КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА

А.П. Креневич, В.А. Бородін

ВИДАВНИЧА СИСТЕМА LaTeX

МЕТОДИЧНІ ВКАЗІВКИ ДО ЛАБОРАТОРНИХ ЗАНЯТЬ
З ДИСЦИПЛІНИ "ПРАКТИКУМ НА ЕОМ"

Видавничо-поліграфічний центр
"Київський університет"
2007

Видавнича система LaTeX:Методичні вказівки до лабораторних занять з дисципліни "Практикум на ЕОМ"/ А.П. Креневич, В.А. Бородін– К.: ВПЦ "Київський університет", 2007. – 49 с.

Рецензенти

Доктор фіз.-мат. наук проф. Станжицький О.М.
Кандидат фіз.-мат. наук доц. Ловейкін А.В.

Затверджено Вченою Радою
механіко–математичного факультету
14 травня 2007 р.

Методична розробка присвячена видавничій системі LaTeX, яка, використовується для оформлення наукових текстів, зокрема математичних. Матеріал розбито по розділах, кожен з яких розрахований на одне або декілька занять, наведено велику кількість прикладів.

Для студентів механіко-математичного факультету

Зміст

Розділ 1

Основи роботи з LaTeX

1.1 Вступ

LaTeX – це комп'ютерна видавнича система. Основне її призначення – підготовка наукових документів. Дана система була розроблена на базі системи TeX.

Наряду з LaTeX існують системи PlainTeX, $\mathcal{AMS}$-TeX.

Переваги LaTeX

1) Дана система надає автору гнучкі та зручні засоби для створення документів високої типографської якості.

2) Всі видавничі системи на базі TeX можуть працювати на будь-якому комп'ютері, незалежно від його потужності та операційної системи.

3) Файли, створені у LaTeX в більшості випадків мають менший об'єм порівняно з файлами, створеними іншими видавничими системами та текстовими редакторами.

Недоліки LaTeX

1) Для друку документів на папері LaTeX потребує принтер високої якості.

2) Він не є системою типу WYSIWYG (що друкую – те й бачу): створення документу та перегляд того, як виглядає документ при друці є різними операціями.

1.2 Робота з системою LaTeX

Для створення документу з допомогою LaTeX, автор повинен підготувати файл з текстом та командами оформлення цього тексту. Надалі цей файл будемо називати *вихідними файлом*. Цей файл створюється за певними правилами, які будуть розглянуті нижче. На даному етапі потрібно зауважити, що вихідний файл можна створювати з допомогою будь-якого текстового редактора. Вихідний файл повинен містити виключно текстову інформацію. Розширення вихідного файлу повинно бути `.tex`.

Подальша робота з документом здійснюється в два етапи.

1) Створений вихідний файл компілюється програмою-транслятором. В результаті компіляції буде створено новий файл з тим же ім'ям, що і вихідний файл, але з розширенням `.dvi`.

2) Отриманий dvi-файл можна переглянути на роздрукувати за допомогою спеціальної програми, що називається `dvi`-драйвером.

Для перегляду вже створеного документу досить мати тільки dvi-файл. Але для зміни документу необхідно мати вихідний файл.

1.3 Компіляція файлів LaTeX

Як було сказано вище, після створення вихідного файлу його потрібно відкомпілювати. Під час компіляції на екрані з'являється вікно в якому від-

ображається інформація, що стосується даного вихідного файлу та його процесу компіляції. У випадку якщо процес компіляції успішно завершується це вікно зникає з екрану.

Зауваження. Інформація про результат компіляції вихідного файлу записується у файл з тим же ім'ям, що і вихідний файл але з розширенням `.log`.

Якщо вихідний файл створено з порушеннями правил LaTeX, то процес компіляції призупиняється на тому етапі де LaTeX виявив першу помилку. У такому разі ви можете виправити цю помилку прямо у вікні компіляції або проігнорувати її натиснувши клавішу Enter і продовжити процес компіляції. Знайшовши нову помилку, LaTeX знову призупинить процес компіляції. І так буде відбуватись поки вихідний файл не буде повністю відкомпільовано.

Зауваження. У випадку, якщо процес компіляції завершився з помилками, ймовірніше за все документ буде відображатися не правильно. Тому необхідно внести виправлення у вихідний файл і повторити процес компіляції знову.

Часто одна помилка у вихідному файлі призводить до виникнення інших помилок, а значить процес виправлення помилок повинен носити поступовий характер. Виправте першу помилку і повторіть процес компіляції.

Наступні команди використовуються для боротьби з помилками під час компіляції

- Enter – компілятор проігнорує поточну помилку.
- h → Enter – компілятор надасть допоміжну інформацію стосовно помилки, що виникла.
- x → Enter – компілятор негайно припинить процес компіляції.
- i → Enter – компілятор дозволить виправити помилку безпосередньо у вікні компіляції.
- r → Enter – компілятор проігнорує всі помилки, що будуть виникати в поточному процесі компіляції.

1.4 Структура документу

Кожен LaTeX-файл починається з *заголовку*. Заголовок – це команда

```
\documentclass{article}
```

яка задає режим оформлення документу. Слово `article` в фігурних дужках вказує, що документ буде оформлений, як стаття. Інші варіанти оформлення текстів будуть розглянуті пізніше.

Зауваження. Потрібно враховувати, що LaTeX розрізняє великі та маленькі літери.

Після заголовку LaTeX-файлу іде фрагмент коду, який називається *преамбулою документу*. У преамбулі розміщуються параметри та команди (такі як розмір шрифтів, відстань між рядками і т.д.), що відносяться до всього документу. Також можливе підключення за допомогою команди `\usepackage{ім'я_пакету}` додаткових пакетів для роботи зі шрифтами, текстом, формулами, графікою і т.ін.

Текст документу розміщують після преамбули і він міститься між командами

```
\begin{document}
та
\end{document}
```

Все, що буде надруковано у файлі після команди `\end{document}` компілюватися не буде

Таким чином можемо навести код мінімального LaTeX-документу

```
\documentclass{article}
\begin{document}
\LaTeX{}-document
\end{document}
```

Зауваження. Для друкування текстів з використанням кириличних літер необхідно підключати в ваш документ додаткові пакети. Для цього в преамбулу документу включають команди, на зразок наступних

```
\usepackage[cp1251]{inputenc}
\usepackage[ukrainian,russian]{babel}
```

Нижче наведено приклад коду вихідного файлу з підключеними пакетами.

```
\documentclass{article}
\usepackage[cp1251]{inputenc}
\usepackage[ukrainian,russian]{babel}
\title{Зразок текстового документу}
\author{Студент мех.-мат факультету ...}
%date{}
\begin{document}
\maketitle
\begin{abstract}
Наш перший \LaTeX{}-документ
\end{abstract}
```

```
Ми побачимо  {\huge великі} можливості цього редактора.
\end{document}
```

1.5 Контрольні запитання

- Чим поступається LaTeX іншим видавничими системами? В чому його переваги?
- Що таке вихідний файл?
- Для чого потрібно робити компіляцію вихідного файлу? Що потрібно зробити, щоб відкомпілювати вихідний файл?
- Як переглянути результат роботи з системою LaTeX?
- Яка структура вихідного LaTeX-файлу?

Розділ 2

Важливі поняття.

2.1 Спеціальні символи

Більшість символів у вихідному tex-файлі означають те, що буде надруковано. Наступні символи мають особливий статус і якщо ви просто так використаєте їх в тексті вихідного файлу то, скоріше за все, під час компіляції виникнуть помилки або у отриманому документі текст буде відображатись не так, як ви планували.

- {, } – обмежують групи в вихідному файлі.
- $ – обмежує математичні формули.
- & , # – спеціальні символи, про які буде сказано пізніше.
- % – коментар, весь код рядка після даного символу під час компіляції буде проігноровано.
- _, ^ – використовуються при наборі математичних формул, для створення нижнього і верхнього індекса відповідно.
- $\sim$ – не розривний пропуск.
- $\backslash$ – вказується при використанні команд чи параметрів.

Зауваження. Якщо в документі вам необхідно надрукувати один з семи перших вище перерахованих символів, то у вихідному файлі надрукуйте перед даним символом знак "\"(backslash).

Для друкування решти символів використовуються спеціальні команди: для друку ^ – команда `^{}`, для $\sim$ – `$\sim$`, а для символу $\backslash$ використовується `$\backslash$`.

2.2 Команди

Команди в LaTeX використовуються для додаткового оформлення документів. Всі команди в LaTeX починаються з символа "\"(backslash). Команди бувають двох типів: команди, у яких після символа "\" іде символ, котрий не є літерою, і команди, у яких після символа "\" іде набір літер, що називається іменем команди. В імені команди, а також після символа "\" не повинно бути пропусків, а також ім'я команди не можна розміщувати в декількох рядках. В іменах команд великі і маленькі символи розрізняються. Після імені команди у вихідному файлі обов'язково повинен стояти пропуск.

Деякі команди мають *аргументи*. Аргументи – це інформація уточнюючого характеру, яка вказує яким чином має працювати команда. Аргументи вказують після імені команди.

Аргументи бувають обов'язковими і не обов'язковими. Не обов'язкові аргументи розміщують у квадратних дужках після імені команди і їх можна опускати. Обов'язкові аргументи розміщують у фігурних дужках до або після необов'язкових аргументів в залежності від специфіки команди. Якщо опустити обов'язковий аргумент, під час компіляції виникне помилка. Якщо необов'язкових аргументів декілька, то вони перераховуються через кому. Кожен обов'язковий аргумент розміщується в окремих фігурних дужках.

Приклад.

```
\documentclass[12pt, twocolumn]{article}
```

2.3 Групи

Поняття груп одне з найважливіших понять LaTeX. Вони використовуються для оформлення деякого фрагменту документа однаковим чином. Фрагмент документу, що входить в групу береться в фігурні дужки. Самі по собі фігурні дужки не генерують ніякого тексту. Їх єдине призначення – обмеження групи. Команди, які використовуються всередині групи впливають виключно на саму групу і не впливають на решту тексту вихідного файлу. Групи можна вкладати одна в одну.

Приклад.

```
Весь текст буде надруковано звичайним шрифтом, {\bf а даний
фрагмент напівжирним}
```

Тут команда `\bf` впливає тільки на групу, яка є фрагментом тексту "а даний фрагмент напівжирним". Надрукуйте даний фрагмент у TeX, проаналізуйте результат.

Фігурні дужки у вихідному файлі повинні бути збалансованими, тобто кожній відкритій фігурній дужці повинна відповідати закрита фігурна дужка. В іншому разі при компілювання виникнуть помилки.

Існують команди, які називають глобальними. Вони зберігають свою дію і за межами групи.

2.4 Параметри

Параметри, аналогічно до команд, починаються з символа "\" і використовуються для задавання різних величин (таких як ширина сторінки, розмір шрифта, тощо), що враховуються при оформленні документа. LaTeX, в більшості випадків, самостійно встановлює всі необхідні для оформлення документа параметри. Якщо запропоновані LaTeXом параметри вас не

задовольняють, їх можна змінити. Наприклад, якщо ви бажаєте встановити величину абзацного відступу 2 сантиметри, то в преамбулі документу потрібно надрукувати

```
\parindent = 2cm
```

Для зміни інших параметрів потрібно діяти аналогічним чином. Вкажіть ім'я параметра, і після знака рівності значення, якого повинен набути цей параметр.

2.5 Оточення

Ще одна важлива конструкція LaTeX – це *оточення*.

Оточення – це фрагмент файлу, який починається з тексту

```
\begin{Ім'я_оточення}
```

Де `Ім'я_оточення` – перший обов'язковий, і можливо не єдиний аргумент команди `\begin`. Закінчується оточення командою

```
\end{Ім'я_оточення}
```

Кожній команді `\begin`, що відкриває оточення повинна відповідати команда `\end` з тим самим ім'ям. Кожне оточення є групою.

Приклад.

```
\begin{center}
Даний текст буде розміщено по середині сторінки
\end{center}
```

Зауваження. Деякі оточення можуть мати аргументи. Ці аргументи вказуються при відкритті оточення після команди `\begin{Ім'я_оточення}`. При закритті оточення, тобто після команди `\end{Ім'я_оточення}`, аргументи вказувати не потрібно.

2.6 Контрольні запитання

- Що таке спеціальні символи, для чого вони використовуються? Чи можна спеціальні символи відображати в LaTeX-документі?
- Чим команди відрізняються від параметрів? Для чого використовуються аргументи команд і параметрів? Наведіть приклад команд і параметрів?
- Що таке група?
- Що таке оточення? Чи є оточення групою?

Розділ 3

Основи верстки документів

3.1 Стилі

Команда `\documentclass` з якої починається будь-який LaTeX-файл має один обов'язковий аргумент – назва основного стилю, котрий вказується у фігурних дужках після команди \documentclass:

- article – даний стиль застосовують до оформлення статей;
- report – використовують для оформлення великих статей, розбитих на розділи або для невеликих книжок чи брошур.
- book – використовують для оформлення книжок.
- letter – для оформлення невеликих документів, наприклад, листів.

Кожний з основних стилів має опції, котрі надалі будемо називати стильовими опціями. Стильові опції вказуються перед назвою стилю в квадратних дужках.

- 10pt, 12pt – вказує на те, що основний текст буде друкуватись кеглем 10 або 12 розміру відповідно. Якщо даний аргумент відсутній, то текст буде друкуватись 10 кеглем.
- twoside – задає оформлення документа з різними полями на парних і непарних сторінках (так зване, "двостороннє" оформлення документу). Для стилю book дана опція встановлюється автоматично.
- twocolumn – використовується, якщо необхідно оформити документ у дві колонки.
- fleqn – вказує на те, що виключні формули, що входять в документ будуть друкуватись не в центрі сторінки, а з лівого боку.
- leqno – вказує на те, що нумерація виключних формул буде не по правому, а по лівому краю сторінки.
- a4paper, a5paper – використовується для оформлення документа розміром сторінки A4 або A5 відповідно.

3.2 Нумерація сторінок

Для нумерації сторінок використовується команда `\pagestyle`, яка вказується у преамбулі документа. Ця команда має один обов'язковий аргумент:

- `empty` – номери сторінок не друкуються.
- `plain` – номери сторінок вказуються знизу, посередині рядка.
- `headings` – номери сторінок вказуються вгорі сторінки.

3.3 Одиниці довжини

Деякі параметри, що задаються в командах є одиницями довжини, наприклад, розмір абзацу, тощо. Наведемо основні одиниці довжини, які використовуються для задавання розмірів.

`pt`	пункт $\approx$ 0.35 міліметра
`pc`	піка = 12 пунктів
`mm`	міліметр.
`cm`	сантиметр = 10міліметрів
`in`	дюйм $\approx$ 25.4 міліметра

Зауваження. Якщо довжина, яку ви вказуєте, дорівнює нулю, то все одно потрібно вказувати одиницю довжини. В противному випадку LaTeX видасть повідомлення про помилку.

3.4 Поля та розмір сторінки

Стандартні стилі самостійно встановлюють значення таких параметрів, як розміри полів, ширина та висота сторінки. Якщо ці значення вас не задовольняють, їх можна змінити. Для цього в преамбулі документу потрібно вказати один (або всі) з перерахованих нижче параметрів з потрібними аргументами.

Висота і ширина тексту

- `\textwidth` – задає ширину тексту на сторінці.
- `\textheigh` – задає висоту тексту на сторінці.

Поля документу

Спосіб задавання лівого краю документу залежить від того чи є оформлення документу одностороннім чи двостороннім.

- `\oddsidemargin` – задає величину відступу від лівого краю кожної сторінки при односторонньому наборі. При двосторонньому наборі дана команда встановлює величину лівого відступу на сторінках з непарними номерами.

- `\evensidemargin` – використовується для встановлення розміру лівого відступу на сторінках з непарними номерами.

- `\topmargin` – даний параметр задає розмір відступу від верхнього краю сторінки.

Приклад.

```
\oddsidemargin = 2cm
\topmargin = 2.5cm
\textwidth = 16cm
\textheigh = 25cm
```

Зауваження.За замовчуванням відступ зліва та зверху документа дорівнює 1 дюйму (1 in $\approx$ 2.54 cm). Значення вище перерахованих параметрів є величинами зсуву відносно цього базового значення.

Наприклад, `\topmargin=-0.54cm` встановлює верхнє поле величиною 2 сантиметри.

Зсув сторінки в цілому

Інколи при друці можна виявити, що реальні розміри полів документу не такі, як було задано з допомогою параметрів `\oddsidemargin` чи `\topmargin`. Це може бути пов'язано з індивідуальними особливостями принтера. Для подолання даної проблеми можна просто змінити розміщення сторінки в цілому на друкованому аркуші. Для цього в преамбулі документу встановлюють наступні параметри з відповідними значеннями.

- `\hoffset` – зсув усієї сторінки, під час друку, на деяку величину вправо.

- `voffset` – зсув усієї сторінки, під час друку, на деяку величину вниз.

Зауваження. Тут ви можете вказувати від'ємні значення параметрів. При цьому буде здійснюватись зсув в протилежну сторону на відповідну величину, наприклад, якщо в преамбулі документу встановлено наступні параметри

```
\hoffset = -5mm
\voffset = 3mm
```

то при друці весь текст буде зсунуто на 5 міліметрів вліво і на 3 міліметри вниз.

3.5 Розділи документу

Ви можете самостійно розбивати ваш документ на розділи і параграфи. При цьому відповідальність за нумерацію розділів, параграфів, пунктів LaTeX повністю покладає на вас. Але працюючи з LaTeX краще створювати заголовки і нумерацію розділів автоматично засобами LaTeX.

Для оформлення розділів існують наступні команди

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

В цьому переліку кожна наступна команда означає більш дрібний підрозділ ніж попередня. Слід зауважити, що команда `\chapter` для стандартного стиля `article` не визначена.

Стандартні стилі забезпечують автоматичну нумерацію розділів, при якій дрібніші розділи підкоряється більшому, наприклад, коли починається новий розділ `\section`, нумерація розділів `\subsection*` починається спочатку. Виключенням з цього правила є тільки команда `\part`.

Кожна з команд має один обов'язковий параметр – ім'я розділу, яке вказується в фігурних дужках після назви відповідного розділу.

Вправа. Надрукуйте наступний фрагмент коду. Проаналізуйте отриманий результат.

```
\part{Все про \LaTeX}
\section{Вступ}
\subsection*{Основи роботи}
\subsection*{Основні прийоми}
\section{Основи верстки документів}
\subsection*{Стилі}
\subsection*{Нумерація сторінок}
\subsection*{Поля та розмір сторінки}
\subsubsection{Висота і ширина документу}
\subsubsection{Поля документу}
```

Для того, щоб почати новий розділ, не нумеруючи його, використовують варіант відповідної команди вказуючи символ "*"після імені команди.

Вправа. Надрукуйте наступний фрагмент коду. Порівняйте отриманий результат з результатом отриманим у попередній вправі

```
\part{Все про \LaTeX}
\section{Вступ}
\subsection**{Основи роботи}
\subsection**{Основні прийоми}
\section{Основи верстки документів}
```

```
\subsection**{Стилі}
\subsection**{Нумерація сторінок}
\subsection**{Поля та розмір сторінки}
\subsubsection*{Висота і ширина документу}
\subsubsection*{Поля документу}
```

3.6 Контрольні запитання

- Для чого використовуються стилі оформлення документів?
- Які основні стилі та їх опції ви знаєте?
- Назвіть основні одиниці довжини LaTeX. Для чого вони використовуються?
- Які атрибути та параметри керують розміром тексту на сторінці?
- Назвіть основні команди для розбиття тексту на розділи та підрозділи. Наведіть приклад коду тексту розбитого на розділи та підрозділи.

Розділ 4

Робота з абзацами

Для того, щоб LaTeX оформив абзац досить вставити в вихідному файлі порожній рядок, що буде вказувати на закінчення абзаца. В даному розділі поговоримо про додаткове оформлення абзаців.

4.1 Перенесення слів на інший рядок

Для більш стилізованого відображення тексту і скорочення інтервалів між словами при форматуванні абзацу, LaTeX може переносити частину слова (за правилами відповідної мови на якій друкується текст) на інший рядок.

Зауваження. За замовчуванням українська і російська мови, як правило, не підключені до LaTeX-компілятора. Тому, під час компіляції LaTeX не буде робити перенесення слів у документах надрукованих українською чи російською мовами. Для виходу з даної ситуації потрібно або змінити параметри в настройках компілятора або вказати LaTeX спосіб перенесення деяких слів самостійно.

Для налаштування способу перенесення слів, у вихідному файлі використовують наступні параметри

- `\righthyphenmin` – параметр, що відповідає за мінімальну кількість літер, яка може бути перенесена на наступний рядок.

Таким чином якщо ви надрукуєте в документі команду

```
\righthyphenmin=2
```

LaTeX дозволятиме перенесення двох останніх літер слова на наступний рядок. За замовчуванням значення даного параметру встановлено 3.

- `\-` – використовується для одноразового вказування LaTeXу яким чином робити перенесення в словах.

Наприклад, якщо в тексті ви надрукуєте `{пе\-ре\-не\-сен\-ня}` то LaTeX буде розуміти, що перенесення можна робити тільки і тих місцях, де стоїть команда `\-`, навіть якщо це суперечить LaTeX-овському алгоритму перенесення.

- `\hyphenation`. Якщо слово, яке потрібно переносити всупереч LaTeX-овському алгоритму перенесення, зустрічається неодноразово ви можете використати дану команду, щоб задати необхідний спосіб перенесення цього слова у всьому документі.

Наприклад, якщо ви в преамбулі документа надрукуєте

```
\hyphenation{доку-мент}
```

то слово "документ" LaTeX зможе розривати тільки в тому місці, де стоїть дефіс. Якщо аргументом команди `\hyphenation`, буде слово, що не містить дефісів, то це означатиме, що дане слово взагалі не можна розривати. Аргументом команди `\hyphenation` може бути декілька слів. У цьому разі їх потрібно розділяти пропусками або символами кінця рядка.

У випадку, якщо потрібно оформити текст без розривів слів використовують наступні команди у преамбулі документу

`\sloppy` – вирівнювання тексту по ширині документу; величина пропуску між словами при цьому може бути досить великою.

`\fossy` – встановлює режим при якому величина пропуску між словами є малою; при цьому рядок може виходити за межі документу.

4.2 Розриви рядків

Як було сказано, LaTeX самостійно форматує абзац. При цьому він сам вирішує яким чином абзац розбивати на рядки. Іноді виникає ситуація коли необхідно штучно вплинути на те, в якому місці LaTeX почне новий рядок.

Нерозривний пробіл

Дана команда використовується для того, щоб вказати LaTeXу у якому місці абзацу не можна починати новий рядок. Позначається ця команда символом "~". Таким чином, якщо у вихідному файлі ви надрукуєте фразу "рівняння~(1)", то у відкомпільованому документі дана фраза буде гарантовано розміщуватись в одному рядку.

Заборона перенесень слів

У випадку, якщо вам потрібно заборонити розбиття слова для перенесення його частити у наступний рядок у всьому документі ви можете використовувати вище описану команду `\hyphenation`. Але у випадку, якщо вам потрібно заборонити LaTeXу переносити деяке слово тільки в одному місці документу зручно використовувати команду `\mbox`. Ця команда має один обов'язковий аргумент – слово або будь-який фрагмент тексту. LaTeX буде розуміти даний фрагмент як один символ. Таким чином розбиття даного фрагменту на декілька рядків буде неможливе.

Примусове розірвання рядка

Якщо в деякому місці абзаца потрібно почати новий рядок, не починаючи при цьому нового абзаца використовують наступні команди

- `\\` – у випадку використання даної команди буде отримано не розтягнутий по всій ширині рядок.
- `\linebreak` – команда аналогічна попередній, за виключенням того, що розірваний рядок буде вирівняно по ширині документа.

Зауваження. Наведені вище команди мають необов'язковий параметр. Якщо в квадратних дужках, після імені команди вказати деяку довжину, то після розірваного рядка буде зроблено вертикальний відступ вказаної довжини.

Приклад.

```
Після цього рядка \\[3mm] зроблено додатковий
вертикальний відступ довжиною 3 міліметри.
```

4.3 Вертикальні відступи

Відстань між абзацами

Відстань між абзацами може бути змінена з допомогою команд `\smallskip`, `\medskip`, `\bigskip`. Тут ці команди розміщені у порядку збільшення відстані між абзацами. Для створення відступу потрібної величини одну з перерахованих команд розміщують або після тексту абзаца або безпосередньо після символа кінця абзацу (наприклад після порожнього рядка).

Величина відстані між абзацами, що встановлюється однією з перерахованих вище команд залежить від стилю оформлення документа. Тому, якщо ви хочете задати величину вертикального відступу між абзацами в явному вигляді зручно користуватись командою `\vspace`. Дана команда має один обов'язковий аргумент – відстань між абзацами. Наприклад, якщо після абзацу буде вказано команду `\vspace{5mm}`, то це буде означати, що потрібно збільшити величину вертикального відступу між абзацами додатково на 5 міліметрів.

Перераховані вище команди діють виключно на поточний абзац. Для встановлення додаткового вертикального відступу між всіма абзацами документа використовується параметр `\parskip`, який, як правило розміщують в преамбулі документа. Припустимо, якщо в преамбулі документу ви вкажете наступну команду `\parskip=3mm`. Це означатиме, що LaTeX збільшить вертикальну відстань між абзацами у всьому документі на три міліметри.

Зауваження. Аргументом команди `\vspace`, і значенням параметра `\parskip` можуть бути від'ємні величини. У такому разі вертикальний абзацний відступ буде не збільшуватись, а зменшуватись на відповідну величину.

У команди `\vspace` є варіант із зірочкою після імені команди. Якщо, наприклад, надрукувати `\vspace*{1cm}`, то буде створено проміжок величиною 1 сантиметр, що не зникне навіть у тому випадку, якщо в цьому місці відбудеться розрив сторінки.

Відстань між рядками абзацу

Всі стандартні стилі LaTeX самостійно встановлюють величину відстані між рядками одного абзацу. Якщо ж, всупереч цьому виникає необхідність змінити цю відстань, то одним із способів це здійснити – розмістити в преамбулі документу наступну команду

`\linespread=x`,

де x – величина міжрядкового інтервалу, наприклад 1.5.

Для зміни міжрядкового інтервалу всередині групи можна використати команду

```
\renewcommand{\baselinestretch}{x}
```

де число x - відношення нової величини відстані між рядками до стандартної величини, встановленої в базовому стилі. Тобто, якщо $x = 1.1$, то це означає, що потрібно збільшити величину відстані між рядками в 1.1 рази (на 10%).

4.4 Розриви сторінок

Під час форматування тексту LaTeX знаходить найбільш оптимальний, з його точки зору, спосіб розбиття документу на сторінки. Але не завжди цей спосіб є допустимим з точки зору автора документу чи правил типографії.

Заборона розриву

Для того, щоб заборонити розрив сторінки використовується команда `\nopagebreak`. Якщо поставити її в кінці абзацу, то розрив сторінки після даного абзацу буде заборонено.

Примусовий розрив

Для примусового розірвання сторінки ви можете використовувати одну з наступних команд:

- `\newpage` або `\pagebreak` – під дією однієї з цих команди сторінка закінчується і решта простору, що залишилась до кінця сторінки заповнюється порожнім простором.

- `\clearpage` – працює аналогічно до наведених вище команд, хоча має деякі особливості при роботі з зображеннями і таблицями.

- `\cleardoublepage` – команда аналогічна попереднім, за виключенням того, що в деяких стилях (наприклад `book`) нова сторінка починатиметься з непарного номера. У випадку, якщо дана команда завершує сторінку з непарним номером, то LaTeX створить порожню сторінку з парним номером.

Створення порожньої сторінки

Для створення порожньої сторінки не досить двічі підряд надрукувати одну з перерахованих в попередньому пункті команд. LaTeX все одно буде відображати документ так, наче там надруковано тільки одну команду розриву сторінки. Тому для друкування порожньої сторінки його приходиться "обманювати" вставивши в нову сторінку деякий символ який відображатись не буде, але вказувати, що сторінка не порожня. Наприклад.

```
\pagebreak \mbox{} \pagebreak
```

4.5 Вирівнювання у абзаці

Відсутність абзацного відступу

Іноді виникає ситуація, коли потрібно надрукувати абзац без абзацного відступу. Для цієї мети зручно використовувати команду `\noindent`. У тому абзаці, в якому відступ не потрібно робити дана команда повинна іти перед текстом. Таким чином команда `\noindent` впливає тільки на ті абзаци які з неї починаються.

Цитати

Для оформлення фрагменту тексту відсунутого від країв документу використовують оточення "`quote`"("цитата").

```
\begin{quote}
    Даний текст є цитатою
\end{quote}
```

Для довгих цитат, що складаються з декількох абзаців краще використовувати оточення "`quotatio`". Дане оточення абсолютно аналогічне попередньому за виключенням того, що в тексті оформленому з допомогою цього оточення робиться абзацний відступ.

Розміщення тексту по центру

Для центрування тексту використовують оточення "`center`". У випадку, якщо центрувати потрібно невеликий фрагмент тексту, що складається з одного рядка використовують команду `\centerline`. Обов'язковим аргументом цієї команди буде відповідний фрагмент тексту.

Вирівнювання по лівому та правому краях документу

Для вирівнювання тексту по лівому та правому краях документа використовують оточення "`flushleft`" і "`flushright`" відповідно.

4.6 Контрольні запитання

- Яким чином Tex форматує абзац? Для чого Tex розбиває слова при перенесенні?
- Що відповідає за перенесення та розбиття слів?
- Чи можна вказати свій спосіб розбиття слова всупереч алгоритму, котрим керується tex при перенесені слів? Які команди для цього використовуються?
- Що означає термін "нерозривний пробіл"? Яка команда відповідає за друкування "нерозривного пробілу"?
- Чи можна Tex "примусити"завжди розміщувати деяке слово в одному рядку? Наведіть приклад.
- Як почати новий рядок в межах одного абзацу?
- З допомогою яких команд можна змінювати відстань між абзацами та рядками окремого абзацу?
- З допомогою яких команд можна починати нову сторінку? Які особливості кожної з них?
- Які способи вирівнювання тексту ви знаєте? Які команди за це відповідають?

Розділ 5

Робота зі списками в LaTeX

Для створення списків у LaTeX використовуються наступні оточення

- `itemize` – оточення для створення маркованих списків.
- `enumerate` – оточення для створення нумерованих списків.

Кожен новий елемент списку починається з команди `\item`.

Зауваження. У документах підготовлених з допомогою LaTeX допускається вкладення одних списків у інші. При цьому максимальний рівень вкладення не повинен перевищувати чотирьох.

5.1 Марковані списки

Розглянемо процес створення багаторівневих маркованих списків на наступному прикладі.

```
\begin{itemize}
\item Перший запис першого рівня.
  \begin{itemize}
    \item Перший запис другого рівня.
      \begin{itemize}
        \item Третій рівень.
          \begin{itemize}
            \item Четвертий рівень.
          \end{itemize}
        \end{itemize}
     \item Другий запис другого рівня.
    \end{itemize}
  \item Другий запис першого рівня.
 \end{itemize}
```

Результатом даного коду буде наступний фрагмент документу

- Перший запис першого рівня.
  - Перший запис другого рівня.
    - Третій рівень.
      - Четвертий рівень.
  - Другий запис другого рівня.
- Другий запис першого рівню.

Команда \item, для маркованих списків, має один необов'язковий параметр – вигляд маркера який передує елементу списку. Вигляд маркерів, що використовуються за замовчуванням у багаторівневих списках можна побачити з попереднього прикладу.

Приклад.

```
\begin{itemize}
 \item[\$] Перший запис.
 \item[ *] Другий запис.
\end{itemize}
```

Результатом виконання попереднього коду буде наступний фрагмент документу

$ Перший запис.

* Другий запис.

Якщо вигляд маркерів у багаторівневих маркованих списках, що використовуються за замовчуванням, вас не влаштовує ви можете його змінити не тільки з допомогою необов'язкового параметра. Для цього використовують наступну команду

```
\renewcommand{рівень}{символ}
```

де

рівень – одна з чотирьох команд \labelitemi, \labelitemii, \labelitemiii чи \labelitemiv що відповідає за вигляд маркера відповідно першого, другого, третього чи четвертого рівня.
символ – символ, що буде маркером.

Зауваження. Якщо вказати вище наведену команду у преамбулі документу, то вигляд маркера буде змінено у всьому документі. В іншому разі вигляд маркера буде змінено виключно для списків документу, що йдуть після цієї команди або всередині групи.

5.2 Нумеровані списки

Для створення нумерованих списків використовується оточення enumerate. Розглянемо його використання на наступному прикладі.

```
\begin{enumerate}
   \item Перший запис першого рівня.
   \begin{enumerate}
      \item Перший запис другого рівня.
      \begin{enumerate}
```

```
        \item Третій рівень.
          \begin{enumerate}
            \item Четвертий рівень.
          \end{enumerate}
        \end{enumerate}
      \item Другий запис другого рівня.
    \end{enumerate}
  \item Другий запис першого рівню.
 \end{enumerate}
```

1. Перший запис першого рівня.
    (a) Перший запис другого рівня.
        i. Третій рівень.
            A. Четвертий рівень.
    (b) Другий запис другого рівня.
2. Другий запис першого рівню.

Команда `\item`, для нумерованих списків, має необов'язковий аргумент. З допомогою цього аргументу можна змінювати нумерацію окремих елементів списку. Наприклад в результаті компіляції коду

```
\begin{enumerate}
   \item[23] Перший запис.
   \item[25] Другий запис.
 \end{enumerate}
```

в документі буде надруковано

23 Перший запис.

25 Другий запис.

5.3 Використання пакету `enumerate`

Існує більш гнучкий спосіб для оформлення нумерованих списків. Він базується на використанні пакету `enumerate`. Для того, щоб підключити пакет `enumerate`, слід у преамбулі документу вказати команду `\usepackage{enumerate}`.

Цей пакет дозволяє створювати власний стиль оформлення списків. Розглянемо його застосування на наступному прикладі:

```
\begin{enumerate}[\No 1).]
  \item Перший запис
  \item Другий запис
    \begin{enumerate}[Запис i:]
     \item Другий рівень
     \item Ще один запис другого рівня
    \end{enumerate}
 \end{enumerate}
```

Тут на першому рівні у квадратних дужках ми вказуємо зразок маркеру першого запису №1)., а далі Tex автоматично буде аналогічним чином створювати наступні записи.

На другому рівні ми вказуємо зразок першого запису вигляду "Запис i:]". Далі Tex буде нумерувати таким же чином наступні записи другого рівня. Результат буде наступним:

№1). Перший запис

№2). Другий запис

Запис i: Другий рівень

Запис ii: Ще один запис другого рівня

5.4 Список літератури

Створення списку

Одним зі способів створення списку літератури є використання оточення thebibliography. Розглянемо його застосування на наступному прикладі.

```
\renewcommand{\bibname}{Література}
\begin{thebibliography}{aa}
   \bibitem{dal_kreyn}\textit{Далецкий Ю.Л., Крейн М.Г.}
   Устойчивость решений дифференциальных уравнений в банаховом
   пространстве. -М.:Наука, 1970.
   \bibitem{dor}\textit{Дороговцев А.Я} Периодические и
   стационарные режимы бесконечномерних детерминированых и
   стохастических динамических систем.-К.:Вища школа, 1992.
   \bibitem{tsarkov}\textit{Царьков~Е.Ф.} Случайные возмущения
   функционально диференциальных уравнений.-Рига:Зинатне,1989.
\end{thebibliography}
```

В результаті наведеного вище коду, LaTeX сформує наступний список літератури.

Література

[1] *Далецкий Ю.Л., Крейн М.Г.* Устойчивость решений дифференциальных уравнений в банаховом пространстве. -М.:Наука, 1970.

[2] *Дороговцев А.Я* Периодические и стационарные режимы бесконечномерних детерминированых и стохастических динамических систем.-К.:Вища школа, 1992.

[3] *Царьков Е.Ф.* Случайные возмущения функционально диференциальных уравнений.-Рига:Зинатне,1989.

Команда `\renewcommand{\bibname}{Література}` використовується для того, щоб переіменувати заголовок списку літератури, що використовується за замовчуванням. В класі `article` замість цієї команди використовується команда `\refname`. Параметр аа використовується для того, щоб вказати довжину лівої границі краю списку (у даному випадку – 2 символи).

Кожен елемент списку літератури починається з команди `\bibitem`, обов'язковим аргументом якої є мітка, яка ідентифікує елемент списку і через яку здійснюється автоматичне генерування посилань на відповідне джерело в списку літератури.

Генерування посилань

Для генерування посилань на елемент зі списку літератури використовується команда `\cite`. Вона має два аргументи – один обов'язковий і один необов'язковий. Обов'язковий параметр – це ім'я мітки, яке використовується при формуванні списку літератури для ідентифікації елементу списку. Необов'язковий аргумент ставиться перед обов'язковим в квадратних дужках. Він використовується для того, щоб разом з посиланням на джерело зі списку літератури вказати деяку додаткову інформацію, наприклад номери сторінок.

Приклад.

```
З \cite[ст.33]{dal_kreyn} випливає, що простір $H$
можна розбити в пряму суму підпросторів інваріантних
по відношенню до оператора $A$.
```

З [1, ст.33] випливає, що простір H можна розбити в пряму суму підпросторів інваріантних по відношенню до оператора A.

5.5 Контрольні запитання

- Що таке списки? Що таке вкладені списки та яка максимальна кількість рівнів вкладення?

- Яким чином можна змінити маркери в маркованих списках? Як зробити різні маркери для кожного елементу списку?
- Як змінити стиль автонумерації в нумерованому списку за допомогою пакету enumerate?
- Як вказати власну назву списку літератури? Яим чином генеруються посилання на літературу?

Розділ 6

Робота зі шрифтами

6.1 Зміна гарнітури шрифту

Для керування зовнішнім виглядом символів одного шрифту використовуються наступні команди

Команда	Альтернативна	Назва гарнітури
\bf	\textbf{}	**Напівжирний (boldface)**
\it	\textit{}	*Курсив (italic)*
\sl	\textsl{}	*Похилий (slanted)*
\sf	\textsf{}	Рублений (sans serif)
\sc	\textsc{}	Капітель (SMALL CAPS)
\tt	\texttt{}	Імітація др.машинки(typewriter)
\rm	\textrm{}	Звичайний (roman)
\em	\emph{}	*Виділений*

В першому стовпчику таблиці наведено команди, що, як правило, використовуються в групах. Вони не мають аргументів. Тому кожна з цих команд впливає на весь текст, що йде після неї.

У другому стовпчику наведено команди, що є еквівалентними до відповідних команд з першого стовпчика. Кожна з цих команд має обов'язковий аргумент, що є фрагментом тексту, на який впливає команда.

Приклад.

```
\textbf{Цей текст надруковано напівжирним шрифтом}
{\it А цей  -- курсивом}
```

6.2 Розмір шрифту

Для зміни розміру шрифту використовують наступні команди.

Команда	Назва розміру
\tiny	крихітний
\scriptsize	індексний
\footnotesize	маленький
\small	менший від нормального
\normalsize	нормальний
\large	великий
\Large	ще більший
\LARGE	дуже великий
\huge	величезний
\Huge	велетенський

Зауваження. Розмір шрифту, що визначається однією з вище перерахованих команд залежить від стилю оформлення документу та базового розміру шрифту. Нагадаємо (див. п.3.1), що розмір базового шрифту задається необов'язковим параметром (стильовою опцією) команди `\documentclass`.

Можливо також вказати розмір шрифту в групі напряму за допомогою команди вигляду

```
\fontsize{xxpt}{len}\selectfont,
```

де `xx` – розмір шрифта, `len` – відстань між рядками в будь-яких одиницях довжини.

6.3 Контрольні запитання

- Що таке гарнітура шрифтів і як її можна змінити?
- Як змінити розмір шрифту в усьому документі, а як у його частині?
- Як зробити текст одночасно жирним та курсивним?

Розділ 7

Друкування формул

7.1 Основні принципи

В LaTeX-документі розрізняють формули, що входять в текст і "виключні"формули, тобто такі, що розміщуються в окремому рядку. Формули, що входять в текст розміщують між двома знаками $. Виключні формули оточують подвійним знаком $ з обох боків або створюють на базі оточення `equation`. Формулами вважаються як цілі формули, так і окремі символи. Пропуски всередині формули ігноруються. Порожні рядки всередині формули є недопустимими.

Під час форматування абзацу, якщо виникає необхідність, LaTeX розриває формулу, що входить в текст, для розміщення її частини в наступному рядку. Для того, щоб LaTeX гарантовано друкував таку формулу, або її частину в одному рядку (без переносів на наступний рядок) потрібно її оточити фігурними дужками. Наприклад, якщо в тексті надрукувати `${x \in R}$`, то формулу $x \in R$ LaTeX завжди буде розміщувати в одному рядку, навіть, якщо це буде призводити до виходу тексту за межі документу.

Виключні формули LaTeX завжди розміщує в одному рядку, не розриваючи їх навіть коли формула виходить за межі документу. Якщо вам потрібно розмістити виключну формулу в декількох рядках, штучно розбийте її на декілька виключних формул.

Кожна буква в формулі розглядається як ім'я змінної і тому друкується математичним курсивом.

Кожна формула утворює групу.

7.2 Друкування простих формул.

Степені і індекси

Як було сказано в розділі 2, степені і індекси набираються з допомогою символів `^` та `_` відповідно.

Приклад.

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

Якщо індекс чи показник степеня складається більш ніж з одного символу, то його беруть у фігурні дужки.

Якщо у змінної чи виразу є як верхні так і нижні індекси то їх можна вказувати в довільному порядку.

Дроби

Дроби, що позначаються похилою рискою вказуються в формулі безпосередньо за допомогою символа /. Для друкування дробів знаменник і чисельник яких розділяється горизонтальною рискою використовується команда

`\frac`. Дана команда має два обов'язкових аргументи. Перший – чисельник, а другий – знаменник.

Приклад.

```
$\frac{a}{b}$
```

$\frac{a}{b}$

Зауваження. На відміну від виключних формул, чисельник і знаменник дробів, що входять у текст LaTeX друкує меншим шрифтом, ніж сам текст.

Вправа. Надрукуйте дріб `\frac{a}{b}` у виключній формулі та такій, що входить до тексту. Порівняйте результати.

Корені

Для друкування коренів використовують команду `\sqrt`. Дана команда має обов'язковий параметр – підкореневий вираз і необов'язковий параметр – степінь кореня.

Приклад.

```
$\sqrt[n]{x}$
$\sqrt{x^2+y^2}$
```

$\sqrt[n]{x}$
$\sqrt{x^2+y^2}$

Дужки

Круглі і квадратні дужки набираються безпосередньо. Фігурні дужки набираються з допомогою команд `\{` та `\}`. Для інших дужок також є спеціальні команди.

У випадку, якщо фрагмент формули взятої у дужки займає багато місця по вертикалі (за рахунок дробів чи інтегралів) то і дужки мають бути більшого розміру, ніж звичайні. В LaTeX на цей випадок передбачений механізм автоматичного вибору розміру дужок. Для цього перед дужкою що відкривається ставлять команду `\left`, а перед дужкою що закривається – `\right`.

Приклад.

```
$$ e=\lim_{n\to\infty}(1+\frac{1}{n})^n $$
$$ e=\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n $$
```

В першому випадку буде надруковано

$$e=\lim_{n\to\infty}(1+\frac{1}{n})^n$$

У другому

$$e=\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n$$

Якщо перед однією дужкою стоїть команда `\left` а перед іншою стоїть `\right`, то розмір цих дужок буде відповідати висоті фрагмента формули, що міститься між `\left` і `\right`.

Конструкцію з \left і \right можна використовувати не тільки для круглих дужок, а і для інших типів дужок.

Зауваження. Кожній команді \left має відповідати команда \right і навпаки. В противному випадку під час компіляції виникнуть помилки. Разом з тим, LaTeX не вимагає щоб дужки були розставлені осмислено з математичної точки зору, тобто, наприклад, ви можете помістити вираз між круглою і квадратною дужками.

Замість дужки після команд \left чи \right можна поставити крапку. В цьому випадку на місці крапки нічого не буде надруковано, а інша дужка буде необхідного розміру. За допомогою даного прийому можна створити похилу дробову риску збільшеного розміру, або записати систему рівнянь.

Приклад.

```
$\left.\int\limits_a^b
f(x)dx \right/(b-a)$
```

$$\int\limits_a^b f(x)dx \Big/ (b-a)$$

```
\int\limits_a^b\frac12(1+x)^
{-3/2}dx=\left.-
\frac{1}{\sqrt{1+x}}\right|_a^b
```

$$\int\limits_a^b \tfrac{1}{2}(1+x)^{-3/2}dx = -\tfrac{1}{\sqrt{1+x}}\Big|_a^b$$

В деяких випадках розмір дужок приходиться задавати самостійно. Для цього використовують команди \bigl, \Bigl, \biggl і \Biggl для лівих дужок і команди \bigr, \Bigr, \biggr і \Biggr для правих. Перелічені команди вказано за зростанням розміру дужок.

Нумерація формул

В математичних текстах зазвичай для зручності посилань приходиться нумерувати формули. LaTeX дозволяє організувати цю нумерацію таким чином, щоб номери формул і посилання на них генерувались автоматично. Здійснюється це наступним чином.

Виключна формула, яку ви нумеруєте, повинна бути оформлена як оточення equation. Кожна така формула при друці автоматично отримає номер. Для того, щоб на цю формулу в подальшому можна було посилатись, в середині даного оточення потрібно поставити команду \label. Дана команда має єдиний обов'язковий агрумент – ім'я формули.

Посилання на формулу в тексті здійснюється з допомогою однієї з двох команд

- \ref – використовується для автоматичного генерування посилань на номер формули.
- \pageref – використовується для генерування посилань на номер сторінки на котру потрапила формула.

Кожна з команд має один обов'язковий параметр – ім'я формули на яку здійснюється посилання.

Ви можете відмовитись від автоматичної генерації номерів формул, а встановлювати їх самостійно. Для цього можна користуватись командами `\eqno` або `\leqno`. Перша команда встановлює номер формули справа, а друга – зліва. Слід зауважити, що в цьому разі автоматичне генерування посилань на формулу в LaTeX не передбачено.

Вправа. Надрукуйте наступний фрагмент коду.

```
\begin{equation}
\label{Formula1} 2\times2=4.
\end{equation}
$$ 2\times2=4\eqno (2)$$
$$ 2\times2=4 \leqno (*) $$
Перша формула на сторінці \pageref{Formula1} має номер
(\ref{Formula1})
```

Перекреслені символи

Для того щоб отримати в математичній формулі зображення перекресленого символа, потрібно перед командою, що генерує цей символ, поставити команду `\not`.

Приклад.

```
$a \not \in A$
```

$a \not \in A$

Штрихи і похідні

Для друкування символа похідної використовують одинарні лапки

Приклад.

```
$$(fg)' = f'g+fg'$$
```

$$(fg)' = f'g+fg'$$

7.3 Таблиці спеціальних знаків

В даному розділі будуть наведені коди всіх математичних символів, що використовуються в LaTeX.

Грецькі літери

Маленькі грецькі літери

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

Великі грецькі літери

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

Символи бінарних операцій

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

Символи бінарних відношень

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

Стрілки

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

Математичні функції

arccos	`\arccos`	arcsin	`\arcsin`	arctan	`\arctan`
arg	`\arg`	ker	`\ker`	cos	`\cos`
lg	`\lg`	cosh	`\cosh`	ln	`\ln`
cot	`\cot`	log	`\log`	coth	`\coth`
hom	`\hom`	sec	`\sec`	dim	`\dim`
sin	`\sin`	exp	`\exp`	sinh	`\sinh`
tan	`\tan`	csc	`\csc`	tanh	`\tanh`

Оператори з верхніми та нижніми границями

У вихідному файлі верхні та нижні границі позначаються як верхні та нижні індекси відповідно.

lim	`\lim`	lim inf	`\liminf`	lim sup	`\limsup`
max	`\max`	min	`\min`	inf	`\inf`
det	`\det`	gcd	`\gcd`		
$\sum$	`\sum`	$\int$	`\int`	$\oint$	`\oint`
$\bigsqcup$	`\bigsqcup`	$\prod$	`\prod`	$\bigsqcup$	`\bigsqcup`
$\bigcap$	`\bigcap`	$\bigcup$	`\bigcup`	$\biguplus$	`\biguplus`
$\bigoplus$	`\bigoplus`	$\bigotimes$	`\bigotimes`	$\bigodot$	`\bigodot`
$\bigvee$	`\bigvee`	$\bigwedge$	`\bigwedge`		

У виключних формулах межі, як правило друкуються над відповідними знаками операторів. У формулах, що входять в текст, межі друкуються збоку від оператора. Щоб змінити дані установки використовують команди `\limits` і `\nolimits`. Перша команда встановлює режим відображення меж над операторами, друга – справа від оператора

Дужки

$($	`(`	$)$	`)`	$[$	`[`
$]$	`]`	$\{$	`\{`	$\}$	`\}`
$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`	$\rceil$	`\rceil`
$\langle$	`\langle`	$\rangle$	`\rangle`	$\lceil$	`\lceil`
$\backslash$	`\`	$\vert$	`|`	$\Vert$	`\\|`

Значки над символами і виразами

Наступні команди дають змогу розмістити спеціальні значки над односимвольними виразами

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

Команди для розміщення значків над математичними виразами

$\widetilde{abc}$	`\widetilde{abc}`	$\widehat{abc}$	`\widehat{abc}`
$\overleftarrow{abc}$	`\overleftarrow{abc}`	$\overrightarrow{abc}$	`\overrightarrow{abc}`
$\overline{abc}$	`\overline{abc}`	$\underline{abc}$	`\underline{abc}`
$\overbrace{abc}$	`\overbrace{abc}`	$\underbrace{abc}$	`\underbrace{abc}`

Трикрапки

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

Різне

$\aleph$	`\aleph`	∞	`\infty`	$\hbar$	`\hbar`
∂	`\partial`	$\triangle$	`\triangle`	∇	`\nabla`
$\jmath$	`\jmath`	$\imath$	`\imath`	ℓ	`\ell`
$\wp$	`\wp`	$\Re$	`\Re`	$\Im$	`\Im`
$\prime$	`\prime`	$\emptyset$	`\emptyset`	$\angle$	`\angle`
$\forall$	`\forall`	$\exists$	`\exists`	$\neg$	`\neg`
$\surd$	`\surd`	$\top$	`\top`	$\bot$	`\bot`
$\flat$	`\flat`	$\natural$	`\natural`	$\sharp$	`\sharp`
$\clubsuit$	`\clubsuit`	$\diamondsuit$	`\diamondsuit`	$\heartsuit$	`\heartsuit`
$\spadesuit$	`\spadesuit`	$\P$	`\P`	$\S$	`\S`
$\copyright$	`\copyright`	$\pounds$	`\pounds`		

7.4 Одне над іншим

В даному розділі мова піде про те, як розмістити один символ над іншим.

Найпростіші випадки

- `\atop` – використовується, якщо необхідно надрукувати одну частину формули трохи вище рядка, а іншу частину формули трохи нижче.

- `\choose` – відрізняється від попередньої команди тільки тим, що автоматично друкує дужки потрібного розміру навколо формули

Приклад.

```
Біноміальні коефіцієнти
часто позначають
так ${n \choose k}$
```

Біноміальні коефіцієнти
часто позначають
так $\binom{n}{k}$

- `\stackrel` – використовується коли необхідно надрукувати формулу так, щоб її нижня частина залишилась на рівні рядка. Дана команда має два обов'язкових аргументи – перший – те що над рядком, і другий – те що залишиться в рядку.

- \underbrace – використовується для того, щоб намалювати горизонтальну фігурну дужку під виразом. Обов'язковий аргумент цієї команди – це фрагмент формули, під якою необхідно намалювати дужку. Напис під дужкою оформляється як ніжній індекс.

- \overbrace – використовується для того, щоб намалювати горизонтальну фігурну дужку над виразом. Напис над дужкою оформляється як верхній індекс.

Приклад.

```
$F(x)\stackrel{x\to2}{\to}1$
```

$F(x)\stackrel{x\to2}{\to}1$

Приклад.

```
$\underbrace{1+1+\cdots+1}\_n =
 \overbrace{1+1+\cdots+1}^n$
```

$\underbrace{1+1+\cdots+1}_n = \overbrace{1+1+\cdots+1}^n$

Друкування векторів і матриць.

Для друкування векторів чи матриць використовують оточення array. Кожна матриця (вектор) складається з рядків та стовпчиків. Рядки матриці розділяються з допомогою команди \\, а елементи всередині одного рядка, що відносяться до різних стовпчиків розділяються з допомогою символу &.

Після відкриття оточення array (тобто після команди \begin{array}) має йти (в фігурних дужках) так звана преамбула матриці, що описує скільки стовпчиків повинно бути в матриці і як в цих стовпчиках буде вирівнюватися текст. Для цього використовують символи "c", "l" або "r". Матриця буде мати стільки стовпчиків, скільки символів буде вказано в преамбулі матриці (по кожному символу на рядок). Символ "c" означає, що вміст відповідного стовпчика буде вирівнюватися по центру, символи "l" і "r" – по лівому або по правому краю відповідно.

Приклад. В результаті виконання наступного коду

```
$$\left\|
\begin{array}{cc}
a_{11}& a_{12}\\
a_{21}& a_{22}\\
\end{array}
\right\|$$
```

LaTeX надрукує матрицю

$$\left\| \begin{array}{cc} a_{11} & a_{12} \\ a_{21} & a_{22} \end{array} \right\|$$

Оточення `array` можна використовувати не тільки для друкування матриць чи векторів. Дане оточення буде корисне для друкування систем рівнянь, розбиття однієї великої формули на рядки і т.д.

7.5 Шрифти в математичних формулах

Слід зауважити, що перераховані в попередньому розділі команди для роботи із шрифтами не можна використовувати в математичних формулах. Тому, якщо вам необхідно вставити форматований фрагмент тексту в математичну формулу використовуйте наступні команди.

- `\mathrm{}` – прямий шрифт в формулах.
- `\mathbf{}` – напівжирний шрифт в формулах.
- `\mathit{}` – курсивний шрифт в формулах.
- `\mathcal{}` – каліграфічний шрифт в формулах.
- `\mathsf{}` – шрифт типу Serif ("рубленый") в формулах.
- `\mathtt{}` – машинописний шрифт в формулах.

Відмітимо, що всі літери, що входять у формулу інтерпретуються як імена математичних змінних. Тому LaTeX їх друкує шрифтом "математичний курсив". При цьому всі пропуски та кириличні літери ігноруються. Для подолання даної проблеми використовують команду `\mbox{}`.

7.6 Контрольні запитання

- Назвіть основні принципи друкування формул.
- Яка формула називається "виключною"?
- Як надрукувати степінь та індекс у Tex?
- Які команди використовуються для друкування різних типів дужок? Для чого використовуються команди \ left і \ right?
- Яким чином здійснюється автоматична нумерація виключних формул? Як здійснюється посилання на автоматично згенерований номер формули?
- Назвіть 10 основних команд, що використовуються для оформлення математичного тексту.
- Надрукуйте фрагмент коду, що буде відображати матрицю 3 на 3.

Розділ 8

Створення нових команд

В цьому розділі буде описано механізм створення нових команд. Такі команди називають "макровизначеннями" або просто "макросами". Вони дозволяють полегшити набір тексту, що складається з подібних фрагментів, переозначити існуючі команди для швидшого їх набору і т.д.

8.1 Робота з макросами

Створення нових макросів

Макроси використовуються для спрощення введення тексту.

Для створення макросів використовується команда `\newcommand`. Дана команда має два обов'язкових аргументи. Перший – ім'я макроса (має задовольняти ті ж правила іменування, що і звичайна команда). Другий аргумент – це фрагмент коду, який буде виконуватись при використанні макроса.

Приклад. `\newcommand{\eqdef}{\stackrel{\rm def}{=}}`

Таким чином LaTeX, під час компіляції команду `\eqdef`, буде заміщати фрагментом коду `\stackrel{\rm def}{=}`.

Ви можете оголошувати макроси як в преамбулі документу, так і в тексті. При цьому, якщо ви оголосите нову команду в середині групи, то смисл даної команди буде забутий при виході з групи. Команди оголошені в преамбулі документу будуть дійсними для всього документу.

Зауваження. Під час опису нового макроса не можна використовувати в якості імені ім'я вже існуючої команди. Фігурні дужки в другому аргументі команди `\newcommand` мають бути збалансованими. Другий аргумент команди `\newcommand` не повинен містити команд `\newcommand`, а також `\renewcommand`.

Переозначення існуючих команд

Як було сказано в зауваженні, переозначити вже існуючу команду з допомогою `\newcommand` неможливо. Та інколи в цьому виникає необхідність. Для цього використовують команду `\renewcommand`. Вона влаштована аналогічно до команди `\newcommand`. Єдиним виключенням є те, що іменем макроса має бути ім'я вже існуючої команди. Якщо в якості імені макроса в команді `\renewcommand` використати ім'я неіснуючої команди, то, під час компіляції, LaTeX видасть повідомлення про помилку.

Приклад. `\renewcommand{\supseteq}{містить}`

8.2 Команди з аргументами

Для створення нових команд з аргументами використовують команду `\newcommand` з необов'язковим аргументом, що вказує скільки аргументів

буде у нової команди. Даний аргумент міститься між двома обов'язковими і його значення не може перевищувати 9. В тексті, що заміщує макрос, місця на які підставляються аргументи позначаються символами #1 для першого аргументу, #2 для другого, і т.д. Ці символи можуть іти в довільному порядку і довільну кількість разів (в тому числі і жодного разу). При використанні макроса в тексті, після його імені має йти така кількість аргументів, яку було вказано в необов'язковому аргументі при оголошенні макроса. Кожен аргумент має розміщуватись в окремих фігурних дужках.

Приклад.

```
\newcommand{\myint}[1]{\int\limits_{t_0}^{\infty}{#1 dx}}
$$\myint{f(x)}$$
```

Тут ми оголосили нову команду `\myint`, з одним обов'язковим аргументом, котрий буде значенням підінтегрального виразу при використанні команди.

Ви також можете переозначувати вже існуючі команди на нові команди з аргументами. Для цього використовується команда `\renewcommand`. Правила роботи з нею аналогічні до правил роботи з командою `\newcommand`.

8.3 Створення нових оточень

Загальний випадок

Як було сказано вище оголошення нових макросів значно спрощує набір текстів. У тому випадку, коли для досягнення потрібної мети вам потрібно виконати складний набір команд на початку і в кінці деякого фрагменту тексту, LaTeX надає можливість оформити відповідні макроси у вигляді оточень.

Загальний синтаксис оформлення нового оточення наступний

```
\newenvironment{Ім'я_оточення}{Команди_початку}{Команди_кінця}
```

де, `Ім'я_оточення` – ім'я нового оточення.

`Команди_початку` – команди і (або) текст, що виконуються замість команди `\begin{Ім'я_оточення}` під час компіляції.

`Команди_кінця` – команди і (або) текст, що виконуються замість команди `\end{Ім'я_оточення}` під час компіляції.

Приклад.

```
\newenvironment{Bg_Bld_fnt}{\Large \bf \it}{}
```

Таким чином оголошується нове оточення яке називається `Bg_Bld_fnt`. В цьому оточенні є тільки команди які виконуються спочатку оточення, а команди закінчення оточення відсутні. Тут слід нагадати, що кожне оточення є групою. Тому дія всіх команд які оформляють текст оточення впливає виключно на текст оточення і не впливає на текст поза оточенням.

Використання даного оточення аналогічне до використання стандартних оточень LaTeX:

```
\begin{Bg_Bld_fnt}
Даний фрагмент тексту оформлений як оточення Bg_Bld_fnt.
\end{Bg_Bld_fnt}
```

Створення оточень з аргументами

Аналогічно до створення нових команд з аргументами можна створювати нові оточення з аргументами. При цьому правила за якими це здійснюється подібні до правил створення команд з аргументами.

Для створення оточення з аргументами використовується команда `\newenvironment` з одним необов'язковим аргументом. Цей аргумент вказується між першим і другим обов'язковими аргументами у квадратних дужках і вказує кількість аргументів, яка буде в новому оточенні. Максимальна кількість аргументів оточення – 9. Місця, куди будуть вставлятися аргументи позначаються #1 – для першого аргументу, #2 – для другого, і т.д., причому, слід зауважити, що ці значки можна використовувати тільки в другому обов'язковому аргументі оголошення оточення.

Приклад.

```
\newenvironment{Theorems}[2]{\par {\bf Теорема #1.#2} \it}{}
```

Під час оформлення тексту у вигляді оточення з аргументами кожен аргумент вказується в окремих фігурних дужках після команди, що починає оточення, наприклад, для виклику оголошеного вище оточення Theorems з аргументами '1' і "(Піфагор)" потрібно у вихідному файлі надрукувати наступне

```
\begin{Theorems}{1}{(Піфагор)}
Квадрат гіпотенузи дорівнює сумі квадратів катетів
\end{Theorems}
```

З допомогою команди `\newenvironment` неможливо переозначити вже існуюче оточення. Для даної процедури потрібно користуватись командою `\renewenvironment`. Дана команда працює аналогічно до команди `\newenvironment` за виключенням того, що першим аргументом даної команди має бути ім'я існуючого вже оточення.

Створення оточень типу "теорема"

Під час набору математичних текстів у вас буде з'являтись велика кількість різних теорем, лем, зауважень і т.д. Як правило ці елементи математичного тексту оформляють спеціальним чином, відмінним від оформлення загального тексту. Наприклад, слово, яке ідентифікує елемент бажано виділяти

напівжирним шрифтом, а формулювання – друкувати шрифтом відмінним від основного тексту. Вище ми створили оточення `Theorems` яке можна застосовувати для оформлення теорем. Але даний спосіб оформлення фрагментів тексту типу "теорема" має деякі недоліки. Наприклад, нумерацію теорем автору прийдеться здійснювати самостійно.

LaTeX має власний механізм для створення оточень типу "теорема". Для цього використовується команда `\newtheorem`, яка має два обов'язкових аргументи. Перший аргумент – ім'я оточення, другий – заголовок математичного елемента.

Приклад. `\newtheorem{Thm}{Теорема}`

Оформлення тексту у вигляді оточення типу "теорема" здійснюється аналогічно до оформлення тексту за допомогою будь-якого іншого оточення. При цьому слід зауважити, що оточення типу "теорема" має один необов'язковий аргумент. Під час компіляції цей аргумент буде надруковано після заголовку "теореми" в круглих дужках. Цей аргумент, як правило, використовують для того, щоб вказати кому належить "теорема". Наприклад

```
\begin{Thm}[Піфагор]
Квадрат гіпотенузи дорівнює сумі квадратів катетів
\end{Thm}
```

8.4 Контрольні запитання

- Що таке макрос? Наведіть приклад опису нового макросу.
- Що означає термін "переозначення існуючої команди"? Чи відрізняється процес переозначення існуючої команди від створення нової команди?
- Який принцип створення команди з аргументами?
- Створіть новий макрос для друкування значка суми з межами від 1 до нескінченності із змінним індексом сумування.
- Переозначте команду `\tan` для того, щоб вона друкувала позначення функції тангенса прийнятої в нас.
- Створіть нове оточення для друкування тексту з наступними параметрами: похилий шрифт розміром більшим ніж стандартний, розміщений посередині сторінки.
- Створіть нове оточення "Зауваження" котре буде мати аргумент - номер цього зауваження в тексті.
- Створіть нове оточення "Зауваження" типу "Теорема".

Розділ 9

Робота з таблицями

Для створення таблиць використовують одне з наступних оточень

- `tabbing` – використовується для створення таблиць подібних до ти, що можна створити на друкарській машинці
- `tabular` – більш універсальний засіб створення таблиць.
- `array` – використовується для створення таблиць в математичному режимі.

Оточення `array` було розглянуто в попередніх розділах. У даному розділі детально розглянемо способи створення таблиць з допомогою оточень `tabbing` та `tabular`.

9.1 Машинописні таблиці

В "машинописних таблицях" розбиття на стовпчики здійснюється за допомогою першого-керуючого рядка. Конкретно це виглядає так. При наборі першого рядка оточення `tabbing` в місці, де повинен починатися новий стовпчик ставиться команда `\=`. LaTeX це місце (від початку рядка) запам'ятовує. При форматуванні наступних рядків перехід від одного стовпчика до наступного здійснюється з допомогою команди `\>`. Перехід на наступний рядок здійснюється з допомогою команди `\\`

Приклад.

```
\begin{tabbing}
  AAAAAAAAAxxxxxxxxxx \=  \hspace{ 3.3 cm}   \= \kill
  \it Назва           \>  Автор              \>К-ть сторінок\\
  C и C++             \>  Березин Б.И.       \> 288 \\
  C++ для чайников    \>  С.Девис            \> 304 \\
  Математичний аналіз \>  Дороговцев А.Я     \> 300
\end{tabbing}
```

Ширина першого стовпчики буде дорівнювати довжині тексту, що міститься у першому рядку і першому стовпчику, другого – у першому рядку і другому стовпчику, і т.д.

Команда `\kill` використовується для того, щоб не відображати перший рядок таблиці. Наша таблиця буде виглядати наступним чином.

Назва	Автор	К-ть сторінок
C и C++	Березин Б.И.	288
C++ для чайников	С.Девис	304
Математичний аналіз	Дороговцев А.Я	300

Зауваження. Оточення `tabbing` не можна вкладати одне в інше. Кожна комірка таблиці є групою.

9.2 Мальовані таблиці

Елементарні таблиці

При оформленні таблиці з допомогою оточення `tabbing`, автор самостійно повинен встановлювати розміри колонок таблиці та слідкувати за тим, щоб одні колонки не наповзали на інші. Для автоматичного створення таблиць частіше використовують оточення `tabular`. Дане оточення надає більш гнучкі засоби для оформлення таблиць.

Після відкриття оточення `tabular`, тобто після команди `\begin{tabular}`, у фігурних дужках вказується обов'язковий аргумент таблиці, що керує зовнішнім виглядом таблиці. Він називається преамбулою таблиці. В найпростіших випадках преамбула складається з послідовності літер, що описують структуру колонок таблиці (по букві на колонку).

- l – означає колонку, що вирівняна по лівому краю.
- r – означає колонку, що вирівняна по правому краю.
- c – означає колонку з центрованим текстом.

Після преамбули іде текст таблиці. Стовпчики таблиці розділяються символом `&`. Рядки таблиці відокремлюються один від одного символом `\\`.

Пояснимо все вище сказане на наступному прикладі.

Приклад.

```
\begin{tabular}{|c|c|c|}[tcb]
  \hline
  Команда      & Альтернативна     &  Назва гарнітури  \\[5pt]
  \hline
  \verb"\bf"   & \verb"\textbf{}"  &  \textbf{Напівжирний}\\
  \verb"\it"   & \verb"\textit{}"  &  \textit{Курсив}     \\
  \hline
\end{tabular}
```

Преамбула таблиці – це команда `{|c|c|c|}`. Вона вказує на те, що таблиця складається з трьох стовпчиків, причому текст у кожному з стовпчиків буде центрованим. Символи `|` використовується для розділення стовпчиків таблиці вертикальними лініями.

Преамбула таблиці може містити (як у вище наведеному прикладі) необов'язковий параметр, що вказуються у квадратних дужках. Необов'язковий параметр, аналогічно до обов'язкового складається з послідовності літер і описує структуру кожного рядка.

- t – означає вирівнювання по верхньому краю рядка.
- b – означає вирівнювання по нижньому краю рядка.
- c – означає вирівнювання по центру рядка.

Команда `\hline` використовується для розділення рядків таблиці горизонтальними лініями.

Зауважимо, що команда `\\` в першому рядку задана з необов'язковим аргументом. Цей аргумент вказує додаткову величину вертикального відступу між рядками таблиці.

Зауваження. Вся таблиця розглядається LaTeXом, як одна велика літера. Тому оточення `tabular` не починає нового абзацу, і не закінчує абзац.

Таблиці, створені оточенням `tabular` можна вкладати одна в одну.

Об'єднання стовпчиків таблиці

Для того, щоб об'єднати кілька стовпчиків одного рядка в одну графу, використовують команду `\multicolumn`. Ця команда має три обов'язкових аргументи.

- Кількість колонок, що буде об'єднано в одну.
- Преамбула – може складатись з символу c, l або r, що несе той же смисл, що і для преамбули таблиці. В преамбулі також можна використовувати символ |, для відокремлення стовпчика вертикальною лінією.
- Текст, що записується в графу.

Приклад.

```
\begin{tabular}{|c|c|c|}
  \hline
  \multicolumn{3}{|c|}{Команди для зміни шрифту} \\[5pt]
  \hline
  Команда      & Альтернативна     &  Назва гарнітури  \\[5pt]
  \hline
  \verb"\bf"   & \verb"\textbf{}"  &  \textbf{Напівжирний}\\
  \verb"\it"   & \verb"\textit{}"  &  \textit{Курсив}     \\
  \hline
\end{tabular}
```

В результаті виконання попереднього коду, LaTeX намалює наступну таблицю.

Команди для зміни шрифту		
Команда	Альтернативна	Назва гарнітури
\bf	\textbf{}	**Напівжирний**
\it	\textit{}	*Курсив*

У випадках, коли потрібно намалювати горизонтальну лінію не на всю ширину таблиці, замість команди `\hline` використовують команду `\cline`. Вона має обов'язковий аргумент – номер першого і останнього стовпчика, що охоплюються горизонтальною лінією. Їх потрібно розділяти знаком "мінус". Таким чином, у результаті виконання коду

```
\begin{tabular}{|c|c|}
  \hline
  Команда      &  Назва гарнітури  \\[5pt]
  \hline
  \verb"\bf"   &  \textbf{Напівжирний}\\
  \cline{2-2}
  \verb"\it"   &  \textit{Курсив}     \\
  \hline
\end{tabular}
```

LaTeX надрукує наступну таблицю

Команда	Назва гарнітури
\bf	**Напівжирний**
\it	*Курсив*

Абзаци в графах таблиці

Інколи в графі таблиці замість одного рядка потрібно розмістити сформатований абзац. Для цього замість літер c, l або r у преамлулі таблиці на відповідному місці вказують команду `p{...}`, де замість трикрапки вказують ширину стовпчика у одиницях довжини LaTeX.

9.3 Контрольні запитання

- Яким чином можна регулювати ширину колонок у машинописних таблицях?
- Як створити мальовану таблицю з невидимими границями?
- Як створити подвійну вертикальну лінію між колонками?
- Як об'єднати стовпці, колонки та комірки у таблицях?
- Як записати декілька відформатованих абзаців у одну комірку?
- Як створити горизонтальну лінію над двома сусідніми комірками?

Література

[1] Львовский С.М. Набор и верстка в пакете LaTeX. 2-е издание. – М.:Космосинформ, 1995. –373с.

[2] Евграфов М.А., Евграфов Л.М. TEX. Руководство по набору математических текстов. – М.:Наука, 1993. –80с.

[3] Загретдинов Р.В.и др. Издательская система LaTeX. Краткое руководство. – Казань:Казанский государственный университет, 1994. –96с.

[4] Кнут Д. Все про TeX. – Протвино, RDTeX, 1993. –575с.

[5] Котельников И., Чаботаев П. Издательская система LATEX 2_ε. – Новосибирск:Сибирский хронограф, 1998. –496с.

[6] Спивак М. Восхитительный TeX. - М.: Мир, 1993. –285с.

[7] Широков Б.М. PCTeX. Простой способ изящного оформления математических текстов: учебное пособие. – Петрозаводск:Петрозаводский государственный университет, 1994. –80с.

[8] T. Oetiker, H. Partl, I. Hyna, E. Schlegl. Не очень краткое введение в LATEX 2_ε. (LATEX-Kurzbeschreibung) – v.3.7, 1999 – 94 с.

[9] L. Lamport: *LATEX, A Document Preparation System, User's Guide and Reference Manual*, Addison-Wesley Publishing Company (1985), ISBN 0-201-15790-X.