Зміст

Розділ I
Верстка в TEX

1
Знайомство с TEX

В TEXасе TEX уTEXа для TEX, кто любит...
TEXническую верстку.

TEXред

1.1 Вступ до розділу

Мета даного розділу – дати загальне уявлення про основні принципи та довести до рівня практичних навичок техніку набора та верстання текстів загального призначення у видавничій системі TEX.

В результаті вивчення матеріалу розділу студент має знати та вміти:

- створювати макети документу;
- підключати пакети;
- встановлювати стилі сторінок;
- здійснювати роботу з документами, які розділені на декілька окремих файлів;
- верстати текст в TEX;
- створювати додаткові тексти (зміст, список літератури, предметний покажчик і ін.) документу;
- верстати зображення у TEX;
- генерувати на основі TEX-файлу, документ у форматі PDF.

Ключові поняття: преамбула, клас документа, пакет, колонтитули, стиль сторінки, спецсимволи, команда, оточення, гарнитура та розмір шрифта, параграф, таблиці, переліки (списки), структурні елементи документу (розідли, підрозділи, параграфи тощо), додакткові тексти(виноски, зміст, предметний покажчик), зображення.

1.2 Теоретичний матеріал

1.2.1 Створення макету документу

TEX-документ складається з тексту, команд та коментарів.

Кожна команда розпочинається із знаку \.

Будь-який TEX-файл складається з декількох частин:

- преамбула, що визначає параметри;
- власне текст документу.

Преамбула. TEX-файл повинен починатися з команди `\documentclass`, яка задає стиль оформлення документу або клас документу. Наприклад:

```
\documentclass{book}
```

Слово `book` в фігурних дужках вказує, що документ буде мати оформлення як у книжці: усі глави розпочинатимуться з непарної сторінки, текст буде забезпечений колонтитулами деякого визначеного виду і т. п. Окрім стилю `book`, до стандартного комплекту TEXa входять стилі `article` (для оформлення статей), `report` (щось середнє між `article` та `book`) та `letter` (для оформлення ділових листів так, як це прийнято в США). Стандартні стилі можно (а іноді і треба) змінювати, можно створювати й нові стилі, але поки що виходитимемо із стандартних стилей.

Для розширення функціональності TEXa до документу можно підгружати різні пакети (будуть розглянуті далі).

Текст документу. Текст будь-якого TEX-документу повинен починатися та закінчуватися операторськими дужками:

```
\begin{document}
...
\end{document}
```

Слова в тексті відокремлюються один від одного пропусками, при цьому TEXне розрізняє, скільки саме пропусків Ви залишили між словами (щоб вручну управляти пропусками між словами, є спеціальні команди, про які піде мова пізніше). Кінець рядка також сприймається як пропуск. Окремі абзаци мають бути відокремлені один від одного порожніми рядками (знову-таки все одно, скільки саме порожніх рядків між абзацами, важливо, щоб була хоч одна).

```
Слова розділяються пробілами,
а абзаци --
порожніми рядками.
```

Слова розділяються пробілами, а абзаци – порожніми рядками.

Абзацний відступ у початковому тексті залишати не потрібно: він виходить автоматично.

```
 Абзацний відступ у початковому
тексті залишати
 не
потрібно: він виходить
автоматично.
```

1.2.2 Одиниці вимірювання

В TEX можуть використовуватися практично будь-які одиниці виміру: см (cm), мм (mm), пункти (pt), піка (pc), дюйми (in). При цьому в TEX існує ще такі одиниці вимірювання як `ex` (висота 'x') и `em` (ширина букви 'M').

1.2.3 Структура сторінки

За умовчанням в TEX закладена певна структура сторінки, із вже заданими значеннями деяких елементів. Вона зображена на рис. 1.

1 – один дюйм (1in) + \hoffset
2 – один дюйм (1in) + \voffset
3 – \oddsidemargin (\leftmargin) = 2pt
4 – \topmargin = -41pt
5 – \headheight = 18pt
6 – \headsep = 21pt
7 – \textheight = 635pt
8 – \textwidth = 448pt
9 – \marginparsep = 12pt
10 – \marginparwidth = 49pt
\marginparpush = 6pt (not shown)
11 – \footskip = 50pt
\paperheight = 845pt
\paperwidth = 597pt

На рисунку пунктирною лінією зображені поля драйверу (1 і 2) відносно яких вибудовуються усі інші поля. За домовленістю відступи до полів драйвера дорівнюють одному дюйму. Перевизначивши \hoffset та \voffset (по умовчанню вони дорівнюють нулю), можна легко зрушити смугу набору цілком по горизонталі і вертикалі, відповідно.

Тіло тексту характеризується висотою \texthight (7) та шириною \textwidth (8) При багатоколоночному верстанні ширина колонки дорівнює \columnwidth. Змінна \linewidth набуває значення, рівного довжині рядка поточного тексту. \oddsidemargin (3) додається ліворуч у разі одностороннього друку. При двусторонньому друці смуги набору для парних і непарних сторінок відрізняються. В цьому випадку для непарних ліворуч знову ж таки додається \oddsidemargin, а для парних \evensidemargin. Верхній колонтитул розташовується на відстані \topmargin (4) від поля драйвера, має висоту \headheight (5), а тіло тексту відступає від колонтитулу на відстань \headsep (6). \footskip (11) позиціонує базову лінію нижнього колонтитулу відносно останнього рядка тексту. Можна помітити, що в списку параметрів відсутній параметр для установки нижнього поля. Він регулюється за допомогою установки інших параметрів. Т.ч. нижнє поле буде дорівнювати відстані, що залишилася, після установки значень \voffset (2) \topmargin (4) \headheight (5) \headsep (6) і \texthight (7). Поля для заміток мають ширину \marginparwidth (10) і відступають від тіла тексту на відстань \marginparsep (9). Ще одна опція управляє мінімальною відстанню між замітками: \marginparpush.

1.2.4 Розмір паперу

Фізичний розмір паперу описується параметрами \paperwidth і \paperheight. Стандартні базові класи LaTeX(article, book, report і letter) за

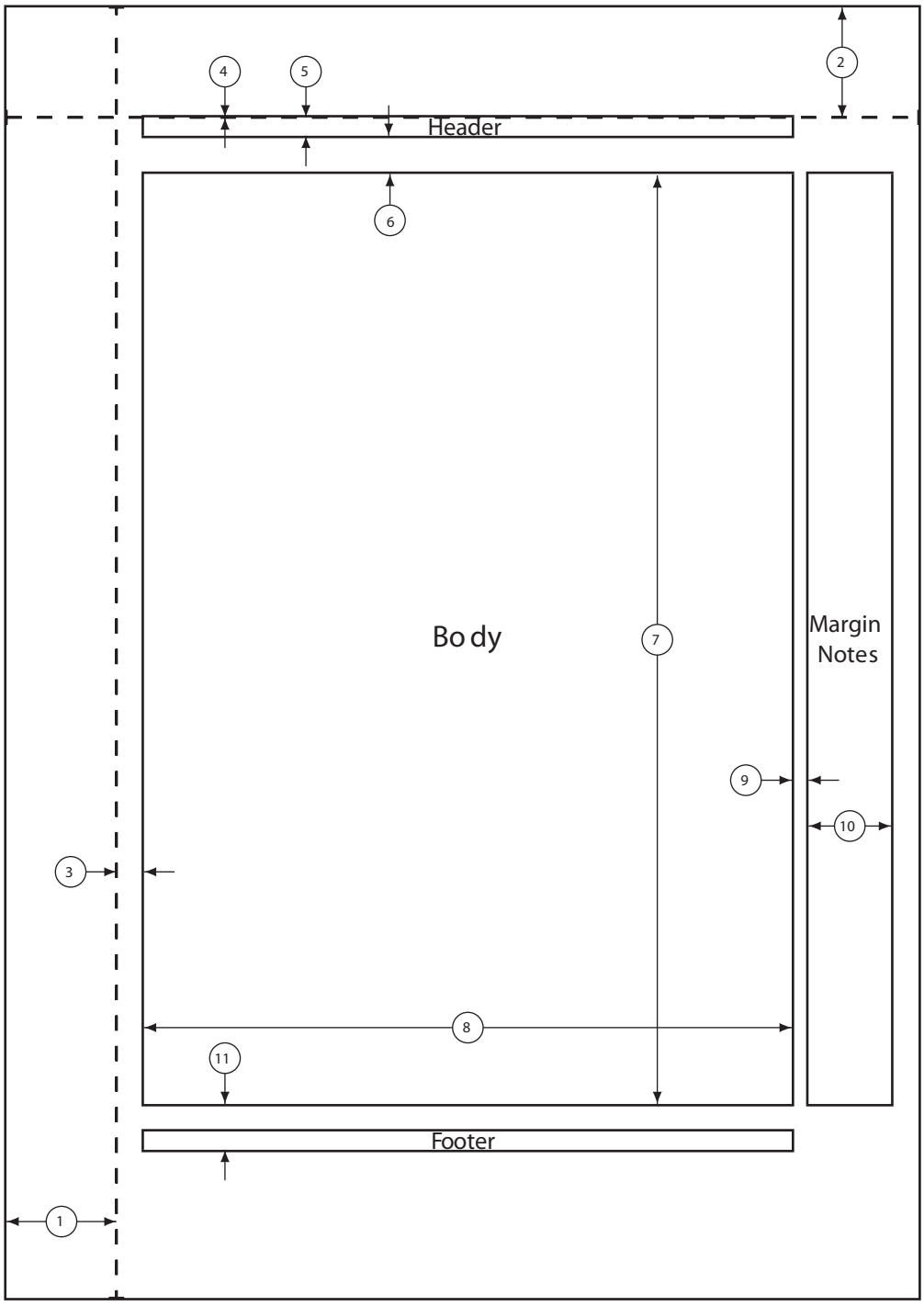


Рис. 1. Макет структури сторінки

умовчанням припускають, що для друку використовується папір формату letter. Встановити правильний формат можна за допомогою передачі параметра a4paper при виборі класу документу :

```
\documentclass[a4paper,12pt,twoside]{book}
```

Для створення невеликих брошур із сторінкою розміру A5(половина розміру A4) використовується опція a5paper.

1.2.5 Інші засоби редагування макету

Існують і інші способи редагування макету. Наприклад, спеціалізований пакет `geometry`.

Пакет `geometry` міняє розміри прямо в процесі завантаження стильового файлу, наприклад, так:

```
\usepackage[margin=1in, paperwidth=5.5in,
paperheight=8.5in]{geometry}
```

або так:

```
\usepackage[margin=3cm,nohead,nofoot]{geometry}
```

або так:

```
\usepackage[top=10mm, bottom=15mm, left=15mm,
 right=12mm, a4paper]{geometry}
```

Детальнішу інформаію можна отримати з офіційної документації до пакету (geometry.pdf).

1.3 Підключення пакетів

Підключення пакетів здійснюється в преамбулі документу. Про те які пакети необхідно підключати, а які немає, слід визначати на основі того, що Ви хочете отримати в кінцевому документі.

Підключення пакету у файл здійснюється командою:

```
\usepackage[<option>]{<package>}.
```

Деякі пакети можуть не мати опцій, а іноді їх вказівка просто не потрібна.

Далі детальніше розглянуті деякі з пакетів, які найчастіше можуть Вам знадобиться в роботі.

1.3.1 Пакет graphicx

Цей пакет дозволяє використати графіку. Формати графічних файлів, які можуть використовуватися при верстанні документу значно різноманітні. Формати підтримувальних графічних файлів : pdf, eps, ps, png, jpg, tiff, mps. Про те, як підключати графічні файли в документ і настроювати їх відображення буде розглянуте пізніше.

1.3.2 Пакет babel

Цей пакет дозволяє настроювати мову документу. Звичайно, цей пакет можна і не вказувати. Але для кожної мови існує свій певний набір правил набор тексту, і установка цього пакету дозволяє перемикати деякі з цих правил.

Підключається пакет аналогічно:

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

В даному випадку цей запис означає, що основна мова документу російська, але в тексті є й англійський текст. Основна мова документу завжди вказується першою.

Існують і інші пакети, що дозволяють налаштувати мову документу. Наприклад, пакет `polyglossia`. Принципово він не відрізняється від `babel`, але все таки є відмінності. Детальнішу інформацію про використання цих пакетів можна отримати з офіційної документації до пакетів (polyglossia.pdf і babel.pdf).

1.3.3 Пакет multicol

Цей пакет дозволяє набирати текст в декілька колонок. Варто мати на увазі, що набір тексту в дві колонки можна задати і в опціях класу, вказавши команду `twocolumn`:

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

але тоді у вас увесь документ буде набраний в 2 колонки, і набрати окремий фрагмент в 1 колонку буде скрутно. Відмінністю цього пакету є те, що кількість колонок може варіюватися по документу. Максимальна допустима кількість колонок – 10.

Підключається пакет аналогічно:

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

Ширина між колонками встановлюється командою:

```
\setlength{\columnsep}{5mm}.
```

1.3.4 Пакет fontspec

Цей пакет дозволяє налаштувати використовувані в документі гарнітури шрифту. Слід мати на увазі, що цей пакет використовується тільки з бібліотеками XeTeX чи XeLaTeX.

Підключається аналогічно:

```
\usepackage{fontspec,xltxtra},
```

але окрім вказівки пакету в преамбулі слід також налаштувати набір використовуваних гарнітур:

```
% встановлює поведінку шрифтів за умовчанням
\defaultfontfeatures{Mapping=tex - text}
% установка основного шрифту документу
\setmainfont{Times New Roman}
% установка гарнітури із зарубками
\setromanfont[ItalicFont={Times New Roman Italic},
 BoldFont={Times New Roman Bold}]{Times New Roman}
% установка гарнітури без зарубок
\setsansfont[ItalicFont={Arial Italic},
 BoldFont={Arial Bold}]{Arial}
% установка моноширинного шрифту
\setmonofont{Courier New}.
```

Перелік опцій і можливостей цього пакету дуже великий. Додаткову инфорамцию можна почитати в офеційній документації до пакету.

1.4 Стилі сторінки

У TeXпередбачене декілька стилів для оформлення сторінок документу. Усі стилі задаються в преамбулі документу за допомогою команди

```
\pagestyle{<name _ style>}.
```

Існують 4 основні стилі:

`empty` – колонтитули і номери сторінки відсутні;

`plain` – колонтитули відсутні, номера сторінок вставлюються по центру внизу сторінки;

`headings` – колонтитули (номери і назви розділів документу), що включають номери сторінок;

`myheadings` – колонтитули, що включають номери сторінок. Відмінність від стилю headings в тому, що текст, що друкується в колонтитулах, не генерується Latex-ом автоматично, а задається користувачем.

У класі документу article сторінки за умовчанням оформляються стилем для оформлення сторінок `plain`.

У класах документу report і book – стилем для оформлення сторінок `headings`.

Стиль для оформлення сторінок `myheadings` підійде для використання, якщо вас влаштовує стандартний стиль оформлення колонтитулів, але не влаштовує текст, який автоматично в ці колонтитули записується.

Вказівка в преамбулі документу стилю поширюється на увесь документ. Якщо для окремої сторінки треба змінити стиль, використовується команда:

```
\thispagestyle{<name_style>}.
```

`\thispagestyle` слід записувати не в преамбулі, а в самому документі – в тексті тієї сторінки, стиль якої вам необхідно змінити. Іншими словами, команда `\thispagestyle` діє лише на ту сторінку, на яку потрапив текст, що оточує цю команду.

Нумерація сторінок, за умовчанням, зазвичай здійснюється арабськими цифрами. За зовнішній вигляд нумерації сторінок відповідає команда:

```
\pagenumbering{<style _ number>}.
```

Для того, щоб змінити арабські цифри, вкажіть в преамбулі документу в обов'язковому аргументі команди `\pagenumbering` один з наступних стилів оформлення нумерації сторінок :

`arabic` – арабські цифри(1, 2, 3,..);
`roman` – римські цифри(i, ii, iii,.);
`Roman` – римські цифри(I, II, III,.);
`alph` – рядкові букви(a, b, c,.);
`Alph` – прописні букви(A, B, C,.).

Команда `\setcounter{<page>}{<number>}` дозволяє нумерувати документ з будь-якої сторінки, яка вам потрібна. Приміром, якщо ви напишете в преамбулі документу `\setcounter{page}{5}`, то нумерація вашого документу розпочнеться не з цифри 1, а з цифри 5.

У TEX існує спеціальний пакет `fancyhdr` для гнучкішого і тоншого налаштування колонтитулів. Його використання більше рекомендується при наборі складних і великих документів.

Для використання цього пакету, необхідно його підключити і настроїти в преамбулі документу. Приклад налаштування колонтитулів документу :

```
\usepackage{fancyhdr}  % підключення пакету
\pagestyle{fancy} % вказівка стилю сторінки
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\mysectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{} % очищаємо поточні установки для колонтитулів
\fancyhead[LE,RO]{\bfseries\thepage} % верхній лівий колонтитул для
```

непарних сторінок і правий – для парних, жирним зображенням вказаний номер сторінки

```
\fancyhead[LO]{\bfseries\rightmark}  % верхній лівий парний
```

колонтитул

```
\fancyhead[RE]{\bfseries\leftmark}  % верхній правий парний
```

колонтитул

`\renewcommand{\headrulewidth}{0.5pt}` % ширина лінійки в верхньому колонтитулі
`\renewcommand{\footrulewidth}{0pt}` % ширина лінійки в нижньому колонтитулі

Складність в налаштуванні колонтитулів в тому, щоб включити туди речі на зразок заголовків розділу або глави. LaTeXдосягає цього в два етапи. У визначеннях колонтитулів можна використати команди `rightmark` і `leftmark`, що представляють заголовки поточної глави і розділу, відповідно. Значення цих двох команд змінюються при обробці команд `\chapter` або `\mysection`.

Для більшої гнучкості команди `\chapter` та їй подібні не перевизначають `\rightmark` і `\leftmark` самі, а викликають ще одну команду, що називається `\chaptermark`, `\mysectionmark` або `\mysubsectionmark`, відповідальну за перевизначення `\rightmark` і `\leftmark`.

Так що, якщо ви хочете змінити вид назви глави у верхньому колонтитулі, ви просто перевизначаєте команду

1.5 Робота з документами розділеними на декілька файлів

У TeX є можливість розбивати складні документи на декілька частин, що зберігаються в різних файлах. Щоб можна було об'єднати їх в одне ціле, в TeXу передбачена команда `\input`. Якщо в тексті написати:

```
\input<file name>,
```

то TeX працюватиме так, як якби замість рядка з командою `\input` стояв текст файлу, ім'я якого Ви вказали.

Зазвичай, коли готують текст великого об'єму, то створюють невеликий файл, в якому між `\begin{document}` і `\end{document}` розміщені рядки з командами `\input`, задаючими включення файлів, в яких і записана основна частина тексту. Наприклад, книгу з чотирьох глав, записаних у файлах `ch1.tex`,. . ., `ch4.tex`, можна організувати у вигляді файлу з декількох рядків (саме його, а не файли з окремими главами, потрібно буде передати для обробки TeXу) :

```
\documentstyle[11pt]{report}
... % other parametrs
\begin{document}
\input ch1.tex
\input ch2.tex
\input ch3.tex
\input ch4.tex
\end{document}
```

Якщо розширення файлу, що є аргументом команди `\input`, не вказане, то TeX за умовчанням вважає, що це розширення має вигляд .tex .

1.6 Верстання тексту

1.6.1 Спецсимволи

Більшість символів в початковому тексті прямо означають те, що буде надруковано. Наступні 10 символів мають особливий статус:

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

Друкарське зображення знаків, що відповідають першим семи з них, можна отримати, якщо в початковому тексті поставити перед відповідним символом без пропуску знак \ (по-англійськи він називається backslash) :

```
Курс тугрика підвищився на 7\% тепер за нього дають \$200.
```

Курс тугрика підвищився на 7% тепер за нього дають $200.

Якщо символ `%`спожитий в тексті не у складі комбінації `\%`,то він є символом коментаря: усі символи, розташовані на рядку після нього TEX ігнорує (і його самого теж).

Коротко про сенс інших спецсимволів. Фігурні дужки обмежують групи в початковому файлі. Знак долара обмежує математичні формули. При наборі математичних же формул використовуються знаки _ і ^ (знак підкреслення і кришка). Знак ~ означає нерозривний пропуск між словами. Зі знаку \ починаються усі TEXовські команди. Знаки # і & використовуються у складніших конструкціях TEXа. Символи < > | у тексті вживати можна в тому сенсі, що повідомлення про помилку це не викличе, але друкуватиметься при цьому щось, зовсім на ці символи не схоже. Справжнє місце для цих символів, так само як і для символів = і – математичні формули, про які мова піде пізніше.

У видавничих системах, грунтованих на TEXе, слід розрізняти дефіс - (по-англійськи hyphen), коротке тире – (по-англійськи en - dash), довге тире – (em - dash) і знак мінуса – (зверніть увагу, що він відрізняється від обох тире).

Щоб отримати на друці дефіс, коротке тире або довге тире, потрібно в початковому тексті набрати один, два або три знаки – відповідно. У російських текстах рекомендується використати довге тире в якості тире як такого, а коротке тире – поєднаннях типу ’я повернуся через 2–3 години’(у початковому тексті це виглядає як через `2--3 години`; зверніть увагу на відсутність пропусків навколо тире). Довге тире, навпаки, має бути оточене пропусками з обох боків(цього вимагає не TEX, а прийняті в Росії друкарські правила). Знак мінуса, на відміну від короткого тире, зустрічається тільки в математичних формулах, і там він, зображається просто знаком -.

Для набору лапок використовуються команди `{\quotedblbase}` для „що відкриває, і `{\textquotedblleft}` – для тієї, що закриває“. Для використання «лапок-ялиночок», треба використати команди `<<текст>>`.

Багатокрапка – це три точки підряд(кожна з яких має стандартну ширину букви). При наборі це не так: для багатокрапки є спеціальна команда `\ldots` (…) або `\dots` (…).

Задати будь-який символ з системи Unicode або ASCII можна командою `{\char"<номер символу>}` для Unicode або `{\char<номер символу>}` для ASCII.

Напрклад: `{\char"A7}` видасть §, а `{\char188}` – ¼.

1.6.2 Зміна шрифтів

Зміна шрифтів (йдеться не про зміну гарнітури шрифту, а про зміну його зображення) – процедура не складна, але необхідна. Усі шрифти в latex можна розділити на три великі сімейства:

із зарубками, прямий – задається командою `\rmfamily` або `\rm`,

без зарубок, рубаний – задається командою `\sffamily` або `\sf`,

`шрифти типу "друкарська машинка"` – задається командою `\ttfamily` або `\tt`.

Шрифти відрізняються насиченістю: **напівжирний шрифт** у тексті задає команда `\bfseries`, `\bf` або `\textbf`, за шрифт середньої насиченості (за умовчанням використовує саме шрифт середньої насиченості) відповідає команда `\mdseries`.

`\upshape` – пряме накреслення

`\itshape`, `\it` або `\textit` – *курсив*,

`\slshape` або `\sl` – *похиле зображення*,

`\scshape` або `\sc` – зображення капітель, коли рядкові букви виглядають як зменшені прописні.

Щоб команда зміни шрифту не поширилася на увесь текст документу, її і текст, який потрібно змінити, слід брати в {}. Наприклад `{\itshape Соціальна мережа викладачів і студентів}`.

Команда `\normalfont` перемикає будь-який шрифт на основний шрифт документу. Для того, щоб перемкнути на основний шрифт частину тексту, можна використати команду з аргументом, що вказує на текст, який необхідно змінити: `\textnormal{текст}`.

1.6.3 Розміри шрифтів в тексті

Стандартні класи документів в TEX, без додаткових пакетів підтримує розміри основного шрифту в 10pt (за умовчанням), 11pt і 12pt.

Змінити розмір шрифту в певній частині документу можна за допомогою наступних команд:

`\tiny`Крихітний
`\scriptsize`Дуже маленький (як індекси)
`\footnotesize`Маленький (як виноски)
`\small`Маленький
`\normalsize`Нормальний
`\large`Великий
`\Large`Ще більший
`\LARGE`Дуже великий
`\huge`Зовсім великий

```
\Huge
```
Страшенно великий

З допомогою спеціального пакету `fix-cm` можна задавати певний розмір шрифту. Для цього треба в преамбулі підключити пакет командою:

```
\usepackage{fix-cm}
```

і використати в тексті документу команду:

```
\fontsize{size}{size}\selectfont
\fontsize{24pt}{16pt}\selectfont Text
```
Text=.

Перший параметр задає розмір шрифту, а другий інтерліньяж набору. Параметр `\selectfont` означає, що використовується вибрана поточна гарнітура. Якщо треба змінити гарнітуру `\selectfont` слід вказувати якийсь з параметрів зміни гарнітури.

Після вказівки цієї команди в тексті вона поширюється на увесь подальший текст. Для того, щоб використати цю команду для окремого фрагмента тексту слід взяти команду у фігурні дужки {}.

Команди зміни розміру шрифтів можуть бути перевизначені користувачем. Перевизначення здійснюється в преамбулі документу, або в зовнішньому стильовому файлі.

Команда перевизначення повинна полягати в оточенні

```
\makeatletter ...  \makeatother
```

і виглядає наступним чином:

```
\makeatletter
\renewcommand\small{\@setfontsize\small{10}{12}}
\renewcommand\normalsize{\@setfontsize\normalsize{12}{14}}
\renewcommand\LARGE{\@setfontsize\LARGE{20}{22}}
\renewcommand\Large{\@setfontsize\Large{18}{20}}
\renewcommand\large{\@setfontsize\large{16}{18}}
\renewcommand\footnotesize{\@setfontsize\footnotesize{9}{11}}
\makeatother
```

У команді присутні 2 параметри, що задаються користувачем: розмір шрифту і інтерліньяж.

1.6.4 Розриви і відступи

Для примусового розриву сторінки в TEXі заборони розриву сторінки існують спеціальні команди:
`\nopagebreak` – заборона розриву сторінки;

\newpage – поточна сторінка завершується і доповнюється знизу порожнім простором, якщо висота сторінки виходить менше, ніж потрібно;
\clearpage – працює так само, як і \newpage, але, якщо до моменту подання цієї команди в тексті залишилися "плаваючі" ілюстрації або таблиці, то вони будуть надруковані перед новою сторінкою;
\cleardoublepage – працює так само, як і \clearpage,але в тих класах документів, в яких передбачено різне оформлення сторінок з парними і непарними номерами, нова сторінка завжди буде непарною (при необхідності створюється додаткова порожня сторінка);
\pagebreak – аналогічна команді \linebreak.

Якщо \nopagebreak поставити після завершення абзацу, то розрив сторінки після цього абзацу буде заборонений. Якщо разом з \nopagebreak використовуються команди для додаткових вертикальних проміжків, то \nopagebreak повинна стояти першою, інакше вона не подіє.

\nopagebreak може приймати необов'язковий аргумент [] – ціле число від 0 до 4. Чим більше числа в аргументі, тим менш обов'язковий розрив сторінки. Іншими словами: \nopagebreak[4] – означає повна заборона розриву сторінки; \nopagebreak[0] – означає, що в цьому місці сторінку в принципі можна розірвати; 1, 2, 3 – міра небажаності розриву документу збільшується із зростанням аргументу.

Якщо поставити підряд дві команди \newpage або \clearpage, чиста сторінка на друці не вийде. Для її створення необхідно між двома командами для розриву сторінки дати команду \mbox{}:

```
\newpage\mbox{}/newpage.
```

Якщо дати \pagebreak без аргументів, то сторінка в цьому місці буде розірвана. При цьому буде зроблена спроба вирівняти сторінку по висоті з іншими сторінками за рахунок розтягування вертикальних інтервалів (абзацних). Якщо задати \pagebreak необов'язковий аргумент – ціле число від 0 до 4, то число виражатиме міру бажаності розриву сторінки в цьому місці: 0 – дозвіл розірвати сторінку; 4 – розрив обов'язковий; 1, 2, 3 – міра бажаності розриву сторінки збільшується із зростанням аргументу.

Будь-яку з цих команд можна давати не лише між абзацами, але і усередині них. Розрив сторінки станеться (чи буде заборонений) після того рядка, в який потрапляє текст, що сусідить з цією командою.

Невеликі горизонтальні проміжки – пропуски вручну – задаються в системі наступними командами:

\, – "тонкий" пропуск;
\: – "середній" пропуск;
\; – "товстий" пропуск;
\! – "негативний" пропуск, зменшує відстань між символами настільки, наскільки \, збільшує;
\quad – 1 em (у поточному шрифті відстань, яка займає прописна буква М);
\qquad – 2em;
\enskip – 0.5em.

Якщо необхідно задати проміжок з вказівкою конкретної довжини, можна скористатися командою \hspace{довжина проміжку} у TEXовских одиницях виміру довжини. Команда \hspace*{довжина проміжку} використовується у

разі, якщо цей проміжок повинен зберігатися і на початку, і у кінці рядка. Якщо команда \hspace потрапить в кінець рядка, то вона буде опущена.

Команда \hspace (\hspace*) може мати рухливу довжину проміжку подібно до команди:

```
\hspace{x plus y minus z} ,
```

де x – основна величина проміжку в одиницях довжини Latex; y – вказує міра розтяжності в одиницях довжини Latex; z – вказує міра стисливості в одиницях довжини Latex.

У аргументі ви можете вказати один з показників (y або z), і тоді заданий вертикальний проміжок або не зможете розтягуватися, або не зможе стискатися.

Для пропуску з необмеженою розтяжністю \hspace{\fill} (команда \fill задає відрізок нульового розміру, здатний нескінченно розтягуватися) існує коротка форма \hfill.

\hrulefill – задає нескінченно розтяжний проміжок подібно \hfill, заповнюючи його суцільною лінією:

```
A\hrulefill B
```

на друці виглядатиме як

A_____B,

причому суцільна лінія розтягуватиметься на всю довжину заданої сторінки так, щоб в її межі ще помістити букви A і B.

\dotfill – задає нескінченно розтяжний проміжок аналогічно \hrulefill, але заповнюваний точками.

Щоб задати певний проміжок, на який повинна поширюватися команда \hrulefill і їй подібні, помістите їх в аргумент команди \hbox:

```
\hbox to 5 cm{A\hrulefill B}
```

A_____B

Створить на друці "бокс" шириною 5 см, по краях якого будуть розташовані букви A і B.

Вертикальні проміжки між абзацами задаються за допомогою команд \smallskip, \medskip і \bigskip.

Конкретна величина цих проміжків залежить від класу документу.

Ці команди слід ставити відразу після порожнього рядка або команди \par, що завершує абзац.

Вертикальний проміжок між абзацами явного розміру можна задати за допомогою команди \vspace{n}, де n – величина проміжку в одиницях виміру. На початку сторінки слід використати команду \vspace*{n} (\vspace на початку рядка не працює).

Значення аргументів команд \vspace (\vspace*) і \\ можуть приймати як позитивні, так і негативні значення. Якщо поставити команди \vspace, \smallskip, \medskip і \bigskip між рядками усередині абзацу, то і вертикальний проміжок буде усередині абзацу.

Команда \vspace (\vspace*) може мати змінну довжину проміжку. В цьому випадку її аргумент виглядатиме таким чином:

```
\vspace{x plus y minus z},
```

де x – основна величина відступу в одиницях довжини Latex; y – вказує міра розтяжності в одиницях довжини Latex; z – вказує міра стисливості в одиницях довжини Latex.

Також ви можете вказати один з показників – або y, або z, і тоді заданий вертикальний проміжок або не зможе розтягуватися, або стискатися.

Додатковий проміжок між абзацами можна задати за допомогою команди \parskip, за умовчанням рівною нулю в усіх класах окрім letter. Наприклад, щоб встановити додатковий інтервал між абзацами в 5мм, слід записати:

```
\setlenght{\parskip}{5mm},
```

де \setlenght – команда, що змінює значення командної довжини.

Значення команди \parskip можна задавати в явному виді, як в попередньому прикладі, або використовуючи компоненти деформації plus і minus, як в прикладі з \vspace.

В аргументі команди \vspace (\vspace*) можна поставити команду \fill, яка задає проміжок нулевого розміру з нескінченною розтяжністю. Приміром, це може знадобитися, якщо ви захочете розташувати текст рівно по центру сторінки. Тоді вам необходио буде записати:

```
\clearpage\vspace*{\fill}
 \begin{center}
 Соціальна мережа студентів і викладачів
 \end{center}
 \vspace*{\fill}\clearpage
```

Якщо в аргументі \vspace (\vspace*) перед \fill поставити ціле число або десятковий дріб, розтяжність \fill множитиметься на це число. Іншими словами, якщо записати:

```
\clearpage\vspace*{0.5\fill}
 \begin{center}
 Заголовок
 \end{center}
 \vspace*{\fill}\clearpage
```

те перед словом "заголовок" буде залишений рівно в два рази менше місця, ніж після нього, оскільки 0.5\fill розтягується в два рази менше, ніж \fill.

1.6.5 Управління абзацами

Згадана вище команда \linebreak відповідає за розрив рядка (аналог команди Shift Enter в MS Word). При цьому текст розтягується, що залишився в рядку, на усю

ширину рядка. Для того, щоб розірвати рядок, аналогічно команді Enter в MS Word, необхідно використати команду `\par` або `\\`.

Декілька записаних підряд команд `\par`, також як і декілька підряд порожніх рядків будуть рівносильні одній команді `\par` і одному порожньому рядку. Команда `\\` може мати необов'язковий аргумент [n], значення якого, вказане в певних одиницях довжини, дорівнюватиме розміру додаткового вертикального проміжку.

```
Соціальна мережа студентів і викладачів.\\[15mm]Блог о
 системі верстання в Latex.
```

На виході відобразить результат так:

Соціальна мережа викладачів і студентів.

Блог про систему верстання в Latex.

Команда `\parindent 12pt` – встановлює значення абзацного відступу.

`\noindent` – використовується для одиничного пригнічення абзацного відступу.

`\looseness=n` – використовується для зменшення або збільшення абзацу на певну кількість рядків. Приміром `\looseness=1` зробить абзац на один рядок довше за рахунок розтягування слів, а `\looseness=-1` – коротше за рахунок стискування.

За умовчанням `\looseness=0`, нове значення повинне привласнюватися для кожного абзацу наново.

`\parfillskip=0pt` – ця команда з нульовим значенням може допомогти у тому випадку, якщо довжина останнього рядка абзацу не набагато коротша за ширину сторінки. `\parfillskip` постарається розтягнути останній рядок, збільшуючи проміжки між словами в рядках абзацу.

`\clubpenalty` – відповідає за небажаність розриву сторінки після першого рядка абзацу.

`\widowpenalty` – відповідає за небажаність розриву сторінки перед останнім рядком абзацу.

По суті ці команди регулюють висячі рядки в документі. Чим вище значення цих параметрів, тим рідше відбуваються розриви. За умовчанням значення `\clubpenalty` і `\widowpenalty` = 150. Максимальне значення цих команд (повна заборона розривів) = 10000.

Оточення `{center}` – вирівнювання абзаців по центру.

Оточення `{flushleft}` – вирівнювання абзаців по лівому краю.

Оточення `{flushright}` – вирівнювання абзаців по правому краю.

1.6.6 Таблиці

Існує декілька оточень для набору таблиць в TEX. Основні з них – `tabular` – для набору таблиць з текстом і `array` – для набору таблиць, що містять формули.

Графи в таблиці розділяються за допомогою знаку амперсенд &. Кількість граф вказується в обов'язковому аргументі `\begin{tabular}` чи `\begin{array}`. Вертикальних розмежувачів – `|` – також вказуються в обов'язковому аргументі. Горизонтальні розмежувачі задаються командою `\hline` (якщо рису треба підвести під усіма рядками таблиці) або командою `\cline{x - y}`, де x - y – рядки, під якими треба підвести рису. Розділяються рядки знаком абзацу `\\`.

Приміром, якщо ми хочемо створити таблицю з трьох колонок і одного рядка, текст в яких буде відцентрований, слідує записати:

```
\begin{tabular}{|c|c|c|}
 \hline
1&2&3\\
 \hline
 \end{tabular}
```

На виході отримаємо:

1	2	3

Текст в комірках (параметри обов'язкового аргументу у оточення) можна розподіляти таким чином:

`l` – означає колонку, вирівняну по лівому краю;
`c` – означає колонку, вирівняну по центру;
`r` – означає колонку, вирівняну по правому краю;
`p{<ширина колонки>}` – означає колонку, текст в якій верстається як абзац;
`m{<ширина колонки>}` – абзац буде вирівняний по середині своєї висоти;
`b{<ширина колонки>}` – абзац буде вирівняний по нижньому рядку.

Опції `m{.}` і `b{.}` доступні при підключенні пакету `array`.

Також оточення `tabular` і `array` може мати необов'язкові аргументи, які розумно використати в зовсім дрібних таблицях, розташованих усередині тексту:

`[t]` – вирівнює таблицю по верхньому рядку
`[b]` – вирівнює таблицю по нижньому рядку.

Щоб об'єднати декілька горизонтальних комірок в одну, існує команда:

```
\multicolumn{x}{y}{z},
```

де x – кількість комірок, які слід об'єднати, y – спосіб розташування тексту в комірках (подібно до обов'язкового аргументу у оточення tabular/array), z – безпосередньо текст, який міститиметься у великій комірці.

1.6.7 Переліки (Списки)

Для друку переліків використовується оточення `itemize`, `enumerate` і `description`.

Оточення `itemize` призначене для простих переліків.

Кожен елемент переліку вводиться командою `\item` без параметра.

На друці кожен елемент переліку забезпечується чорним кружечком (маркером).

Переліки можуть бути вкладеними один в одного. Максимальна глибина вкладеності дорівнює 4. Відступи і символи перед елементами вибираються автоматично. На другому рівні елементи переліку відзначаються напівжирними короткими тире, на третьому – зірочками, на четвертому – точками.

При спробі вкласти п'ять такого оточення TEXвидасть повідомлення про помилку.

Приклад:

```
\begin{itemize}
\item Перший;
\item Другий:
\begin{itemize}
\item Це два-один;
\item Два-два;
\end{itemize}
\item Третій;
\item Четвертий.
\end{itemize}
```

- Перший;
- Другий:
  - Це два-один;
  - Два-два;
- Третій;
- Четвертий.

Усередині оточення `itemize` до першої команди `\item` не повинне йти ніякого тексту або ж команд, що генерують текст.

Оточення `enumerate` призначене для нумерованих переліків.

У таких переліках кожен елемент також вводиться командою `\item` без параметра, але на друк він буде відмічений не маркером, а номером. Також можуть бути вкладеними.

Приклад:

```
\begin{enumerate}
\item Перший;
\item Другий:
\begin{enumerate}
\item Це два-один;
\item Два-два;
\end{enumerate}
\item Третій;
\item Четвертий.
\end{enumerate}
```

1. Перший;
2. Другий:
   (a) Це два-один;
   (b) Два-два;
3. Третій;
4. Четвертий.

Оточення `description` призначене для переліків, в яких кожен пункт має заголовок (наприклад, словникових статей або інших описів).

У цих переліках кожен елемент забезпечений заголовком. Тому елементи переліку вводяться командою `\item` з необов'язковим аргументом (ув'язненим в квадратні дужки), що є цим заголовком.

Приклад:

```
Відомі цитати керівників :
\begin{description}
\item[Ю.В.~Андропов:]
Політика -- це не гра
у покер, де, продувшись,
можна відігратися.
\item[К.У.~Черненко:]
Щоб краще жити, потрібно
краще працювати.
\end{description}
```

Відомі цитати керівників :

Ю.В. Андропов: Політика – це не гра у покер, де, продувшись, можна відігратися.

К.У. Черненко: Щоб краще жити, потрібно краще працювати.

1.6.8 Розділи документу

Працюючи з TEXом, розумно робити заголовки і нумерацію розділів не вручну, а за допомогою спеціальних команд.

Для оформлення розділів існують такі команди:

```
\part \chapter \mysection \mysubsection
\mysubsectiono \paragraph \subparagraph
```

У цьому переліку кожна подальша команда означає дрібніший підрозділ, ніж попередня. Слід мати на увазі, що команда `\chapter` ('глава') в стилі `article` не визначена (завдяки цьому обставині статтю легко переробити в главу книги), інші команди визначені в усіх трьох стандартних стилях. Стандартні стилі забезпечують нумерацію розділів, при якій дрібніший розділ 'підлеглий' більшому: коли, наприклад, починається нова `\mysection`, нумерація розділів `\mysubsection` і більше дрібних починається наново. Виключенням з цього правила є команда `\part` ('частина') : якщо частина 2 кінчається главою 5, то перша з глав частини 3 матиме номер 6, а не 1. При модифікації стандартних стилів можна міняти як принцип нумерації розділів, так і вид цих номерів на друці (наприклад, якщо ми захочемо, щоб розділи позначалися послідовними буквами алфавіту) Все те, що ми говорили про необов'язковий аргумент і варіант з зірочкою у команди `\mysection`, застосовно і до команд, перерахованих в цьому розділі. 'Занадто дрібні' розділи, згідно стандартним стилям, не відбиваються ні в змісті, ні в колонтитулах і не нумеруються, але, якщо Ви споживете задаючі їх команди з необов'язковим аргументом або із зірочкою, помилкою це не буде.

Існує можливість змінювати стандартні заголовки. Змінити можна вирівнювання, зображення, відступи і нумерацію.

Зміна здійснюється командою `\renewcommand`, яка має бути поміщена в `\makeatletter ... \makeatother`:

```
\makeatletter
\renewcommand{\mysection}{\@startsection{section}{1}
{0pt}{0pt}{12.5pt}{\normalfont\LARGE\centering}}

\renewcommand{\mysubsection}{\@startsection{subsection}{1}
{0pt}{25pt}{12.5pt}{\normalfont\Large\centering}}
```

```
\renewcommand{\mysubsectiono}{\@startsection{subsubsection}
{1}{0pt}{25pt}{12.5pt}{\normalfont\large\centering}}
\makeatother
```

У цьому визначенні у команди `\@startsection` вказані шість аргументів, в яких закодовані різні параметри оформлення розділу. Розберемо послідовно, що ці аргументи означають.

Перший з аргументів (у нашому випадку `section`) – це "внутрішнє ім'я", під яким LaTeX дізнаватиметься визначуваний тип розділів документу.

Другий аргумент (у нашому випадку `1`) – це "рівень вкладеності" розділу, про який йшла мова вище.

Третій аргумент задає відступ заголовка від лівого поля (у нашому випадку цей відступ дорівнює нулю). Четвертий аргумент (у нашому випадку це opt або 25 pt) – задає величину вертикального відступу, що залишається перед заголовком. П'ятий аргумент (у нашому випадку 12.5pt) задає величину вертикального відступу після заголовка розділу.

Нарешті, шостий аргумент команди `\@startsection` задає стиль оформлення заголовка. Точніше кажучи, в цьому аргументі записаний текст і/або команди, які будуть вставлені перед заголовком розділу.

Перевизначення заголовків здійснюється в преамбулі документу.

Нумерація заголовком змінюється тією ж командою, але має дещо інші параметри:

```
\makeatletter
\renewcommand{\thesection}{\arabic{section}.}
\renewcommand{\thesubsection}{\thesection.\alph{subsection})}
\makeatother
```

У цьому прикладі для заголовка `\mysection` вказана нумерація арабськими цифрами з точкою, а для заголовка `\mysubsection` – нумерація виглядатиме, як номер попереднього заголовка з точкою і буква алфавіту з дужкою, наприклад, ось так: 1.a) або так 3.c) . Якщо потрібно, щоб нумерація заголовка `\mysubsection` здійснювалася, наприклад, просто римськими цифрами, то слід написати таку команду:

```
\makeatletter
\renewcommand{\thesubsection}{\Roman{subsection})}
\makeatother
```

Принципи нумерації ті ж, що і для нумерації сторінок (див. п. 3 на стор. 7).

Якщо потрібно, щоб в заголовку перед цифрою стояв наприклад символ параграфа (чи будь-який інший символ або слово), то слід записати так:

```
\makeatletter
\renewcommand{\thesubsection}{{\char"A7} \Roman{subsection})}
\makeatother
```

Якщо треба створити новий лічильник, наприклад, букви російського алфавіту, то це слід робити так:

```
\makeatletter
\newcommand*{\ralph}[1]{\@ralph{\@nameuse{c@#1}}}
\newcommand*{\@ralph}[1]{\ifcase #1\or а\or б\or в\or г\or д\or
 е\or ж\or з\or и\or к\or л\or м\or н\or о\or п\or р\or
 с\or т\or у\or ф\or х\or ц\or ч\or ш\or щ\or
 э\or ю\or я\else\@ctrerr \fi}
\makeatother
```

Аналогічним чином, можна створити лічильник з оригінальних маркерів, грецьких букв і ін., і потім їх можна використати для нумерації переліків, сторінок, заголовків і іншого.

1.7 Додаткові тексти в TEX

До додаткових текстів, що вимагають особливого набору в TEX, відносять: зміст, предметний покажчик, список ілюстрацій, список таблиць, виноски і деякі інші.

1.7.1 Виноски

Щоб зробити виноску до якогось місця в тексті, досить використати команду `\footnote` з одним обов'язковим аргументом – текстом виноски. У стандартних стилях TEXa виноски[1] нумеруються підряд упродовж усієї глави або навіть (у стилі article) всього документу. Приклад коду набору виноски:

```
виноски\footnote{На зразок цієї} нумеруються ...
```

Нумерацію кожної виноски[33] можна проставляти вручну, за допомогою необов'язкового аргументу в квадратних дужках:

```
виноски\footnote[33]{На зразок такої} нумеруються ...
```

Нумерацію виносокВ також можна змінювати*:

```
\renewcommand{\thefootnote}{\Alph{footnote}}
```

У TEXa існує спеціальний пакет для гнучкої роботи з виносками – `footmisc`. Він дозволяє розміщувати їх у вигляді окремого параграфа в нижній частині сторінки або на полях документу, позначати виноски різними символами, коректно використати виноски в заголовках, використати дві виноски в одному місці і

1 На зразок цієї
1 На зразок такої
В Наприклад так
* Чи так

багато що інше. Детальнішу інформацію про нього можна отримати з офіційної документації.

1.7.2 Зміст

Складання змісту здійснюється командою `\tableofcontents`, яку можна розмістити у будь-якому місці документу (слід розміщувати там, де ви власне хочете помістити в документі зміст). Ця команда збирає у файл з розширенням TOC інформацію про наявні команди секціонування, а потім виводить вміст цього файлу в якості змісту. Для правильного формування списку знадобиться два проходи.

У автоматично збираний зміст не потрапляють заголовки помічені зірочкою, наприклад `\mysection*{Заголовок}`. Для того, щоб додати такий заголовок в зміст необхідно відразу після заголовка вказати таку команду:

```
\mysection*{Заголовок}
\addcontentsline{toc}{section}{Заголовок}
```

Ця команда має 3 обов'язкові аргументи. Перший – означає до якого саме списку додається запис, другий – якого рівня цей заголовок третій – власне текст заголовка.

Зміст буде озаглавлений словом, визначуваним командою `\contentsname` (за умовчанням набуває значення залежно від мови документу), яку з потреби можна перевизначити:

```
\renewcommand{\contentsname}{Зміст}
```

Якщо необхідно змінити зображення заголовка змісту, слід використати таку команду:

```
\makeatletter
  \renewcommand{\tableofcontents}{\mysection*{\contentsname}
 \@starttoc{toc}}
\makeatother
```

чи, наприклад, таку:

```
\renewcommand{\contentsname}{\normalfont\LARGE Зміст}
```

Відмінність полягає в тому, що в першому випадку зміняться налаштування і відступів, а в другому випадку – тільки зображення.

Тут означає, що заголовок змісту прийме стиль заголовка `\mysection`, при цьому не поміщатиметься в зміст.

1.7.3 Предметний покажчик

На відміну від змісту, створення предметного покажчика автоматизоване не повністю.

Спершу треба підключити в преамбулі спеціальний пакет `makeidx`. Потім вказати власне команду для створення предметного покажчика: `\makeindex`. І тільки потім аналогічно як і для змісту, розмістити в тексті документу команду `\printindex`, яка виведе на друк предметний покажчик. Розміщувати останню команду слід там, де ви хочете в тексті отримати предметний покажчик. Наприклад запис, може мати такий вигляд (предметний покажчик розміщуватиметься у кінці документу):

```
\documentclass{book}
...
\usepackage{makeidx}
\makeindex
...
\begin{document}
...
\printindex
\end{document}
```

Предметний покажчик буде озаглавлений словом, визначуваним командою `\indexname`. Її перевизначення аналогічне, і в нім же можна задати і зображення заголовка предметного покажчика:

```
\renewcommand{\indexname}{\normalfont\LARGE Покажчик}
```

При складанні документу на окремій сторінці з'явиться напис із заголовком предметного покажчика і в два стовпці підуть переліки термінів і сторінок, на яких вони знаходяться.

У тексті, де трапляється слово, яке має бути в предметному покажчику, слід вставити команду: `\index{слово}`. При цьому до слова можуть бути застосовані параметри зображення, які відіб'ються на цьому терміні в предметному покажчику.

1.8 Робота із зображеннями

Для використання зображень в TEX-документі, необхідно підключити пакет `graphicx`. Цей пакет дозволяє імпортувати як векторну, так і растрову графіку. У пакету є необов'язковий, але бажаний аргумент, який може набувати значень: dvips, pdftex або ін. Від цього аргументу залежать формати підтримуваних файлів: для dvips це файли з розширенням – eps, ps, а для pdftex – png, pdf, jpg, mps, tif.

Щоб вставити зображення, використовуємо команду

```
\includegraphics[keyval-list]{file},
```

де {file} – ім'я файлу, а [keyval - list] – список ключів, які задаються у вигляді key = value через кому.

Розширення файлу з малюнком в команді \includegraphics можна не вказувати, оскільки TEX сам знає, які типи файлів він може обробити, а які немає. Коли розширення файлу в команді не вказане, драйвер послідовно додає до імені файлу усі відомі йому розширення, поки не знайде перший відповідний файл.

Можливі ключі:

```
\includegraphics{image}
% без ключів, вставить зображення
% в оригінальному розмірі
```

```
\includegraphics[width=4cm]{image}
% задається ширина зображення
```

```
\includegraphics[height=10mm]{image}
% задається висота зображення
```

```
\includegraphics[scale=0.75]{image}
% масштабування малюнка в n разів
```

```
\includegraphics[angle=30]{01}
% повертає малюнок на n градусів
% проти годинникової стрілки
```

Пакет `graphicx` має і інші можливості налаштування відображення графіки. Детальнішу інформацію можна отримати з офіційної документації до пакету.

Для розміщення рисунку з підписом слід використати таке оточення:

```
\begin{figure}[h]
\includegraphics[keyval - list]{image}
\caption{Підпис рисунку}
\end{figure}
```

Це оточення має один аргумент, який може набувати таких значень : [h] – ”хотілося б картинку тут”, але рішення приймається виходячи із заповненої сторінки; [h!] – ”дуже хочу картинку тут”; [H] – ”ХОЧУ картинку тут і точка”; зі збільшенням букви p([pH]) ви примушуєте помістити TEX картинку окремо на сторінку.

Для розміщення малюнків в оборку, слід використати таке оточення:

```
\begin{floatingfigure}[розміщення]{ширина}
...
\end{floatingfigure}
```

чи таке:

```
\begin{wrapfigure}[число рядків в оборці]
{розміщення}{ширина}
...
\end{wrapfigure}
```

Детальнішу інформацію можна отримати в довідковій літературі.

Завдання

Метою даної роботи є відтворення електронного видання на основі аналізу .

Використовуючи TEX-файл даної лабораторної роботи, підготувати звіт, зверстаний у системі TEX, який повинен містити:

1. Титульний аркуш, оформлений засобами TEX.
2. Титул, оформлений відповідно до вимог щодо оформлення звітів.
3. Згідно з варіантом зверстаний текст документу.

Розділ II
Верстка спеціальних видань у TEX

2
Верстка математичних видань

2.1 Складання формул

2.1.1 Загальні положення

Основним призначенням системи TEX є підготовка до друку математичних текстів будь-якої складності. Ознайомитися з детальним списком цих символів та командами для їх складання можна, наприклад, у [5] чи [6]. У даній темі буде розглянуто найбільш поширені символи, знання яких є достатнім для підготовки значної кількості навчальних та наукових видань.

У документах TEX'а розрізняють два види математичних формул:

- внутрішньотекстові;
- виключні (виділені в окремий рядок).

Внутрішньотекстові формули оточуються в знаки $, або `\(` та `\)`, або `\begin{math}` та `\end{math}`. До формулами, крім цілих формул, так само відносяться окремі символи, у тому числі, грецькі, верхні і нижні індекси, спецзнаки. Пробіли всередині вихідного тексту, що задані у такі формули, ігноруються: TEX розставляє їх автоматично. Порожні рядки всередині тексту, що задає формулу, не дозволяються. Знаки пунктуації, що мають стояти після внутріншьотекстової формули, повинні стояти після знаку (команди), що закриває формулу.

```
Зворотня теорема Піфагора:\\
для будь-якої трійки
позитивних чисел $a$, $b$ і
$c$ такій, що $a^2+b^2=c^2$
існує прямокутній трикутник
з катетами $a$, $b$ і
гіпотенузою $c$.
```

Зворотня теорема Піфагора:
для будь-якої трійки позитивних чисел a, b і c такій, що $a^2 + b^2 = c^2$ існує прямокутній трикутник з катетами a, b і гіпотенузою c.

Виключні формули оточуються парами знаків долара `$$` з обох сторін, або `\[` та `\]`, або `\begin{displaymath}` та `\end{displaymath}`.

```
Зворотня теорема Піфагора:\\
для будь-якої трійки
позитивних чисел $a$, $b$ і
$c$ такій, що $$a^2+b^2=
c^$2$ існує прямокутній
трикутник з катетами $a$,
$b$ і гіпотенузою $c$.
```

Зворотня теорема Піфагора:
для будь-якої трійки позитивних чисел a, b і c такій, що

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

існує прямокутній трикутник з катетами a, b і гіпотенузою c.

Для набору формул, що повинні мати нумерацію використовується оточення `\begin{equation}` та `\end{equation}`. Якщо надалі у тексті необхідно реалізувати посилання на певний математичний вираз, використовуються команди `\label{eq:<ім'я мітки>}`, яка створює мітку на формулу, та `\ref{eq:<ім'я мітки>}`, яка власне робить вставку посилання у тексті.

```
Теорема Піфагора:\\
\begin{equation}
    \label{eq:Pifagor}
    a^2+b^2=c^2
\end{equation}
З (\ref{eq:Pifagor}) видно,
що...
```

Теорема Піфагора:

$$a^2 + b^2 = c^2 \tag{2.1}$$

З (2.1) видно, що...

Команди, які використовуються при наборі формул діють лише на наступний символ. Тому, для того щоб команда діяла на декілька символів необхідно, створити з них групу за допомогою фігурних скобок {...}.

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

$$a^x + y \neq a^{x+y} \tag{2.2}$$

2.1.2 Складові математичної формули

Математичні символи

Літери грецької абетки

α	\alpha	λ	\lambda	σ	\sigma	φ	\varphi
β	\beta	μ	\mu	τ	\tau	ϖ	\varpi
χ	\chi	μ	\mu	θ	\theta	ϱ	\varrho
δ	\delta	o	o	υ	\upsilon	ς	\varsigma
ϵ	\epsilon	ω	\omega	ξ	\xi	ϑ	\vartheta
η	\eta	ω	\omega	ζ	\zeta	Δ	\Delta
γ	\gamma	π	\pi	F	\digamma	Γ	\Gamma
ι	\iota	ψ	\psi	ε	\varepsilon	Λ	\Lambda
κ	\kappa	ρ	\rho	$\varkappa$	\varkappa	Ω	\Omega

Φ	\Phi	Σ	\Sigma	Θ	\Theta
Π	\Pi			Υ	\Upsilon
Ψ	\Psi			Ξ	\Xi

Символи операцій

$\amalg$	\amalg	$\otimes$	\otimes	$\bigcirc$	\bigcirc
$\cap$	\cap	$\circ$	\circ	$\triangleleft$	\triangleleft
$\cup$	\cup	$\bullet$	\bullet	$\Diamond$	\Diamond
$\uplus$	\uplus	$\diamond$	\diamond	$\bigtriangleup$	\bigtriangleup
$\sqcap$	\sqcap	$\lhd$	\lhd	$\bigtriangledown$	\bigtriangledown
$\sqcup$	\sqcup	$\rhd$	\rhd	$\Box$	\Box
$\vee$	\vee	$\unlhd$	\unlhd	$\triangleright$	\triangleright
$\wedge$	\wedge	$\unrhd$	\unrhd	$\setminus$	\setminus
$\oplus$	\oplus	$\oslash$	\oslash	$\wr$	\wr
$\ominus$	\ominus	$\odot$	\odot		

Символи відносин

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

Інші символи відносин, які не наведені у переліку, такі як «=», «<» та «>» в TEX набираються з клавіатури.

Три крапки

У науковій літературі часто використовуються символ «три крапки», який може бути:

```
горизонтальним
$$
(a_1,a_2,\ldots)
```

$$
\text{горизонтальним}
(a_1,a_2,\ldots)
$$

```
центрованим
$$
(a_1+a_2+\cdots+a_n)
```

$$
\text{центрованим}
(a_1+a_2+\cdots+a_n)
$$

```
вертикальним
$$A=\left(\begin{array}{c}
a_1\\ a_2\\ \vdots\\
a_n \end{array}\right)$$
```

вертикальним

$$A=\left(\begin{array}{c} a_1\\ a_2\\ \vdots\\ a_n \end{array}\right)$$

```
діагональним
$$A=\left(\begin{array}{ccc}
a_{11}&\cdots&a_{1n}\\
\vdots&\ddots&\vdots\\
a_{1n}&\cdots&a_{nn}
\end{array}\right)$$
```

діагональним

$$A=\left(\begin{array}{ccc} a_{11}&\cdots&a_{1n}\\ \vdots&\ddots&\vdots\\ a_{1n}&\cdots&a_{nn} \end{array}\right)$$

Стрілки

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

Інші символи

∞	\infty	$\S$	\S	$\Box$	\Box
$\aleph$	\aleph	£	\pounds	$\diamondsuit$	\diamondsuit
$\jmath$	\jmath	$\triangle$	\triangle	$\bowtie$	\Join
$\Re$	\Re	$\hbar$	\hbar	¶	\P
$\prime$	\prime	ℓ	\ell	$\angle$	\angle
$\surd$	\surd	$\Im$	\Im	$\imath$	\imath
$\bot$	\bot	$\emptyset$	\emptyset	$\wp$	\wp
$\forall$	\forall	∂	\partial	$\mho$	\mho
$\flat$	\flat	$\vdash$	\vdash	∇	\nabla
\	\backslash	$\exists$	\exists	$\top$	\top
$\clubsuit$	\clubsuit	$\natural$	\natural	$\dashv$	\dashv

¬ \neg	♡ \heartsuit	■ \blacksquare
♯ \sharp		© \copyright
◇ \Diamond		

Верхні, нижні індекси. Надрядкові та підрядкові знаки. Акценти

Набір верхніх та нижніх індексів здійснюється за допомогою команд _ та ^ відповідно. Наприклад,

`$a_{1}$`	⇒	a_1
`$x^{2}$`	⇒	x^2
`$e^{-\alpha t}$`	⇒	$e^{-\alpha t}$
`$a^{3}_{ij}$`	⇒	a^3_{ij}
`$e^{x^2} \neq {e^x}^2$`	⇒	$e^{x^2} \neq e^{x2}$

Бувають випадки, коли верхній або нижній індекс слід розміщувати не один під одним, а на різних відстанях від виразу, до якого вони належать. У цьому випадку їх слід оформити як відповідний індекс до порожнього виразу.

Для відтворення надрядкових та підрядкових знаків використовуються наступні команди:

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

Слід зазначити, що команди `\overbrace` та `\underbrace` дають можливість робити написи над або під фігурними дужками. Наприклад, конструкція

$$\underbrace{1+3+5+7+\ldots+(2n-1)}_{n \text{ доданків}}=n^2$$

побудована таким чином:

```
$$\underbrace{1+3+5+7+\ldots+(2n-1)}_
                              {\mbox{$n$ доданків}}=n^2$$
```

У стильовому пакеті `amsmath` існують команди `\xleftarrow` і `\xrightarrow`, спеціально призначені для нанесення написів над і під стрілками. В обов'язковому аргументі цих команд ставиться напис над стрілкою, в необов'язковому – під стрілкою (необов'язковий аргумент ставиться перед обов'язковим). Якщо напис довгий, то розмір стрілки автоматично збільшується. Наприклад, для відтворення конструкції

$$A \xleftarrow[z]{f} B \xrightarrow{f+g-h} C$$

використовується команда:

```
$$A\xleftarrow[z]{f}B \xrightarrow{f+g-h}C$$
```

Система TEX дає змогу застосовувати широкий набір акцентів та спеціальних символів. Нижче наведено приклади застосування в математичному режимі акцентів стосовно до букви «a». Зрозуміло, що на її місці можуть бути інші букви.

$\acute{a}$	`\acute{a}`	$\Acute{\Acute{A}}$	`\Acute{\Acute{A}}`
$\breve{a}$	`\breve{a}`	$\Breve{\Breve{A}}$	`\Breve{\Breve{A}}`
$\ddot{a}$	`\ddot{a}`	$\Ddot{\Ddot{A}}$	`\Ddot{\Ddot{A}}`
$\grave{a}$	`\grave{a}`	$\Grave{\Grave{A}}$	`\Grave{\Grave{A}}`
$\tilde{a}$	`\tilde{a}`	$\Tilde{\Tilde{A}}$	`\Tilde{\Tilde{A}}`
$\bar{a}$	`\bar{a}`	$\Bar{\Bar{A}}$	`\Bar{\Bar{A}}`
$\check{a}$	`\check{a}`	$\Check{\Check{A}}$	`\Check{\Check{A}}`
$\dot{a}$	`\dot{a}`	$\Dot{\Dot{A}}$	`\Dot{\Dot{A}}`
$\hat{a}$	`\hat{a}`	$\Hat{\Hat{A}}$	`\Hat{\Hat{A}}`
$\vec{d}$	`\vec{a}`	$\Vec{\Vec{A}}$	`\Vec{\Vec{A}}`

```
$p(A)+p(\bar{A})=1$
```
$\Rightarrow \qquad p(A)+p(\bar{A})=1$

Дроби та корені

Прості дроби в внутрішньотекстових формулах позначають навскісною рискою /. Такі дроби набираються безпосередньо:

```
Нерівність $x+1/x \geqslant 2$
справедлива для всіх $x>0$
```
Нерівність $x+1/x \geqslant 2$ справедлива для всіх $x>0$

Слід відзначити, що знаки ⩽ (`\leqslant`) та ⩾ (`\geqslant`) набрані в українському стилі та відрізняються від американських аналогів ≤ (`\leq`) і ≥ (`\geq`) відповідно. Для використання команд `\geqslant` та `\leqslant`, в преамбулі слід прописати підключення пакету `amssymb`.

Дроби, у яких чисельник розміщений над знаменником, набирають за допомогою команди `\frac`. Ця команда має два обов'язкових аргументи: перший – чисельник, другий – знаменник.

```
$$\frac{a+b}{c+d}$$
```
$$\Rightarrow \qquad \frac{a+b}{c+d}$$

Якщо чисельник та знаменник дроби складається з однієї лише букви або цифри, то їх необов'язково брати у фігурні дужки. Наприклад, конструкція

$$\frac12+\frac13+\frac14+\ldots+\frac1n$$

будується таким чином:

```
$$\frac12+\frac13+\frac14+\ldots+\frac1n$$
```

Квадратний корінь набирається за допомогою команди `\sqrt` з обов'язковим аргументом – підкореневим виразом. Корінь довільної степені набирається за допомогою тієї ж команди з необов'язковим аргументом – показником кореня.

```
$$ c = \sqrt{a^2+b^2} $$
```
$\Rightarrow$ $c = \sqrt{a^2+b^2}$

```
$$ x = \sqrt[n]{a+b} $$
```
$\Rightarrow$ $x = \sqrt[n]{a+b}$

Дужки

У навчальній чи науковій літературі використовують велику кількість різних обмежувальних дужок: круглі, квадратні, фігурні, кутові й інші дужки та розділювачі. Як правило, такі дужки вживають парами. Круглі «`()`», квадратні «`[]`» дужки та знак модуля «`|`» набирають безпосередньо на клавіатурі. Для набору відкритої та закритої фігурних дужок використовують прості команди `\{` і `\}`, а для кутових дужок «⟨» та «⟩» використовуються команди `\langle` та `\rangle`.

Розглянуті символи (дужки) належать до класу розділювачів, які мають властивість змінювати розміри залежно від виразу, який вони обмежують. Наприклад, якщо деякий фрагмент займає більше місця по вертикалі (за рахунок дробів, степенів і тому подібного), то обмежувальні дужки повинні мати відповідно більші розміри.

Для цього в TEX передбачено спеціальний механізм автоматичного вибору розміру дужок, для включення якого перед лівим обмежувальним символом (наприклад, відкривальною круглою дужкою) ставлять команду `\left`, а перед правим (закривальною круглою дужкою) – `\right`. Тоді розміри обмежувальних символів (дужок) будуть автоматично підлаштовуватися під розмір формули. Традиційними прикладами можуть бути різні типи матриць.

```
$$
A= \left(\begin{array}{cccc}
1&0&0&0\\
0&1&0&0\\
0&0&1&0\\
0&0&0&1\end{array}\right)
$$
```
$$A= \left(\begin{array}{cccc} 1&0&0&0\\ 0&1&0&0\\ 0&0&1&0\\ 0&0&0&1\end{array}\right)$$

Команди `\left` і `\right` є парними, тобто для кожної відкривальної команди `\left` повинна бути відповідна закривальна команда `\right`.

Бувають ситуації, коли потрібен лише один з обмежувальних символів (лівий або правий). У цьому разі замість непотрібного обмежувального символа слід поставити крапку «.», на місці якої нічого не надрукується, проте другий обмежувальний символ буде необхідних розмірів. Наприклад,

```
$$
\left\{\begin{array}{c}
x_1+x_2 = 25\\
x_1+25 = 0
\end{array}\right.
$$
```
$$\left\{\begin{array}{c} x_1+x_2 = 25\\ x_1+25 = 0 \end{array}\right.$$

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

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

Інколи, для покрашення читабельності формули, яка складається з декількох обмежувальних символів, що стоять поруч, доцільно зробити так, щоб зовнішні знаки були дещо більші за внутрішні. Для цього в TEX використовуються команди `\bigl`, `\Bigl`, `\biggl`, `\Biggl` для лівих обмежувальних символів та `\bigr`, `\Bigr`, `\biggr`, `\Biggr` для правих (команди перераховані в порядку зростання створюваних ними символів).

```
$$
\Biggl[[x,y],[x,z],[y,z]\Biggr]
$$
```

$$\Biggl[[x,y],[x,z],[y,z]\Biggr]$$

Границі

Границя являє собою складний знаковий комплекс, для набору якого використовуються кілька команд. Для набору границі використовується команда `\lim`.

```
$$f'(x)=\lim_{\Delta x \to 0}
\frac{f(x+\Delta x) - f(x)}
{\Delta x}$$
```

$$f'(x)=\lim_{\Delta x \to 0}\frac{f(x+\Delta x) - f(x)}{\Delta x}$$

```
$$
\lim_{n \to \infty} x_n = a
$$
```

$$\lim_{n \to \infty} x_n = a$$

Сума. Добуток. Інтеграли

Сума задається командою `\sum`, а добуток – `\prod`. Межі набираються як верхні або нижні індекси й розміщуються відповідно над і під знаком суми чи добутку.

```
$$\sum_{i=0}^{n} x_i$$
```

$$\sum_{i=0}^{n} x_i$$

```
$$\prod_{j=0}^{n} j=n!$$
```

$$\prod_{j=0}^{n} j=n!$$

Слід звернути увагу на те, що внутрішньотекстові формули відрізняються від виключних. Правило розташування меж таке: якщо формула внутрішньотекстова, то межі розташовуються збоку як індекси, якщо ж формула виключна, то – зверху та знизу.

```
$\sum_{i=0}^{n} x_i $
```
$\sum_{i=0}^{n} x_i$

```
$\prod_{j=0}^{n} j=n!$
```
$\prod_{j=0}^{n} j=n!$

Оскільки TEX використовує це правило за замовченням, інколи виникає потреба змінити їхнє розташування. Для цього призначені команди `\limits` та `\nolimits`. Команда `\limits` ставить межі над і під знаком оператора, `\nolimits` – збоку як індекси.

Знак інтеграла набирається командою `\int`. Межі інтегрування за замовченням друкуються збоку від знака як індекси, незважаючи на те, виділена формула чи ні. Управляти розташуванням меж можна за допомогою команд `\limits` та `\nolimits`.

```
$$
\int_0^{\pi}\sin x\,dx = 2
$$
```
$$\int_0^{\pi}\sin x\,dx = 2$$

```
$$
\int\limits_0^{\pi}\sin x\,dx=2
$$
```
$$\int\limits_0^{\pi}\sin x\,dx=2$$

Матриці

Набір матриць може здійснюватися двома способами:

- за допомогою оточення `array`;
- за рахунок використання стильового пакету `amsmath` та його оточень.

Набір матриці за допомогою оточення array

```
$$A=\left(\begin{array}{cccc}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{array}\right)$$
```
$$A=\left(\begin{array}{cccc}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{array}\right)$$

Матриця складається з рядків і стовпчиків, елементами яких можуть бути довільні символи чи вирази. У вхідному тексті рядки матриці розділяються командою `\\` (останній рядок закінчувати командою `\\` не потрібно), а елементи стовпчиків усередині одного рядка відділяються один від одного символами `&`.

Число стовпчиків і спосіб їх форматування задаються в обов'язковому аргументі оточення `array`. У наведеному прикладі цей спосіб задається як cccc. Це означає, що в матриці чотири стовпчики (по букві на стовпчик) і що дані кожного стовпчика розміщені по його центру (оскільки кожна з літер – літера «c»). Окрім букви «c», в аргументі оточення можуть стояти символи «l» або «r», що означає вирівнювання по лівому або правому краю відповідно. Дужки або інші роздільники навколо матриці легко поставити за допомогою конструкції `\left(` та `\right)`.

Оточення стильового пакету amsmath для роботи з матрицями

Найбільш зручні оточення для створення матриць міститься у стильовому пакеті `amsmath`. До таких оточень відносяться: `matrix`, `bmatrix`, `Bmatrix`, `pmatrix`, `vmatrix`, `Vmatrix`. Розглянемо деякі з них.

```
$$A= \begin{matrix}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{matrix}$$
```

$$A= \begin{matrix}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{matrix}$$

```
$$A= \begin{pmatrix}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{pmatrix}$$
```

$$A= \begin{pmatrix}
1 & 0 & 0 & 0\\
0 &1 & 0 & 0\\
0 & 0 & 1 & 0\\
0 & 0 & 0 &1\end{pmatrix}$$

2.1.3 Складні конструкції

Довгі формули

Доволі часто виникає ситуація, коли формула настільки велика, що не вміщується в рядку. В цьому випадку її можна розбити на кілька частин за допомогою оточення `multiline` стильового пакету `amsmath`. Для позначення місця розриву рядка використовується команда `\\`. Перший рядок розщепленої формули друкується виключеним вліво, останній – виключеним вправо, решта рядків центруються. Формула, побудована цим оточенням, нумерується, але нумерацію можна відмінити, якщо скористатися варіантом цієї команди із зірочкою: `multiline*`. Розглянемо відповідні приклади:

$$p_n(x) = (x - x_0)(x - x_1)\cdots (x - x_{j-1})(x - x_{j+1})\cdots (x - x_{n-1})(x - x_n) \quad (2.3)$$

$$p(A_1 + A_2 + \cdots + A_n) = \sum_{i=l}^{n} p(A_i) - \sum_{i_1<i_2} p(A_{i_1}A_{i_2}) + \sum_{i_1<i_2<i_3} p(A_{i_1}A_{i_2}A_{i_3}) - \cdots + (-1)^{n-1} p(A_1A_2\ldots A_n)$$

Ці формули набрано за допомогою оточення `multline` так, як показано нижче.

```
\begin{multline}
p_n(x)=(x - x_0)(x - x_1)\cdots \\
(x - x_{j-1})(x - x_{j+1})\cdots \\
```

```
(x - x_{n-1})(x- x_n)
\end{multline}

\begin{multline*}
p(A_1+A_2 + \dots + A_n)= \sum_{i=1}^{n}p(A_i) -
\sum_{i_1<i_2}p(A_{i_1} A_{i_2}) +\\
+\sum_{i_1<i_2<i_3}p(A_{i_1} A_{i_2} A_{i_3})-\dots+
(-1)^{n-1} p(A_1 A_2 \dots A_n)
\end{multline*}
```

Блоки рівнянь

В ситуаціях коли виключені формули складають групу, доцільно користуватися оточенням `gather`, яке дозволяє уникнути значних вертикальних інтервалі між сусідніми формулами (такі інтервали будуть, якщо скористатися оточенням `$$ ... $$` або `\begin{equation} ... \end{equation}`). Відділення одної формули від іншої здійснюється командою `\\`.

$$\frac{\sqrt{2}}{4}=d \tag{2.4}$$

$$\sqrt{1+(d/h)^2}=3$$

$$\frac{h}{\sqrt{h^2+d^2}}=\frac{1}{\sqrt{1+(d/h)^2}}=\frac{1}{3} \tag{2.5}$$

Слід звернути увагу, що в оточенні `gather` для середньої формули було скасовано нумерація за допомогою команди `\notag`.

```
\begin{gather}
\frac{\sqrt{2}}{4}=d\\
\sqrt{1+(d/h)^2}=3 \notag \\
\frac{h}{\sqrt{h^2+d^2}}=\frac{1}{\sqrt{1+(d/h)^2}}=
\frac{1}{3}
\end{gather}
```

Оточення `gather` завжди розміщює формули по центру. Якщо потрібне інше розташування, то доцільно скористатися оточенням `align`. Це оточення призначено для вертикального вирівнювання кількох рівнянь чи формул. Цим оточенням також зручно скористатися для горизонтального розміщення кількох стовпчиків формул, кожний з яких вирівняний по вертикалі згідно із заданою позицією.

Для оточення `align` правила нумерації й розділення формул такі само, як і для `gather`. Символ `&` задає позицію вертикального вирівнювання. Наступний фрагмент демонструє приклад застосування цього оточення. У вихідному тексті символ `&` поставлений у кожному рівнянні біля знака рівності, тому вирівнювання відбулося відносно цього знака.

```
\begin{align*}
x + y & = 4 \\
x^2 + y^2 & = 4\\
a&=b
\end{align*}
```

$$\begin{aligned} x + y &= 4 \\ x^2 + y^2 &= 4 \\ a &= b \end{aligned}$$

Оточення `align` дає можливість розміщувати по горизонталі кілька стовпчиків. Правило застосування символа `&` тут дещо складніше. Непарні $(1, 3, \dots)$ знаки `&` задають точки вертикального вирівнювання відповідного стовпчика, парні $(2, 4, \dots)$ – правлять лише розділювачами для стовпчиків. Прикладом такого верстання формул є таблиця множення.

$$\begin{aligned} 1\times1&=1 & 2\times1&=2 & 3\times1&=3 & 4\times1&=4 \\ 1\times2&=2 & 2\times2&=4 & 3\times2&=6 & 4\times2&=8 \\ 1\times3&=3 & 2\times3&=6 & 3\times3&=9 & 4\times3&=12 \\ & & & & & & 4\times4&=16 \end{aligned}$$

```
\begin{align*}
1 \times 1 & = 1 & 2 \times 1 & = 2 &
3 \times 1 & = 3 & 4 \times 1 & = 4 \\
1 \times 2 & = 2 & 2 \times 2 & = 4 &
3 \times 2 & = 6 & 4 \times 2 & = 8\\
1 \times 3 & = 3 & 2 \times 3 & = 6 &
3 \times 3 & = 9 & 4 \times 3 & = 12\\
& & &  &  &  & 4 \times 4 & = 16\\
\end{align*}
```

Якщо при верстанні документу потрібно поставити ліворуч від системи відповідну фігурну дужку та нумерувати систему цілком, а окремі рівняння, доцільно використовувати оточення `aligned` стильового пакету `amsmath`.

```
\begin{equation}\left\{
\begin{aligned}
x + y & =4 \\
x^2 + y^2 & = 4
\end{aligned}\right.
\end{equation}
```

$$\left\{\begin{aligned} x + y &= 4 \\ x^2 + y^2 &= 4 \end{aligned}\right. \tag{2.6}$$

Умовні конструкції

Для побудови умовних конструкцій в TEX використоується оточення `cases`, яке застосовується тільки внутрі оточення `\equation`.

```
\begin{equation*}
\sigma(x)=
\begin{cases}
-1,&\text{якщо $x < 0$},\\
0,&\text{якщо $x = 0$},\\
1,&\text{якщо $x > 0$}
\end{cases}
\end{equation*}
```

$$\sigma(x)=\begin{cases}-1, & \text{якщо } x<0,\\ 0, & \text{якщо } x=0,\\ 1, & \text{якщо } x>0\end{cases}$$

Слід звернути увагу на команду `\text`, яка дозволяє робити текстові вставки у формулу.

2.2 Генерація математичної графіки

2.2.1 Оточення picture

Основні команди

Оточення `picture` створюється одною з двох команд:

$$\texttt{\textbackslash begin\{picture\}}\ (x,y)\ldots\ \texttt{\textbackslash end\{picture\}}$$

або

$$\texttt{\textbackslash begin\{picture\}}(x,y)(x_0,y_0)\ldots\texttt{\textbackslash end\{picture\}}.$$

Числа x, y, x_0, y_0 вимірюються в розмірності `\unitlength`, яку можна змінювати в будь-який момент (але не всередині оточення `picture`) за допомогою команди:

$$\texttt{\textbackslash setlength\{\textbackslash unitlength\}\{1.2cm\}}$$

Значення `\unitlength` за замовчуванням становить `1pt`. Перша пара (x, y) резервує для картинки прямокутний простір всередині документа. Необов'язкова друга пара (x_0, y_0) – присвоює довільні координати нижньому лівому кутку зарезервованого прямокутника.

До основних команд малювання відносяться:

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

або

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

Для малювання кривих Бєз'є використовується команда

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

До основних об'єктів, які можна малювати за допомогою команди `\put` відносяться: відрізки, вектори, окружності, овали, текст та формули.

Відрізки та вектори

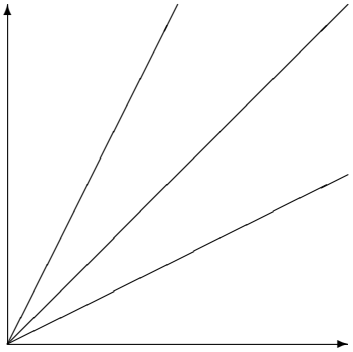
Відрізки задаються за допомогою команди \line

\line(x_1, x_2){довжина відрізку}.

Команда яка має два параметра нахил і довжину відрізка. Нахил відрізка задається парою цілих чисел, розташованих в круглих дужках через кому безпосередньо після \line. Відношення цих чисел має дорівнювати «кутовому коефіцієнту» відрізка (тангенсу кута нахилу до горизонту). Якщо ці числа (1, 0), то відрізок горизонтальний, якщо (0, 1), то відрізок вертикальний. Кожне число має бути цілим, не повинно перевищувати 6 по абсолютній величині, і, крім того, ці два числа не повинні мати спільних дільників, крім 1.

Розмір відрізка задається в фігурних дужках після круглих дужок, в яких заданий нахил. Цей розмір задає довжина проекції відрізка на горизонтальну вісь (крім випадків, коли відрізок вертикальний – тоді задається його довжина по вертикалі).

```
\setlength{\unitlength}{45mm}
\begin{picture}(1,1)
\put(0,0){\vector(0,1){1}}
\put(0,0){\line(1,2){.5}}
\put(0,0){\line(1,1){1}}
\put(0,0){\line(2,1){1}}
\put(0,0){\vector(1,0){1}}
\end{picture}
```



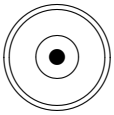
Стрілки задаються за допомогою команди \vector, синтаксис якої абсолютно такий же, як у команди \line. Единою відмінністю від команди \line є те, що нахилів стрілок ще більш обмежений, ніж у відрізків: цілі числа, які задають нахил, не повинні перевищувати 4 по абсолютній величині (і як і раніше не повинні мати спільних дільників).

Окружність, круг і овал

Окружність задається командою \circle, а круг (суцільний чорний кружок) – її варіантом «із зірочкою» \circle*. Єдиний аргумент цих команд – діаметр. Він задається в одиницях, рівних значенням параметра \unitlength. Точкою відліку кола або кола є центр.

При цьому слід відзначити, що оточення picture дозволяє малювати окружності діаметром не більше 14мм, при цьому, в цим межах доступні ні всі значення діаметру. Якщо окружність або круг з діаметром, зазначеним в якості аргументу команди \circle або \circle*, в TEX'овських шрифтах немає, то буде надрукована окружність (круг), діаметр якої найбільш близький до вказанного.

```
\setlength{\unitlength} {1mm}
\begin{picture}(50,40)
\put(25,25){\circle{18}}
\put(25,25){\circle{12}}
\put(25,25){\circle{6}}
\put(25,25){\circle*{2}}
\end{picture}
```



Для малювання овалу (хоча якщо точніше, то це прямокутник з закругленими кутами) використовується команда `\oval`. Аргументами цієї команди є ширина і висота овалу, які задаються в круглих скобках через кому.

```
\setlength{\unitlength} {1mm}
\begin{picture}(50,20)
\put(25,10){\oval(25,10)}
\end{picture}
```

Якщо необхідно намалювати «неповний» овал, то можна задати необов'язковий аргумент (в квадратних дужках після обов'язкового). Для завдання половини овалу цей аргумент повинен бути однією з таких літер:

`t`	верхня половина;
`b`	нижня половина;
`r`	права половина;
`l`	ліва половина.

Крім половини, TEX може намалювати як $1/4$ так й $3/4$ овалу. Для малювання $1/4$ необов'язковому аргументу необхідно задати комбінацію літер, яка вказує, яку саме чверть необхідно відображати. Для малювання $3/4$ овалу слід використовувати дві команди.

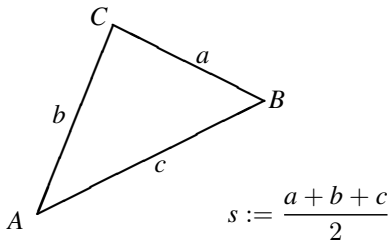
```
\setlength{\unitlength} {1mm}
\begin{picture}(50,20)
\put(25,10){\oval(25,10)[tr]}
\put(25,10){\oval(25,10)[b]}
\end{picture}
```

Текст і формули

Текст і формули розміщуються в оточенні `picture` без додаткових команд.

```
\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(0.6,0.3){$A$}
\put(4.05,1.9){$B$}
\put(1.7,3.0){$C$}
\put(3.1,2.5){$a$}
\put(1.2,1.7){$b$}
\put(2.55,1.05){$c$}
\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}
```

$$F = \sqrt{s(s-a)(s-b)(s-c)}$$



Як показав огляд основних команд оточення `picture`, він є достатньо простим, але має певну кількість обмежень.

2.2.2 Пакет mfpic

Пакет `mfpic` призначений для побудови рисунків, графіків, діаграм для науково-технічних документів. Підключення стильового пакету `mfpic` здійснюється наступним чином:

```
\usepackage[metapost]{mfpic}
\mfpicunit = 1mm
\opengraphsfile{figures}

\begin{document}
    ...
    \closegraphsfile
\end{document}
```

З наведеного прикладу видно, що підключення здійснюється звичайним способом: командою `\usepackage[metapost]{mfpic}`. Окремо слід звернути увагу на необов'язковий аргумент даної команди, який залежно від нашого вибору може принимати значення `Metafont` або `Metapost`. Різниця полягає форматі вихідної ілюстрації. Для випадку команд metapost будуються векторні рисунки, а для команд metafont будуються – растрові.

Параметр `\mfpicunit` стильового пакету `mfpic` працює аналогічно `\unitlength`, тобто задає розмірності елементів рисунка. За умовчанням його значення дорівнює одному пунктові.

Команда `\opengraphsfile` і `\closegraphsfile` відповідно відкривають і закривають спеціальний файл, призначений для збереження побудованих рисунків. Ім'я файлу вказується в якості обов'язкового параметру команди `\opengraphsfile`.

Для побудови власне рисунка використовується оточення `mfpic`, яке відкриває й закриває середовище, в якому можна використовувати макрокоманди, призначені для опису рисунка, а також встановлює локальну систему координат. Воно має наступний формат:

```
\begin{mfpic}[x_scale][y_scale]{x_min}{x_max}{y_min}{y_max}
    ...
\end{mfpic}
```

Параметри x_{scale} і y_{scale} задають масштабні одиниці для координатних осей x та y відповідно. Для випадку рівних масштабів можна записати лише один параметр. У будь-якому разі їхні значення повинні бути кратні довжині параметра `\mfpicunit`. Відсутність необов'язкових параметрів x_{scale} і y_{scale} відповідає випадкові, коли масштабні одиниці рівні й кожна з них дорівнює `\mfpicunit`.

Параметри x_{min} і x_{max} встановлюють нижню й верхню межі для рисунка по координатній осі x; аналогічно y_{min} і y_{max} встановлюють межі по осі y. Ці межі виражаються в локальних одиницях, заданих параметрами x_{scale} і y_{scale}. Для визначення фактичної ширини зображення можна від верхньої межі x_{max} відняти нижню x_{min} і різницю помножити на x_{scale}.

Осі координат. Координатна сітка. Текстові вставки

Координатні осі можна будувати командами

```
\axis[hlen]{one-axis}
\axis[hlen]{axis-list}
```

де опція `hlen` задає довжину стрілки осі, а аргумент `one-axis` – одну з осей x чи y. Аргумент `axis-list` більш функціональний: він може задати як одну вісь, так і обидві.

Довжина наконечника стрілки на кожній осі визначається величиною `\axisheadlen`, яка за умовчанням дорівнює 5 пунктів. Форму наконечника стрілки також можна змінювати (див. опис макрокоманди `\arrow`).

Окрім основних осей x та y, пакет `mfpic` вводить ще чотири додаткові l, b, r та t для проведення осей зліва, знизу, праворуч та зверху рисунка відповідно. Команда `\doaxes{...}` дає можливість побудувати довільну комбінацію основних та допоміжних осей. Так, макрокоманда `\doaxes{xylbrt}` побудує всі шість осей так, що вони повністю охоплять ділянку, призначену для рисунка.

Якщо необхідно перенести осі у середину базового прямокутника на величину `num` використовуються наступні команди:

```
\axismargin{axis}{num}
\setaxismargins{leftnum}{bottomnum}{rightnum}{topnum}
\setallaxismargins{num}
```

Аргумент `axis` позначає одну з осей x, y, l, b, r, t.

Пакет `mfpic` дозволяє розмітити кожну з координатних осей невеликими перпендикулярними рисочками за допомогою таких команд:

```
\xmarks[len]{numberlist}
\ymarks[len]{numberlist}
\tmarks[len]{numberlist}
\bmarks[len]{numberlist}
```

```
\lmarks[len]{numberlist}
\rmarks[len]{numberlist}
\axismarks{axis}[len]{numberlist}
```

Перша буква команди вказує на вісь, що підлягатиме міченню. Більш універсальна команда `\axismarks{axis}[len]{numberlist}` дозволяє мітити довільну вісь (проте лише одну). Аргумент `numberlist` задає послідовність координат для проведення вертикальних/горизонтальних рисочок відносно даної осі. Ця послідовність може бути задана двома способами або як перелік значень розділених комою, або виразом {⟨мінімальне значення⟩ `step` ⟨величина прирощення⟩ `until` ⟨максимальне значення⟩}. Необов'язковий аргумент `len` задає довжину такої вертикальної рисочки. Задати довжину цих рисочок можна й глобально за допомогою параметра `\hashlen`. За умовчанням він дорівнює 4 пункти.

Для керувати розміщенням вертикальних рисочки в пакеті `mfpic` використовуються макрокоманди:

```
\setaxismarks{axis}{pos}
\setbordermarks{leftpos}{bottompos}{rightpos}{toppos}
\setallbordermarks{pos}
\setxmarks{pos}
\setymarks{pos}
```

Аргумент `pos` визначає спосіб розміщення рисочок відносно координатних осей й може приймати такі значення: `inside`, `outside`, `centered`, `onleft`, `onright`, `ontop`, `onbottom`.

Для позначення осей та/або рисок текстовими або числовими мітками використовується макрокоманда

`\axislabels{axis}[pos]`{$\{t_1\}n_1, \{t_2\}n_2, \ldots$}

Числа $n_1, n_2, \ldots$ визначають точки розташування міток $t_1, t_2, \ldots$ а опція `pos` – спосіб їхнього позиціонування, який визначається за допомогою двох символів: перший – для задання вертикального розташування (`t` – зверху, `c` – по центру, `b` – знизу), а другий – для задання горизонтального розташування (`l` – зліва, `c` – по центру, `r` – праворуч). До двох указаних символів може бути додане ціле число, що визначає кут повороту.

Для задання відстані між рисочою та точкою прив'язки мітки використовується параметр `\tlabelsep{len}`, який за умовчанням дорівнює нулеві. Для зміни цієї відстані необхідно встановити інше значення `len`.

Якщо необхідно покрити площину рисунку координатною сіткою з крапок або прямих, можна скористатися однією з наступних команд:

```
\grid[size]{xsep,ysep}
\gridpoints[size]{xsep,ysep}
\lattice[size]{xsep,ysep}
\hgridlines{ysep}
\vgridlines{xsep}
\gridlines{xsep,ysep}
```

Необов'язковий параметр `size` задає розмір крапки, а значення `xsep,ysep` вказують TEXу, з яким шагом необхідно розташовувати крапки відповідно по осі x та y.

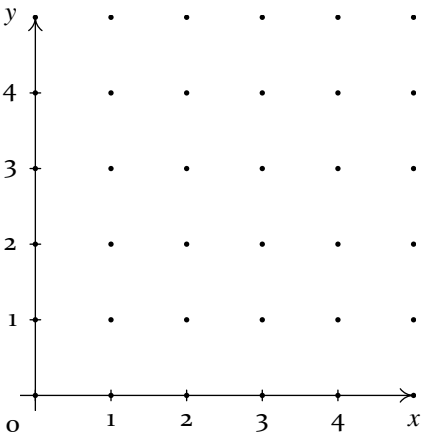
При побудові рисунка часто виникає потреба зробити текстові вставки прямо в площину рисунка з метою ідентифікувати чи пояснити його окремі деталі. Для цього в пакеті `mfpic` передбачена команда

```
\tlabel[pos](x, y){labeltext}
```

Значення `x,y` задають координати точки, в якій буде розміщено текст `labeltext`, а необов'язковий параметр `pos` вказує спосіб вирівнювання текста відносно цієї точки.

Нижче наведено приклад, у якому розглянуті деякі з наведених вище команд:

```
\begin{mfpic}{-2}{50}{-2}{50}
\doaxes{xy}
\tlabelsep{7pt}
\setallbordermarks{centered}
\xmarks{0 step 10 until 40}
\ymarks{10,20,30,40}
\axislabels{x}{{1}10,{2}20,%
{3}30,{4}40,{$x$}50}
\axislabels{y}{{1}10,{2}20,%
{3}30,{4}40,{$y$}50}
\grid[2pt]{10,10}
\tlabel(-6, -6){0}
\end{mfpic}
```



Команди створення графічних примітивів: точка, багатокутник, коло, еліпс, дуги, ламані, криві

Точка

Для відтворення точки використовується команда,

$$\texttt{\textbackslash point[size]}\{(x_0, y_0), (x_1, y_1), \dots\ \}$$

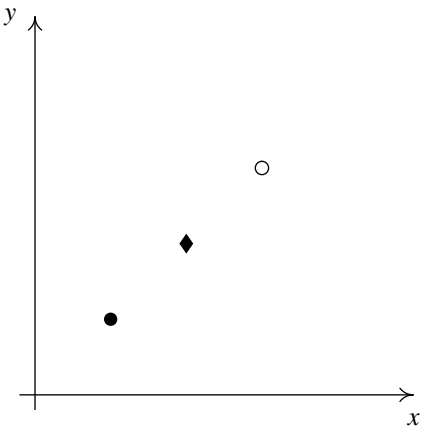
яка малює маленькі кола з центрами в точках $(x_0, y_0), (x_1, y_1)$ і т.д. Діаметр цих кіл визначає необов'язковий параметр `size`. За відсутності цього аргумента розмір точок визначається параметром `\pointsize`, який за замовчанням дорівнює 2 пункти. Передбачено можливість користуватися маленькими колами чи зафарбованими кружечками. Команди `\pointfilltrue` та `\pointfillfalse` визначають відповідно, чи будуть кола зафарбовані чи ні. За замовченням використовується `\pointfilltrue`.

Якщо замість кола необхідно поставити інший символ доцільно використовувати команду

`\plotsymbol[size]{name}{`$(x_0,y_0),(x_1,y_1),\ldots$`}`

яка подібна команді `\point`, але потрібний символ задається аргументом `name`. Цей параметр може приймати настуні значення: Triangle, Square, Circle, Diamond, SolidTriangle, SolidSquare, SolidCircle, SolidDiamond, Plus, Cross, Star.

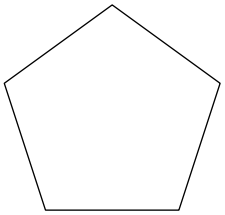
```
\begin{mfpic}{-2}{50}
             {-2}{50}
\doaxes{xy}
\tlabelsep{7pt}
\axislabels{x}{{$x$}50}
\axislabels{y}{{$y$}50}
\point[5pt]{(10,10)}
\plotsymbol[5pt]%
{SolidDiamond}{(20,20)}
\pointfillfalse
\point[5pt]{(30,30)}
\end{mfpic}
```



Багатокутник

- `\rect{`$(x_0,y_0),(x_1,y_1)$`}`
  Макрокоманда рисує прямокутник із протилежними вершинами в точках (x_0,y_0) і (x_1,y_1).
- `\regpolygon{num}{name}{`egn_1`}{`egn_2`}`
  Будує правильний многокутник з *num* вершинами. Ім'я *name* – довільне.

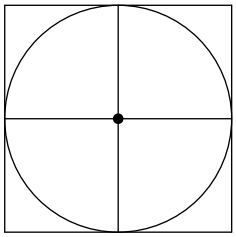
```
\begin{mfpic}{-1}{1}{-1}{1}
\regpolygon{5}{A}{A0=(0,0)}
                 {A1=(0,1)}
\end{mfpic}
```



Коло, еліпс

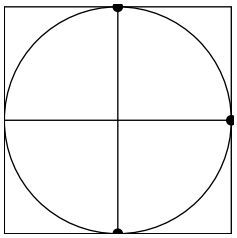
- `circle[p]{`$(x_0,y_0),r$`}`
  Побудувати коло радіуса *r* із центром у точці (x_0,y_0). Для цього випадку опцію p можна опускати.

```
\begin{mfpic}{0}{2}{0}{2}
\point[4pt]{(1,1)}
\circle[p]{(1,1),1}
\gridlines{1,1}
\end{mfpic}
```



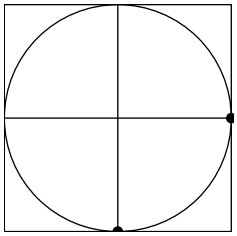
- `circle[t]{`$(a_1,b_1),(a_2,b_2),(a_3,b_3)$`}`
  Побудувати коло з трьома точками.

```
\begin{mfpic}{0}{2}{0}{2}
\point[4pt]{(1,0),(2,1),
                    (1,2)}
\circle[t]{(1,0),(2,1),
                    (1,2)}
\gridlines{1,1}
\end{mfpic}
```



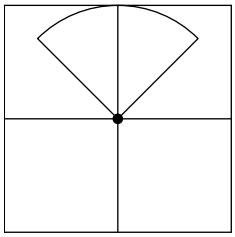
- `circle[s]{`$(a_1,b_1),(a_2,b_2),\theta$`}`
  Побудувати коло за двома точками й величиною дуги між ними θ.

```
\begin{mfpic}{0}{2}{0}{2}
\point[4pt]{(1,0),(2,1)}
\circle[s]{(1,0),(2,1),90}
\gridlines{1,1}
\end{mfpic}
```



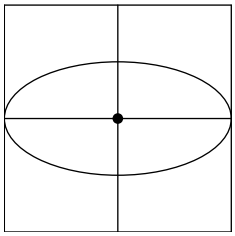
- `\sector{`$(x_0,y_0),\theta_1,\theta_2,r$`}`
  Сектор від кута θ_1 до кута θ_2 із центром (x_0,y_0), де обидва кути вимірюються в градусах проти стрілки годинника.

```
\begin{mfpic}{0}{2}{0}{2}
\point[4pt]{(1,1)}
\sector{(1,1),1,45,135}
\gridlines{1,1}
\end{mfpic}
```



- `\ellipse[`θ`]{`$(x,y),r_x,r_y$`}`
  Еліпс із центром у точці (x,y), що має ширину r_x і висоту r_y. Якщо заданий кут θ, то еліпс повертається на цю величину.

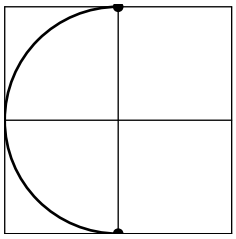
```
\begin{mfpic}{0}{2}{0}{2}
\point[4pt]{(1,1)}
\ellipse{(1,1),1,0.5}
\gridlines{1,1}
\end{mfpic}
```



Дуги

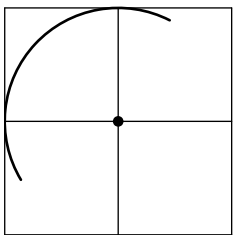
- `\arc[s]{`$(x_1,y_1),(x_2,y_2),\theta$`}`
  Дуга сполучає точки (x_1,y_1) та (x_2,y_2) так, що має при цьому θ градусів. Кут вимірюється проти годинникової стрілки.

```
\begin{mfpic}{0}{2}{0}{2}
\gridlines{1,1}
\pen{1pt}
\point[4pt]{(1,2),(1,0)}
\arc[s]{(1,2),(1,0),180}
\end{mfpic}
```



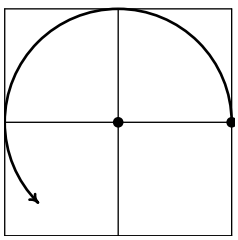
- `\arc[p]{`$(x_0,y_0),\theta_1,\theta_2,r$`}`
  Дуга з центром (x_0,y_0) та радіусом r від кута θ_1 до кута θ_2.

```
\begin{mfpic}{0}{2}{0}{2}
\gridlines{1,1}
\pen{1pt}
\point[4pt]{(1,1)}
\arc[p]{(1,1),63,211,1}
\end{mfpic}
```



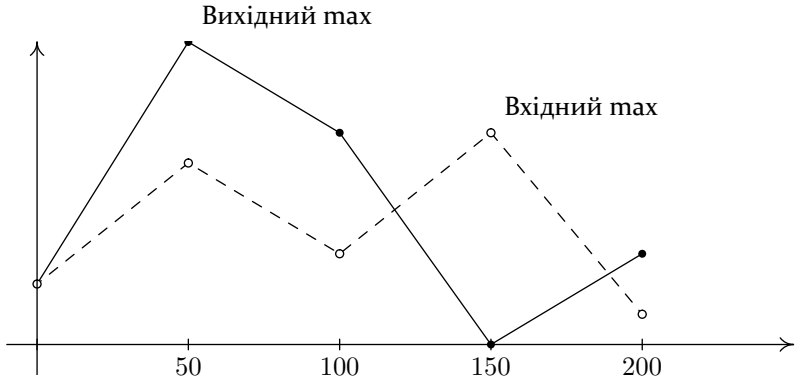
- `\arc[c]{`$(x_0,y_0),(x_1,y_1),\theta$`}`
  Дуга із центром (x_0,y_0), починаючи з точки (x_1,y_1) і проходячи в додатному напрямі кут θ відносно центра.(Це базовий спосіб побудови дуг в mfpic – усі інші зводяться до даного.)

```
\begin{mfpic}{0}{2}{0}{2}
\gridlines{1,1}
\pen{1pt}
\point[4pt]{(1,1),(2,1)}
\arrow
\arc[c]{(1,1),(2,1),225}
\end{mfpic}
```



Ламані та криві

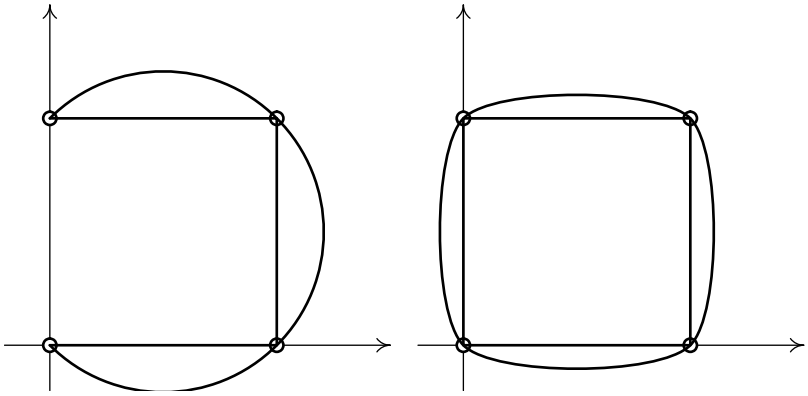
- `\lines{`$(x_0,y_0),(x_1,y_1),\ldots,(x_n,y_n)$`}`
  Команда послідовно сполучає задані точки відрізками. Результат – ламана лінія. Команда `\polylines` править псевдонімом для `\lines`.




```
\begin{mfpic}[3]{0}{2.5}{0}{1}
\tlpointsep{5pt}
\polyline{(0,.2),(.5,1),(1,.7),(1.5,0),(2,.3)}
\point[3pt]{(0,.2),(.5,1),(1,.7),(1.5,0),(2,.3)}
\tlabel[bl](.5,1){Вихідний max}
\dashed\polyline{(0,.2),(.5,.6),(1,.3),(1.5,.7),(2,.1)}
\pointfillfalse
\point[3pt]{(0,.2),(.5,.6),(1,.3),(1.5,.7),(2,.1)}
\tlabel[bl](1.5,.7){Вхідний max}
\axes
\xmarks{0,0.5,1,1.5,2}
\axislabels{x}{{$50$}.5,{$100$}1,{$150$}1.5,{$200$}2}
\end{mfpic}
```

- `\polygon{`$(x_0,y_0),(x_1,y_1),\ldots,(x_n,y_n)$`}`
  Подібна команді `\lines`, проте додатково сполучає відрізком останню й першу точки, утворюючи тим замкнутий многокутник із вершинами в даних точках.
- `\turtle{`$(x,y),(\theta_1,l_1),(\theta_2,l_2),\ldots$`}`
  Ця макрокоманда, починаючи з точки (x,y), рисує перший відрізок довжиною l_1 під кутом θ_1. Аналогічно, з кінця першого відрізка проводить другий довжиною l_2 під кутом θ_2. Цей процес продовжується для всіх пар (θ_i,l_i) подібно «графіці черепашки».
- `\curve[tension]{`$(x_0,y_0),(x_1,y_1),\ldots$`}`
  Ця макрокоманда створює криву Безьє, що сполучає точки у вказаному порядку.
- `\cyclic[tension]{`$(x_0,y_0),(x_1,y_1),\ldots$`}`
  Аналогічна попередній. Означає створити криву Безьє, що проходить через дані точки, і з'єднує першу й останню точки для створення замкнутого об'єкта.
- `\plr[tension]{`$(r_0,\theta_0),(r_1,\theta_1),\ldots$`}`
  Ця макрокоманда задає полярні координати. Не має самостійного значення, а тому застосовується з базовими командами типу `\point, \lines, \curve, \cyclic` тощо.

```
\begin{figure}[h]
\centering
\mfpic[2]{0}{1.5}{0}{1.5}
\doaxes{xy}
```



```
\penwd{1pt}
\xmarks{0,1}
\ymarks{1}
\pointfillfalse
\point[5pt]{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\lines{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\curve{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\endmfpic
\quad
\mfpic[2]{0}{1.5}{0}{1.5}
\doaxes{xy}
\penwd{1pt}
\xmarks{0,1}
\ymarks{1}
\pointfillfalse
\point[5pt]{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\polygon{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\cyclic[2]{\plr{(0,0),(1,0),(sqrt 2,45),(1,90)}}
\endmfpic
\end{figure}
```

Графіки функцій

- `\function[spec]{`$x_{min}, x_{max}, \Delta x$`}{`$f(x)$`}`

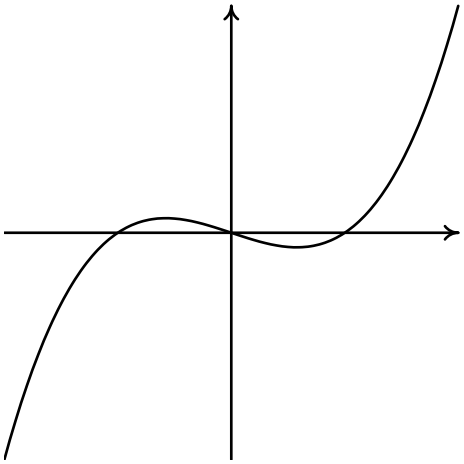
  Макрокоманди для побудови графіків функцій мають декілька параметрів. Серед них перший необов'язковий параметр *spec* визначає метод побудови кривої графіка. Якщо *spec* рівний *s*, то крива буде гладенькою (типу кривої Безьє), якщо *spec* рівний *p*, то крива буде ламаною лінією.

  Другий обов'язковий параметр містить три значення x_{min}, x_{max} і Δx. Аргумент функції x змінюється від x_{min} до x_{max} із кроком Δx.

  Третій, також обов'язковий параметр $f(x)$ містить вираз, значення якого обчислюється для величини x.

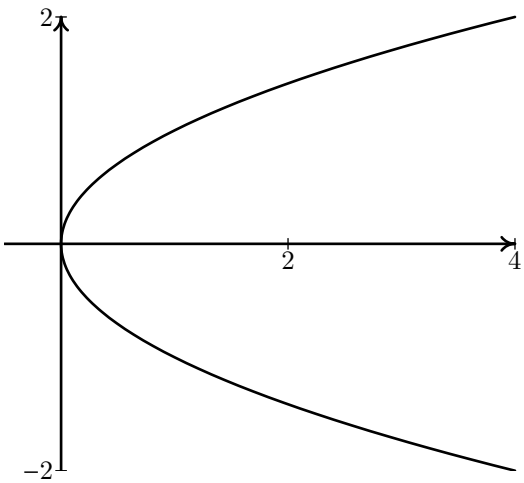
Макрокоманда будує графік залежності $f(x)$. Значення за умовчанням для необов'язкового параметра *spec* дорівнює *s*.

```
\begin{mfpic}{-2}{2}{-2}{2}
\penwd{1pt}
\doaxes{xy}
\function{-2,2,0.1}{((x**3)
                        -x)/3}
\end{mfpic}
```



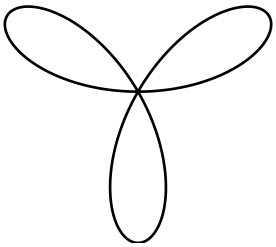
- `\parafcn[spec]{`$t_{min}, t_{max}, \Delta t$`}{`$x(t), y(t)$`}`
  Макрокоманда будує графік параметричної залежності виразів $x(t)$ та $y(t)$ від змінної *t*. Значення за умовчанням для необов'язкового параметра *spec* рівне *s*.

```
\begin{mfpic}{0}{4}{-2}{2}
\tlabelsep{3pt}
\xmarks{2,4}
\ymarks{-2,2}
\axislabels{x}{{$2$}2,
                {$4$}4}
\axislabels{y}{{$-2$}-2,
                {$2$}2}
\penwd{1pt}
\doaxes{xy}
\parafcn{-2,2,0.1}{(t**2,t)}
\end{mfpic}
```



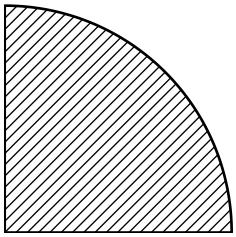
- `\plrfcn[spec]{`$t_{min}, t_{max}, \Delta t$`}{`$\rho(t)$`}`
  Ця макрокоманда рисує графік залежності в полярних координатах, де $\rho(t)$ є деяким виразом зі змінною *t*. Величина *t* інтерпретується як кут у градусах. Значення за умовчанням для необов'язкового параметра *spec* рівне *s*.

```
\begin{mfpic}{-2}{2}{-2}{2}
\penwd{1pt}
\plrfcn{0,180,1}{2*sind 3t}
\end{mfpic}
```



- `\plrregion[spec]{`$t_{min}, t_{max}, \Delta t$`}{`$\rho(t)$`}`
  Ця макрокоманда рисує криволінійний сектор у полярних координатах, який обмежений кривою $\rho(t)$ та двома прямими t_{min} та t_{max}, що виходять із центру. Вираз $\rho(t)$ залежить від кута у градусах t. Значення за умовчанням для *spec* дорівнює p.

```
\begin{mfpic}{0}{2}{0}{2}
\penwd{1pt}
\draw\rhatch
\plrregion{0,90,1}{2}
\end{mfpic}
```



2.2.3 Пакет XY-pic

Пакет XY-pic призначений для побудови різноманітних кому-тативних діаграм, для яких бувають потрібні різні стрілки: похилі, зігнуті, подвійні тощо. Цей пакет – великий і складний, і в цьому розділі ми лише на прикладах розглянемо деякі його можливості (про решту їх можна прочитати в оригінальній документації, що поширюється в електронному вигляді разом із пакетом).

Щоб можна було користуватися пакетом XY-pic для складання комутативних діаграм, слід підключити стильовий пакет xy з опціями `matrix, arrow, curve` та іншими:

```
\usepackage[matrix,arrow,curve]{xy}
```

У науковій та навчальній літературі часто виникає потреба будувати діаграми, елементами яких є формули, які певним чином розташовані на площині й сполучені стрілками. Формула тут розуміється в широкому сенсі: нею може бути довільний вираз або символ. Для побудови таких діаграм існує розширення xymatrix пакета XY.

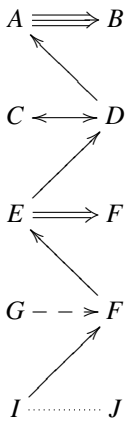
Перш ніж складати діаграму, треба в уяві розмістити її елементи у вершинах прямокутної ґратки. Рядки ґратки розділяються командами `\\`, а елементи рядка – символом `&`. Якщо в якихось вузлах ґратки елементи відсутні, то слід залишити порожнє місце. Символи `&` у необхідній кількості мають бути наявними.

```
\[
\xymatrix{A & \sum_{i=0}^n i^2 \\
\sin(x) & D}
\]
```

$$\begin{array}{ll} A & \sum_{i=0}^{n} i^2 \\ \sin(x) & D \end{array}$$

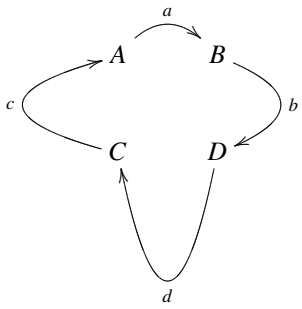
Відношення між елементами діаграми зручно зобразити стрілками. У пакеті є можливість застосовувати стрілки різних типів, причому можна вводити нові стрілки користувача. Для їх побудови існує команда `\ar`, яка може мати до п'яти аргументів, що визначають форму, напрям та різні написи. Наприклад, команда `\ar[r]` проведе горизонтальну стрілку праворуч на одну позицію (колонку), а команда `\ar[ru]` – діагональну: на одну позицію праворуч і на одну позицію вгору. Модифікатори `@{...}` задають тип стрілки. Так, `@{<--}` задає штрихову лінію, напрямлену ліворуч, а модифікатор `@{.}` – пунктирну.

```
\[
\xymatrix{
A \ar@3{->} [r] & B \\
C \ar@{<->} [r] & D \ar@{->}[lu] \\
E \ar@2{->} [r]\ar@{->}[ru] & F \\
G \ar@{-->} [r] & F \ar@{->}[lu] \\
I \ar@{.} [r] \ar@{->}[ru] & J
}
\]
```



Стрілки можна проводити не лише прямі, а й вигнуті. Робиться це за допомогою модифікатора `/.../`. Зверніть увагу, як міра вигину залежить від параметра. Також можна формувати підписи над або під стрілкою.

```
\[
\xymatrix{A\ar@/^1pc/[r]^a &
B\ar@/^2pc/[d]^b \\
C\ar@/^3pc/[u]^c &
D\ar@/^4pc/[l]^d }
\]
```



Підсумуємо викладене. Кожне позначення для стрілки складається з п'яти елементів (не всі вони обов'язкові). Перший (обов'язковий) елемент, призначений для задання стрілки, це команда `\ar`.

Другий елемент – модифікатор вигину стрілки (якщо стрілка пряма, його можна опустити). Він має вигляд `@/.../`, де на місці крапок записується вказівка про

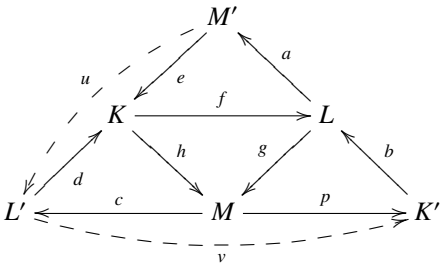
напрям і ступінь вигину. Ця вказівка складається із символа ^ або _ , за яким слідує параметр довжини. Цей параметр можна й не вказувати, написавши просто `@/^/` або `@/_/`, – тоді стрілці буде надано деякий вигин за умовчанням. У будь-якому разі символ ^ або _ вказує, в який бік вигинається стрілка: якщо ^, то вправо, якщо _, то вліво (якщо дивитися від початку стрілки до її кінця).

Третій елемент – модифікатор форми стрілки (якщо стрілка «звичайна», його можна опустити). Він має вигляд `@{...}`, де на місці крапок ставиться умовне позначення, що більш-менш імітує необхідну форму. В розглянутому прикладі цей модифікатор має вигляд `@{-->}`.

Четвертий (обов'язковий) елемент – модифікатор напрямку. Кожна стрілка розглядається як така, що йде з одного вузла ґратки в інший. Для задання напряму (точніше, точки призначення) стрілки необхідно вмістити в квадратні дужки комбінацію з букв `u` (вгору), `d` (вниз), `r` (вправо) та `l` (вліво). Наприклад, `[ddll]` означає, що пункт призначення стрілки знаходиться на нашій гратці на два кроки вниз і на два кроки ліворуч від її попереднього положення.

П'ятий і останній елемент (необов'язковий) визначає підпис при стрілці. Він складається із символа ^ або _ і тексту підпису (якщо підпис має більше одного символа, його слід узяти у фігурні дужки). Знак ^ указує, що підпис розміщений зліва від стрілки (якщо дивитися від її початку до кінця), а знак _ означає, що підпис розміщений праворуч. При одній стрілці можуть бути підписи з обох боків.

Для прикладу розглянемо складнішу діаграму:



Їй відповідає такий початковий текст:

```
$$
\xymatrix{
&& {M'}\ar@{->}[dl]^e \ar@/_1pc/@{-->}[ddll]_u \\
& K \ar[rr]^f \ar[dr]^h && L \ar[ul]_a \ar[dl]_g \\
{L'} \ar@{->} [ur]_d \ar@/_1pc/@{-->}[rrrr]_v &&
M \ar[rr]^p \ar[ll]_c && {K'} \ar@{->}[ul]_b
 }
$$
```

Завдання

Підготувати звіт, зверстаний у системі TEX, який повинен містити:

1. Титул, оформлений відповідно до вимог щодо оформлення звітів.
2. Типові сторінки спеціалізованого наукового видання (отримати у викладача) (за варіантом).

Варіанти

№ варіанту	Сторінки
1	с. 3 (тільки таблиця), с. 20 (Example 1.9), с. 25 (Figure 1.8)
2	с. 205, с. 76 (Figure 3.1)
3	с. 226, с. 77 (Figure 3.2)
4	с. 232, с. 96 (Figure 3.4)
5	с. 258, с. 135 (Figure 4.1)
6	с. 319, с. 136 (Figure 4.2)
7	с. 78, с. 138 (Figure 4.3)
8	с. 79, с. 148 (Figure 4.5)
9	с. 80, с. 176 (Figure 4.12)
10	с. 81, с. 236 (Figure 6.1)
11	с. 386-388 (до розділу Chain Letter Problem, без рисунка), с. 238 (Figure 6.2)
12	с. 94, с. 239 (Figure 6.3)
13	с. 128-129 (розділ Variation Distance, без рисунка), с. 378 (Figure 10.1)
14	с. 143, с. 417 (Figure 11.3)
15	с. 146