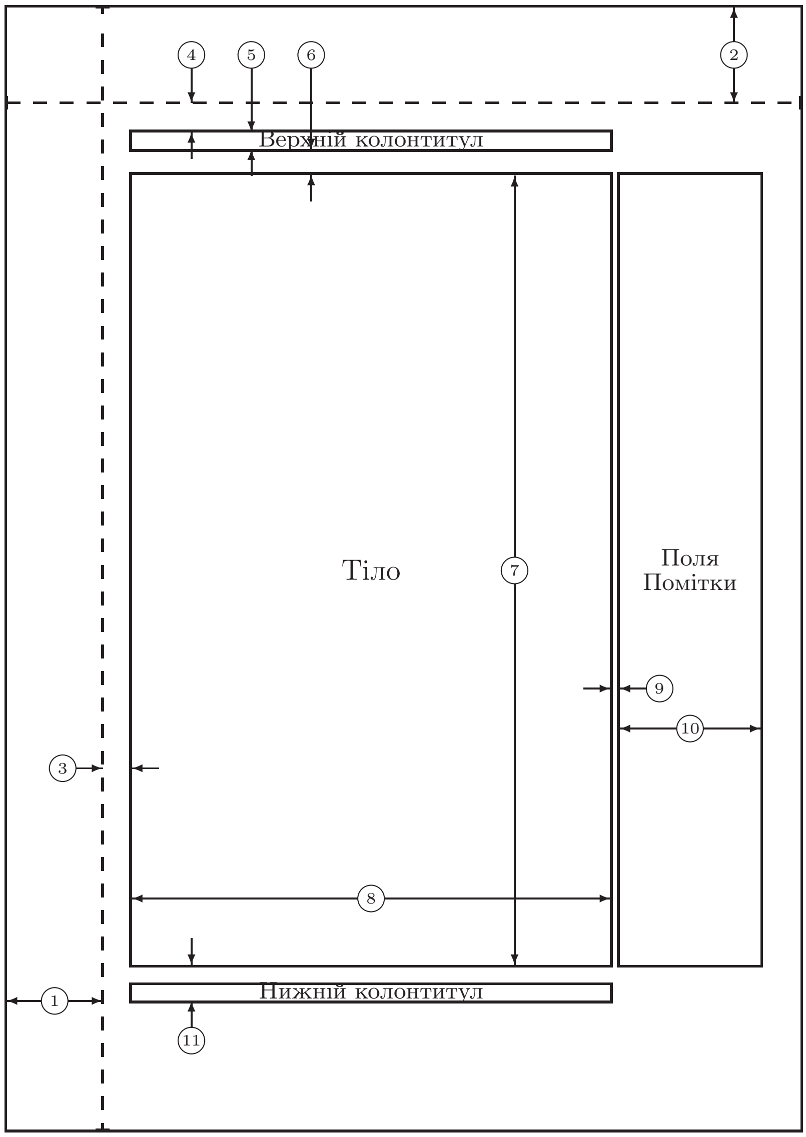
LaTeX 2_ε дозволяє вказати розмір аркуша у команді `\documentclass`. Потім він автоматично встановлює розміри полів. Але іноді автоматично встановлені значення можуть вас не влаштовувати. Безумовно, їх можна змінити. Рис. 6.2 показує всі параметри, які можна змінити. Ілюстрація була згенерована пакетом layout з набору «tools»[3].

ЗАЧЕКАЙТЕ! …перш, ніж ринутися «Давайте зробимо цю завузьку сторінку трошки ширшою», поміркуйте кілька секунд. Як і багато інших речей у LaTeX, існує вагома причина, чому макет сторінки такий, яким він є.

Безумовно, якщо порівняти із сторінкою документа, створеного свіжевстановленим MS Word, сторінки документів LaTeX виглядають жахливо вузькими. Однак, гляньте на вашу улюблену книгу[4] і порахуйте кількість літер на одному рядку. Ви побачите, що на кожному рядку не більше 66 символів. Тепер зробіть те ж саме зі сторінкою LaTeX. Ви побачите, що тут також близько 66 літер у рядку. Досвід показує, що читання ускладнюється при більшій кількості літер в одному рядку, тому, що очам стає важче переходити від кінця одного рядка до початку наступного. Саме тому газети часто набираються у декілька шпальт.

[3] `CTAN:/tex-archive/macros/latex/required/tools`

[4] Я маю на увазі справжню друковану книгу, видану шанованим видавництвом.



1	Один дюйм + \hoffset	2	Один дюйм + \voffset
3	\oddsidemargin = 22pt or \evensidemargin	4	\topmargin = 22pt
5	\headheight = 13pt	6	\headsep = 19pt
7	\textheight = 595pt	8	\textwidth = 360pt
9	\marginparsep = 7pt	10	\marginparwidth = 106pt
11	\footskip = 27pt		\marginparpush = 5pt (не показано)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 597pt		\paperheight = 845pt

Рис. 6.2: Параметри макета сторінки

Тому, збільшуючи ширину вашого тексту, майте на увазі, що ви ускладнюєте життя його читачам. Однак, досить попереджень, я обіцяв розповісти про те, як це зробити…

LaTeX надає дві команди для зміни цих параметрів. Їх звичайно використовують у преамбулі документа.

Перша команда призначає фіксоване значення будь-якому параметру

`\setlength{`*параметр*`}{`*довжина*`}`

Інша команда додає довжину до будь-якого параметра

`\addtolength{`*параметр*`}{`*довжина*`}`

Ця друга команда навіть більш корисна, ніж `\setlength`, тому що вона дозволяє робити настроювання відносно існуючих установок. Щоб додати сантиметр до загальної ширини тексту, у преамбулу потрібно помістити наступні команди:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

У цьому контексті вас може зацікавити пакет calc, що дозволяє використовувати арифметичні операції в аргументі `\setlength` і в інших місцях, де ви можете уводити чисельні значення як аргументи функцій.

6.5 Ще про довжини

Скрізь, де це можливо, я уникаю використання абсолютних величин в документах. Я радше спробую прив'язати розміри до ширини чи висоти інших елементів сторінки. Для ширини ілюстрації це може бути `\textwidth`, щоб вона заповнювала сторінку цілком.

Наступні три команди дозволяють визначити ширину, висоту і глибину текстового рядка.

`\settoheight{`*команда*`}{`*текст*`}`
`\settodepth{`*команда*`}{`*текст*`}`
`\settowidth{`*команда*`}{`*текст*`}`

Наступний приклад показує можливе застосування цих команд.

```
\flushleft
\newenvironment{vardesc}[1]{%
  \settowidth{\parindent}{#1:\ }
  \makebox[0pt][r]{#1:\ }}{}

\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}

\begin{vardesc}{Де}$a$,
$b$ "--- катети прямокутного
трикутника.

$c$ "--- самотня гіпотенуза
цього трикутника.

$d$ "--- взагалі тут не
з’являється. Чи це не дивно?
\end{vardesc}
```

$$a^2 + b^2 = c^2$$

Де: a, b — катети прямокутного трикутника.

c — самотня гіпотенуза цього трикутника.

d — взагалі тут не з’являється. Чи це не дивно?

6.6 Блоки

LaTeX складає сторінки з блоків. Спочатку кожна літера є маленьким блоком, що приклеюється до інших літер, формуючи слово. Слова склеюються з іншими словами спеціальним еластичним клеєм, який дозволяє серії слів розтягуватися чи стискатися, щоб точно заповнити рядок на сторінці.

Я визнаю, що це досить спрощена версія того, що відбувається насправді, але ідея в тім, що TeX завжди працює з блоками і клеєм. Не тільки літера може бути блоком. Ви можете помістити в блок практично все, що завгодно, включно з іншими блоками. Кожен блок потім обробляється LaTeX так, ніби це окрема літера.

В попередніх розділах ви уже зустрічали деякі блоки, хоча про це і не говорилося. Наприклад, і команда `\includegraphics`, і оточення `tabular`, обидва створюють блок. Це значить, що ви легко можете розмістити поруч дві таблиці чи ілюстрації. Тільки переконайтеся, що їхня загальна ширина не перевищує `\textwidth`.

Ви також можете упакувати будь-який абзац у блок командою

`\parbox[`*поз*`]{`*ширина*`}{`*текст*`}`

чи оточенням

`\begin{minipage}[`*поз*`]{`*ширина*`}` текст `\end{minipage}`

Параметр *поз* може приймати одну з літер `c`, `t` або `b`, контролюючи вертикальне вирівнювання блоку відносно базової лінії оточуючого тексту. *Ширина* приймає аргументом довжину, що визначає ширину блоку. Основна відмінність між `minipage` і `\parbox` у тім, що всередині `\parbox` можна використовувати не усі команди й оточення, в той час як практично все можливо всередині `minipage`.

У той час, як `\parbox` упаковує цілий абзац, розбиваючи рядки і таке інше, існує клас блокових команд, що працюють тільки з горизонтально розташованим матеріалом. Одну з них ми вже знаємо. Вона називається `\mbox` і просто упаковує послідовність блоків, що можна використовувати для запобігання розриву кількох слів. Оскільки можна розміщувати одні блоки всередині інших, це пакування горизонтальних блоків дає вам надзвичайну гнучкість. У команді

`\makebox[`*ширина*`][`*поз*`]{`*текст*`}`

ширина визначає ширину результуючого блока так, як її буде видно ззовні[5]. Крім виразів довжини, тут можна використовувати `\width`, `\height`, `\depth` і `\totalheight`. Вони встановлюються рівними значенням, отриманим виміром параметрів набраного *тексту*. Параметр *поз* приймає одне із значень: **c** — центрувати, **l** — вліво, **r** — вправо **s** — рівномірно заповнити блок текстом.

Команда `\framebox` працює точно так само, як `\makebox`, але креслить рамку навколо тексту.

Наступний приклад показує деякі можливості використання команд `\makebox` і `\framebox`.

```
\makebox[\textwidth]{%
    ц е н т р}\par
\makebox[\textwidth][s]{%
    р о з т я г н у т и й}\par
\framebox[1.1\width]{Гадаю,
    я тепер  у рамці!} \par
\framebox[0.8\width][r]{Ой,
    я занадто товстий} \par
\framebox[1cm][l]{нічого,
  я теж}
Можете це прочитати?
```

ц е н т р

р о з т я г н у т и й

Гадаю, я тепер у рамці!

Ой, я занадто товстий

нічого, я тежМожете це прочитати?

Тепер, коли ми керуємо горизонталлю, очевидний наступний крок —

[5]Це означає, що вона може бути меншою, ніж матеріал усередині блоку. Ви можете навіть установити її в 0pt, так що текст усередині блоку буде набиратися не впливаючи на навколишні блоки.

вертикаль[6]. Для LaTeX це не проблема. Команда

`\raisebox{`*підняття*`}[`*глибина*`][`*висота*`]{`*текст*`}`

дозволяє вам встановити вертикальні характеристики блоку. У перших трьох параметрах можна використовувати `\width`, `\height`, `\depth` і `\totalwidth`, щоб мати можливість посилатися на розміри блоку всередині аргументу *текст*.

```
\raisebox{0pt}[0pt][0pt]{\Large%
\textbf{Aaaa\raisebox{-0.3ex}{a}%
\raisebox{-0.7ex}{aa}%
\raisebox{-1.2ex}{a}%
\raisebox{-2.2ex}{a}%
\raisebox{-4.5ex}{a}}}
кричала вона, але навіть наступний
в черзі не помітив, що з нею
трапилося щось жахливе.
```

Aaaaaaaaa кричала вона, але навіть наступний в черзі не помітив, що з нею трапилося щось жахливе.

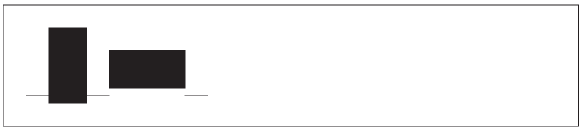
6.7 Лінійки і розпірки

Кілька сторінок тому ви могли помітити команду

`\rule[`*підняття*`]{`*ширина*`}{`*висота*`}`

При звичайному використанні вона генерує простий чорний прямокутник.

```
\rule{3mm}{.1pt}%
\rule[-1mm]{5mm}{1cm}%
\rule{3mm}{.1pt}%
\rule[1mm]{1cm}{5mm}%
\rule{3mm}{.1pt}
```



Це корисно для малювання вертикальних і горизонтальних ліній. Наприклад, лінія на титульному аркуші намальована командою `\rule`.

Спеціальним випадком є лінійка без ширини, але певної висоти. У професійній типографії її називають *розпіркою*. Її використовують, щоб забезпечити визначену мінімальну висоту елемента сторінки. Ви можете використовувати її в оточенні `tabular`, щоб гарантувати рядку визначену мінімальну висоту.

[6]Повний контроль можна отримати тільки одночасно контролюючи горизонталь і вертикаль . . .

```
\begin{tabular}{|c|}
\hline
\rule{1pt}{4ex}Підпірка \ldots\\
\hline
\rule{0pt}{4ex}Розпірка\\
\hline
\end{tabular}
```

Підпірка …
Розпірка

Кінець.

Бібліоґрафія

[1] Michel Goossens, Sebastian Rahtz and Frank Mittelbach. *The LaTeX Graphics Companion.* Addison-Wesley, Reading, Massachusetts, 1997, ISBN 0-201-85469-4. 84, 87

[2] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1. v, 1, 63, 83

[3] Donald E. Knuth. *The TeXbook,* Volume A of *Computers and Typesetting*, Addison-Wesley, Reading, Massachusetts, second edition, 1984, ISBN 0-201-13448-9. v, 1

[4] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8. v, 3, 10, 12, 13, 50, 63, 66, 83, 84

[5] Кожна установка LaTeX повинна містити так званий *LaTeX Local Guide*, в якому описані речі, особливі для даної системи. Цей документ повинен міститися у файлі з назвою `local.tex`. Нажаль, деякі ліниві системні адміністратори не надають такого документа. У цьому випадку зверніться за порадою до місцевого LaTeX 2_ε гуру. vi, 10, 25

[6] LaTeX3 Project Team. *LaTeX 2_ε for authors.* Входить в дистрибутив LaTeX 2_ε як файл `usrguide.tex`.

[7] LaTeX3 Project Team. *LaTeX 2_ε for Class and Package writers.* Входить в дистрибутив LaTeX 2_ε як `clsguide.tex`.

[8] LaTeX3 Project Team. *LaTeX 2_ε Font selection.* Входить в дистрибутив LaTeX 2_ε як `fntguide.tex`.

[9] D. P. Carlisle. *Packages in the 'graphics' bundle.* Входить в набір пакетів «graphics» як файл `grfguide.tex`, доступний із того ж джерела, що і дистрибутив LaTeX. 65

[10] Rainer Schöpf, Bernd Raichle, Chris Rowley. *A New Implementation of LaTeX's verbatim Environments*. Входить в набір пакетів «tools» як файл `verbatim.dtx`, доступний із того ж джерела, що і дистрибутив LaTeX. 69

[11] Graham Williams. *The TeX Catalogue* — детальний список багатьох пакетів, що мають відношення до TeX and LaTeX. Доступний за адресою `CTAN:/tex-archive/help/Catalogue/catalogue.html`.

[12] Keith Reckdahl. *Using EPS Graphics in LaTeX 2ε Documents*, що пояснює усе і навіть більше, ніж ви захочете знати, про файли EPS і як їх використовувати у документах LaTeX. Доступний за адресою `CTAN:/tex-archive/info/epslatex.ps`. 65

[13] Kristoffer H. Rose. *XY-pic User's Guide*. Доступний на CTAN з дистрибутивом XY-pic

[14] Vladimir Volovich, Werner Lemberg and LaTeX3 Project Team. *Cyrillic languages support in LaTeX*. Входить в дистрибутив LaTeX 2ε як файл `cyrguide.tex`. 27

[15] John D. Hobby. *A User's Manual for MetaPost*. Downloadable from `http://cm.bell-labs.com/who/hobby/` 84

[16] Alan Hoenig. *TeX Unbound*. Oxford University Press, 1998, ISBN 0-19-509685-1; 0-19-509686-X (pbk.) 84

[17] Urs Oswald. *Graphics in LaTeX 2ε*, containing some Java source files for generating arbitrary circles and ellipses within the `picture` environment, and *MetaPost - A Tutorial*. Both downloadable from `http://www.ursoswald.ch` 84, 87, 92

[18] Андрій Швайка. *TeX & Ukrainian*. На цій сторінці ви знайдете інформацію про TeX і його можливі види, а також як його навчити української мови. Адреса: `http://ph.icmp.lviv.ua/~ashv/tex/TeXandUkrainian.koi.html`. 28

Покажчик