МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

О. М. НІКІТЕНКО

LaTeX в дії

Методичні рекомендації з використання видавничої системи LaTeX
для студентів, науковців, викладачів

Електронне видання

Харків 2018, 2019, 2020, 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
з використання видавничої системи LaTeX
для студентів, науковців, викладачів

Електронне видання

ЗАТВЕРДЖЕНО
кафедрою метрології
та технічної експертизи.
Протокол № 8 від 30.05.2018

Харків 2018, 2019, 2020, 2021

LaTeX в дії. Методичні рекомендації з використання видавничої системи LaTeX для студентів, науковців, викладачів [Електронний ресурс] / упоряд. О.М. Нікітенко. — Електрон. дані. — Харків : ХНУРЕ, 2018, 2019, 2020, 2021. – 266 с. — pfd 2,7 Мб

Упорядник О.М. Нікітенко

Рецензент Ж.В. Дейнеко, доц. каф. МСТ

Зміст

Вступ

Це електронне видання орієнтоване на читача, котрому в своїй діяльності необхідно готувати видання високої якості, тексти яких містять формули, графіки, кольорові діаграми, ілюстрації, а також на фахівців з комп'ютерної графіки.

Видавнича система LaTeX є стандартом у науковому світі. Найкращі математичні, фізичні та економічні журнали видаються в LaTeX й рекомендують авторам використовувати його для підготовки рукописів. LaTeX не є простим у його засвоєнні, але має багато переваг відносно популярних текстових процесорів й редакторів презентацій:

- високоякісне верстання — текст виглядає «як в книжці»;
- зручна робота зі складними математичними формулами;
- відмінна крос-платформна сумісність;
- автоматизація багатьох рутинних процесів:
  - нумерація формул;
  - нумерація рисунків;
  - нумерація таблиць;
  - нумерація розділів документу;
  - перехресні посилання;
  - створення колонтитулів;
  - оформлення стилів заголовків тощо;
- широке коло користувачів та розробників;
- пакети розширення "на всі випадки життя".

LaTeX полегшує життя всім, кому необхідно оформлювати есеї, реферати, курсові та атестаційні роботи, дисертації, а також слайди презентацій на їх базі. Користувачі, котрі засвоїли LaTeX, згадують верстання складних документів у Word і створення великих презентацій у PowerPoint як страшний сон.

Наразі майже єдиною потужною видавничою системою, яку реалізовано практично для всіх платформ та операційних систем і здатною створювати на їх основі ідентичні результати з повною сумісністю файлів, є TeX. Незважаючи на досить солідний вік і на появу нових надзвичайно потужних видавничих систем, популярність TeX'а в науковому середовищі всього світу продовжує зростати. Його прийнято як стандарт більшістю відомих науково-технічних видавництв світу, зокрема: Addison Wesley Longman, Springer-Verlag, John Wiley & Sons, AMS, SIAM, Kluwer, Elsevier, Мир, ТВП, Факториал.

Ця система дозволяє:

- одержувати під час друкування тексти високої поліграфічної якості робіт з математики, фізики, хімії та інших природничих наук;
- виконувати математичні розрахунки, залучати до тексту будь-які складні формули (під час друкування не мають спотворень та естетично виглядають), конструювати абзаци, знаходити точки переносу;
- не прив'язувати до якогось одного типу комп'ютерів текст, що підготовлено у вигляді файлу;
- працювати вітчизняному користувачеві з легальним програмним забезпеченням тому, що реалізація цієї програми поширюється вільно.

Видавнича система LaTeX є мовою розмітки даних та пакетом макросів TeX для створення високоякісного оформлення документів. LaTeX широко використовується для підготовки математичних та технічних текстів для публікації у наукових виданнях.

LaTeX-документ (розширення tex) — це звичайний текстовий файл, в якому міститься певна кількість команд для LaTeX процесора. У певному сенсі це програма, після виконання якої отримуємо якісно оформлену друковану чи електронну копію документа. LaTeX — найбільш популярний набір макророзширення (або макропакет) системи комп'ютерної верстки TeX, який полегшує набір складних документів. Пакет надає змогу автоматизувати багато завдань набору тексту і підготовки статей, включно з набором тексту кількома мовами, нумерацією розділів і формул, перехресними посиланнями, розміщенням ілюстрацій і таблиць на сторінці, веденням бібліографії тощо.

На відміну від текстових процесорів, що базуються на філософії WYSIWYG (What You See Is What You Get — "що бачиш — те й отримуєш"), LaTeX відокремлює зміст від оформлення документу. Тому, використавши шаблон, можна не приділяти увагу форматуванню тексту, а зосередитися на змісті й розкритті теми.

Перша версія системи комп'ютерної верстки TeX була випущена у 1979 році американським професором математики Дональдом Кнутом (Donald Ervin Knuth). Вона призначена для верстки тексту та математичних формул на високому типографському рівні. В TeX користувач задає тільки текст та його структуру, система сама на базі вибраного шаблону формує кінцевий документ, виконуючи функції і дизайнера, і верстальника.

У його сучасному вигляді TeX вийшов у світ у 1982 році. Деякі вдосконалення були зроблені у 1989 році для підтримки 8-бітних символів і багатьох мов. TeX відомий стабільністю, здатністю працювати на багатьох комп'ютерних платформах і операційних системах, відсутністю помилок.

LaTeX (http://www.latex-project.org/) — надбудова до системи TeX у вигляді набору макропакетів та макросів, що є системою більш високого рівня, ніж TeX. LaTeX був написаний Леслі Лампортом (Leslie Lamport) у 1984 році.

У 1994 році пакет LaTeX був оновлений командою LaTeX3 на чолі з Франком Міттельбахом (Frank Mittelbach) з метою вдосконалення версії LaTeX 2.09. Щоб не плутати цю версію зі старою, вона отримала назву LaTeX 2_ε. З 1998 року діє офіційний стандарт з підтримкою кирилиці в TeX/LaTeX (зокрема з українською мовою).

Система комп'ютерного набору LaTeX — це вільнорозповсюджуване програмне забезпечення, призначене для підготовки та створення науково-технічної документації високої типографської якості. Для перегляду створеного чи виправленого документу необхідно запустити окрему програму та скомпілювати його. Вихідним документом для LaTeX є звичайний текстовий ASCII файл (файл з розширенням tex, що підготовлено за допомогою будь-якого текстового редактора) з текстом документу та командами його форматування.

Для отримання кінцевого документу (готового до перегляду чи виводу на друк) вихідний файл необхідно скомпілювати за допомогою програми-транслятора.

Для внесення будь-яких змін у документ необхідно мати вихідний LaTeX документ (документ з розширенням tex) та будь-який текстовий редактор.

1 Встановлення програмного забезпечення

Перед тим, як працювати з LaTeX необхідно встановити необхідне програмне забехпечення. Тут розглянемо встановлення MiKTeX.

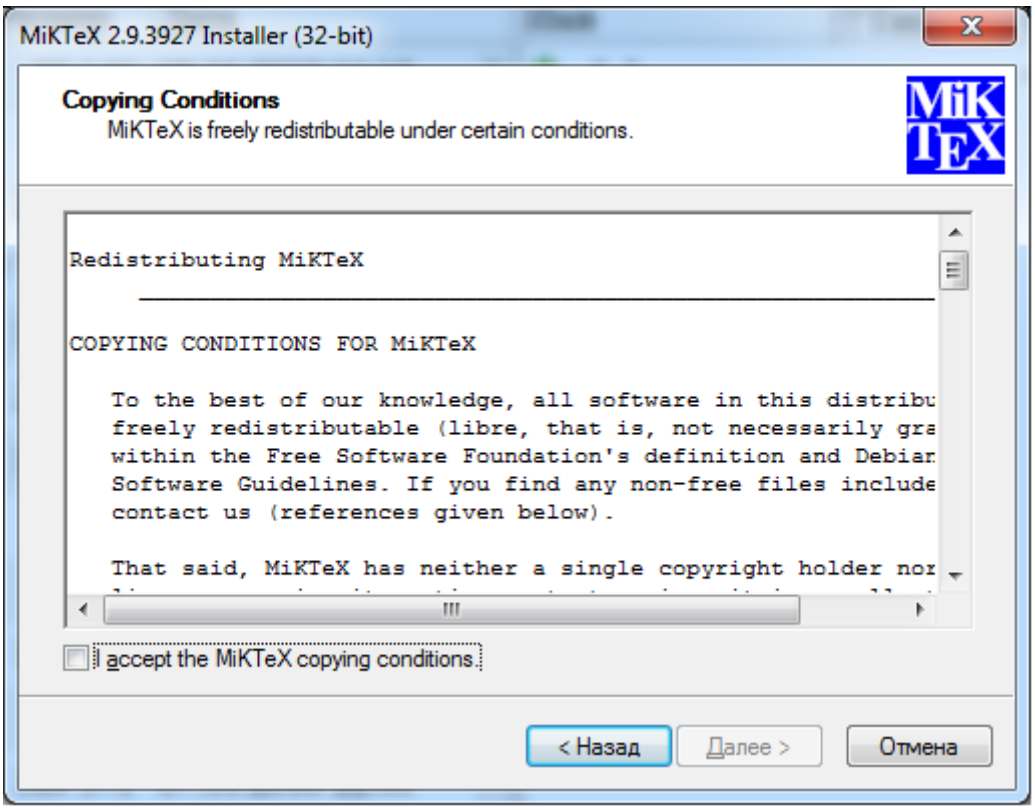
Для повноцінної роботи з втіленням TeX для Windows потрібно встановити такі програми:

- **Ghostscript** — інтерпретатор мови **PostScript**;
- **GSView** — графічна оболонка для **GhostScript** перегляду і друку **ps** і **pdf** файлів;
- **MiKTeX** — втілення TeX для Windows;
- **WinEdT** — багатофункціональний текстовий редактор (необов'язковий, але суттєво полегшує створення коду tex);
- **TeXnicCenter** — альтернатива WinEdt, що не вимагає обов'язкової реєстрації;
- **TeXCAD** — графічний редактор для створення алгоритму побудови рисунків;
- **Jpeg2ps** — програма переформатування jpg в eps [1].

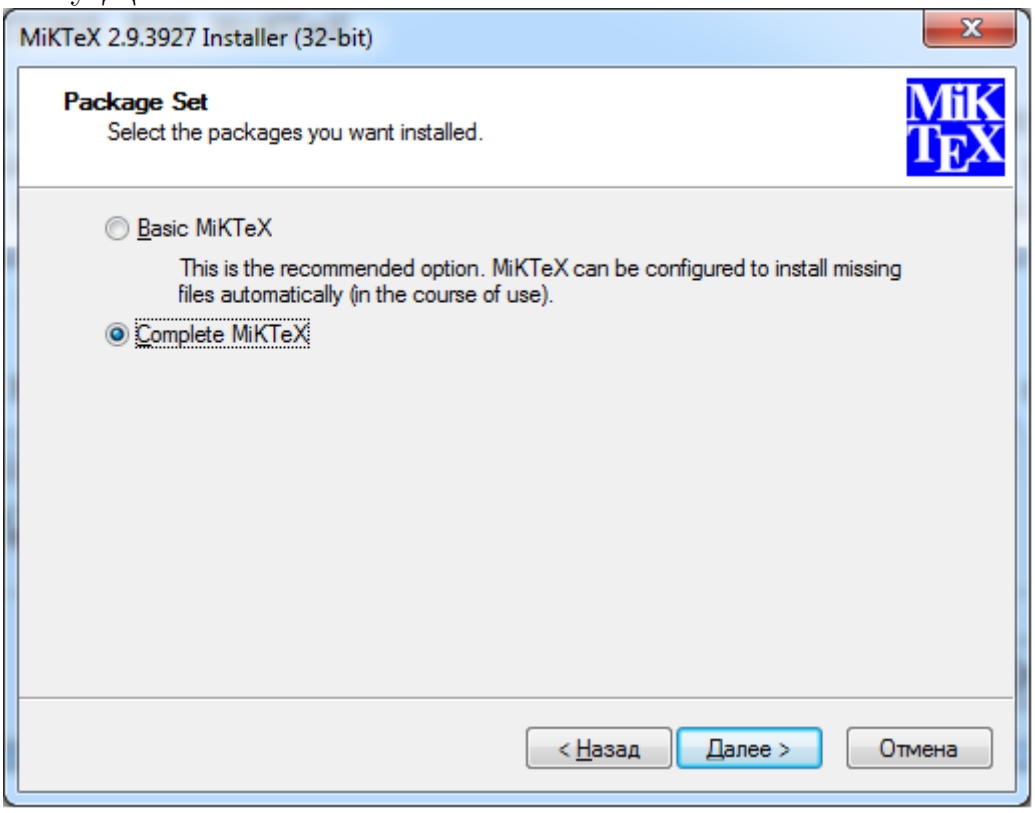
Обов'язковим є MiKTeX.

1.1 Встановлення MiKTeX

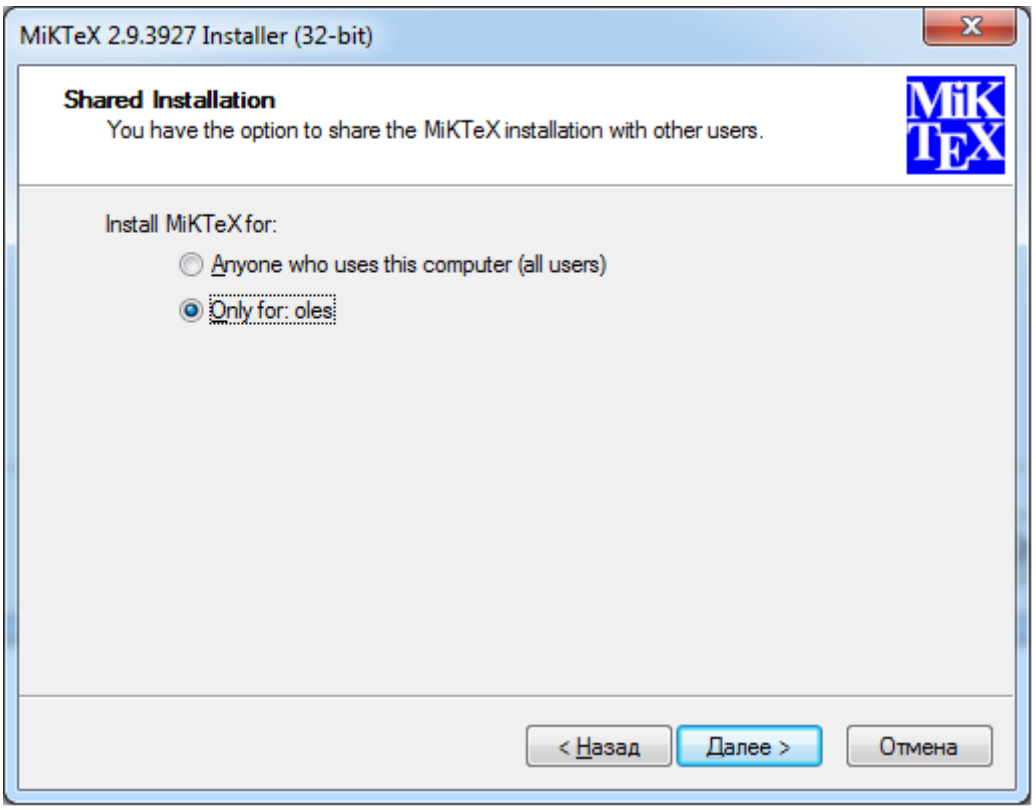
Запустіть програму **setup-2.9.3927.exe**. Відкриється вікно **MiKTeX 2.9.3927 Installer (32 bit)**, у якому Ви маєте ознайомитися з ліцензійною угодою (англійською мовою), погодитися з її положеннями й натиснути клавішу **Далі**.



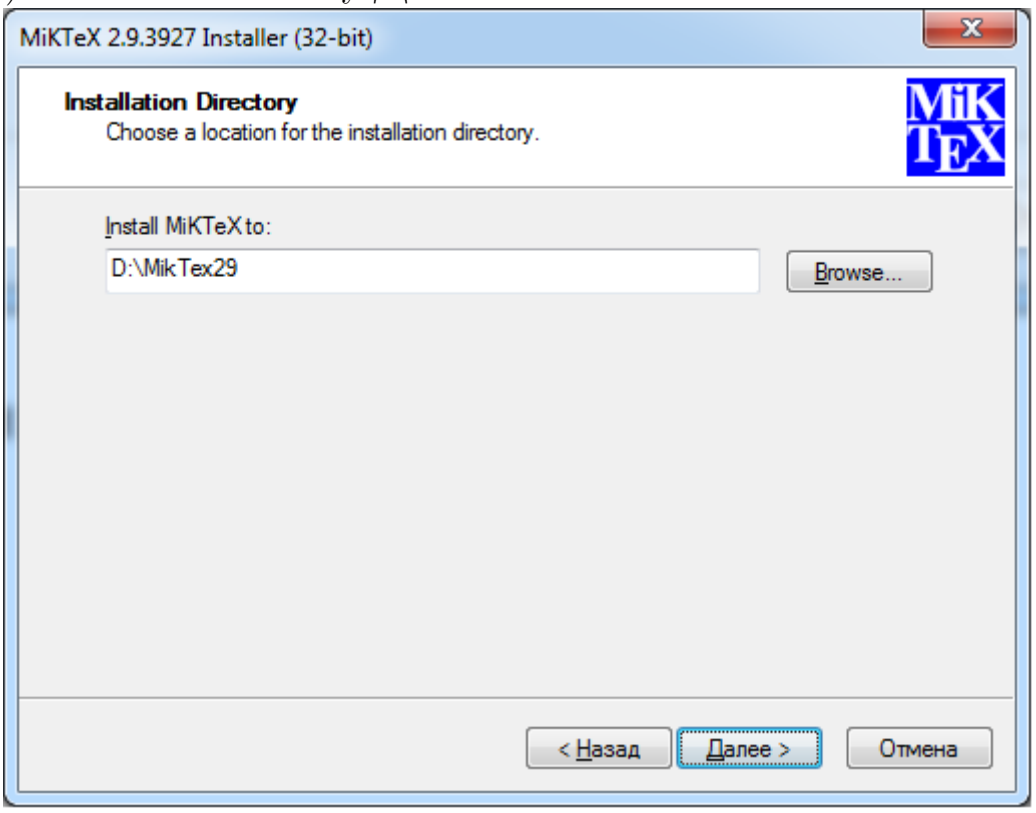
У вікні **MiKTeX 2.9.3927 Installer (32 bit)** оберіть повну версію й натисніть клавішу **Далі**.



У вікні **MiKTeX 2.9.3927 Installer (32 bit)** виберіть варіант встановлення **MiKTeX** для всіх користувачів чи одного користувача і натисніть клавішу **Далі**.

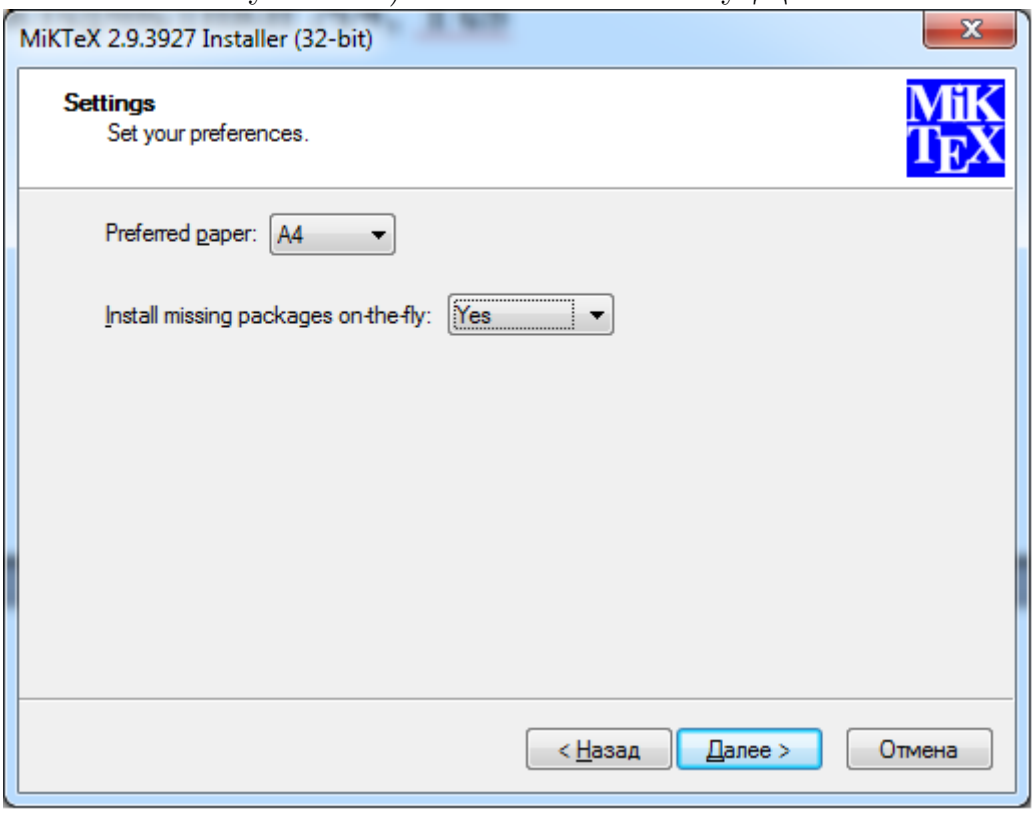


У вікні **MiKTeX 2.9.3927 Installer (32 bit)** вкажіть шлях до теки встановлення, в яку інсталюють **MiKTeX** (ця тека має бути попередньо створена і порожня) і натисніть клавішу **Далі**.

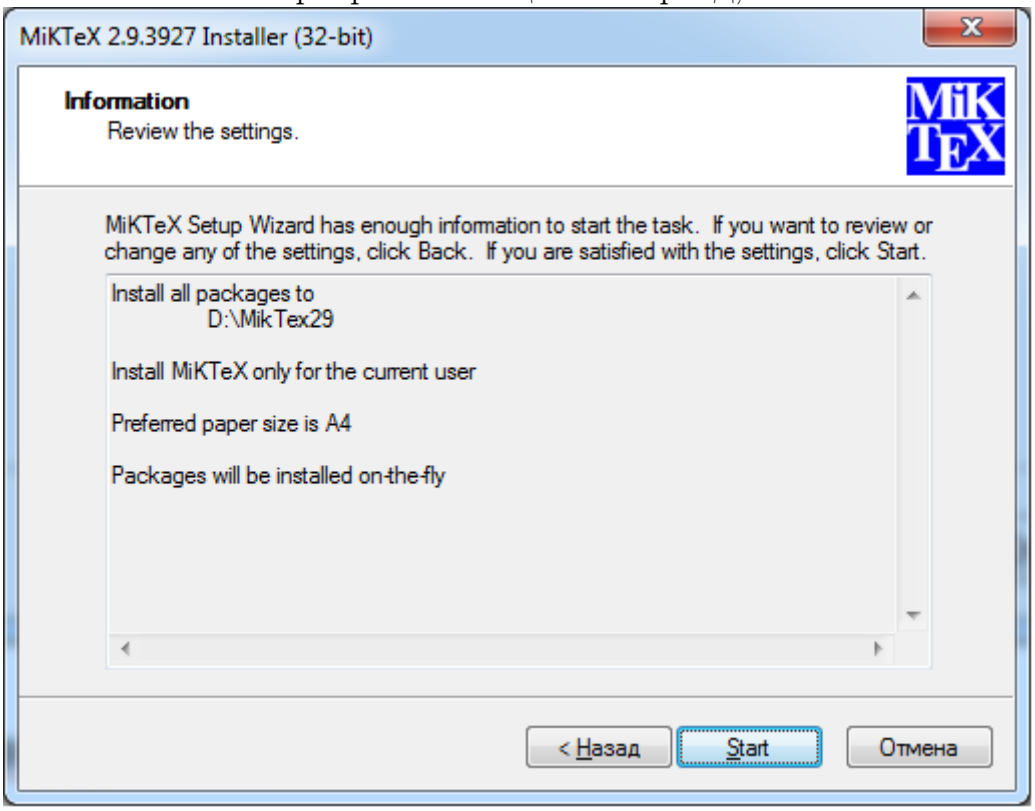


У вікні **MiKTeX 2.9.3927 Installer (32 bit)** погоджуємося з розміром паперу, хоча його можна змінити (формат паперу A4 за замовчуванням) і дії під час встановлення нових пакетів **MiKTeX**, яких не вистачає, на льоту on-the-fly

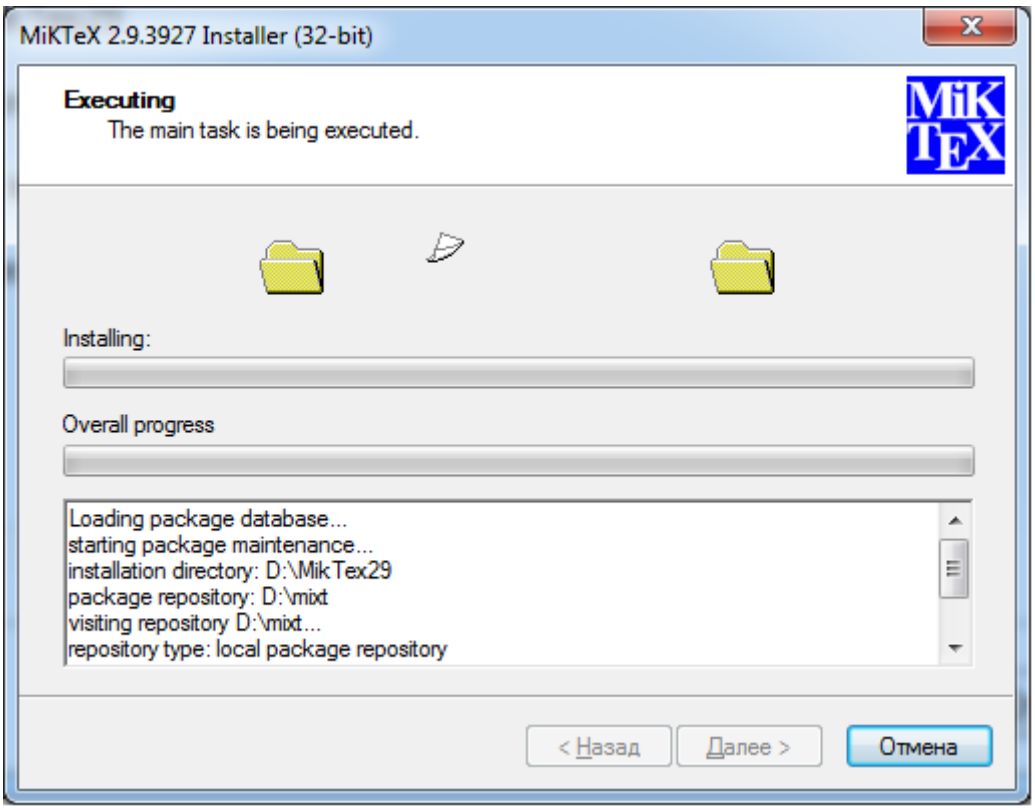
— Yes (запит за замовчуванням). Натисніть клавішу **Далі**.



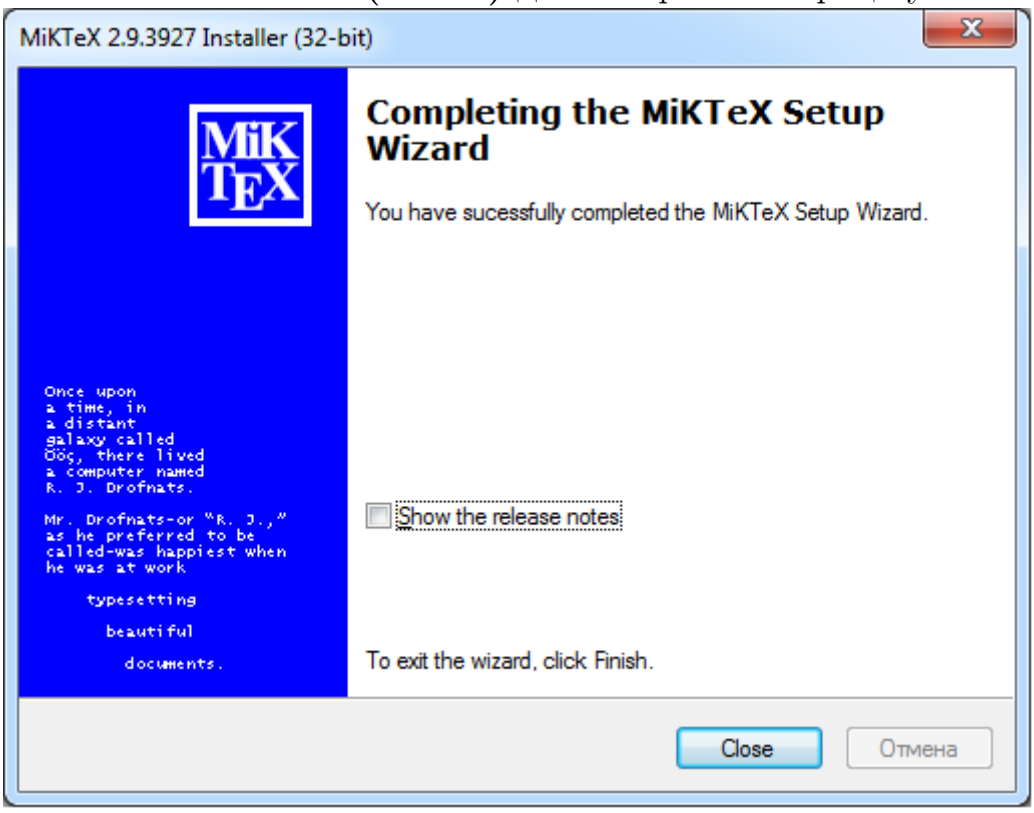
У вікні **MiKTeX 2.9.3927 Installer (32 bit)** буде повідомлено про налаштування для встановлення програми. Якщо все гаразд, натисніть клавішу **Start**.



У вікні **MiKTeX 2.9.3927 Installer (32 bit)** протягом кількох хвилин буде відображено процес встановлення.



Після завершення встановлення у вікні **MiKTeX 2.9.3927 Installer (32 bit)** натисніть клавішу **Далі**. і після цього натисніть клавішу **Close** у вікні **MiKTeX 2.9.3927 Installer (32 bit)** для завершення процесу встановлення.



На цьому процес встановлення **MiKTeX** завершується.

1.2 Налаштування MiKTeX

Після встановлення програми її слід налаштувати для подальшої роботи. Для цього у меню програм **Windows** (**Пуск\Програми\MiKTeX 2.9\Maintenance**) запустіть програму **Settings**. З'явиться вікно **MiKTeX Options** з кількома закладками.

Поки не завершено всі операції з цією програмою, не потрібно натискати клавішу **Ok** в головному вікні, інакше вікно закриється і всю процедуру налаштування потрібно буде здійснити знову і спочатку. Перехід від однієї закладки до іншої здійснюють вибором потрібної закладки, тобто наведенням курсора й натисканням клавіші миші.

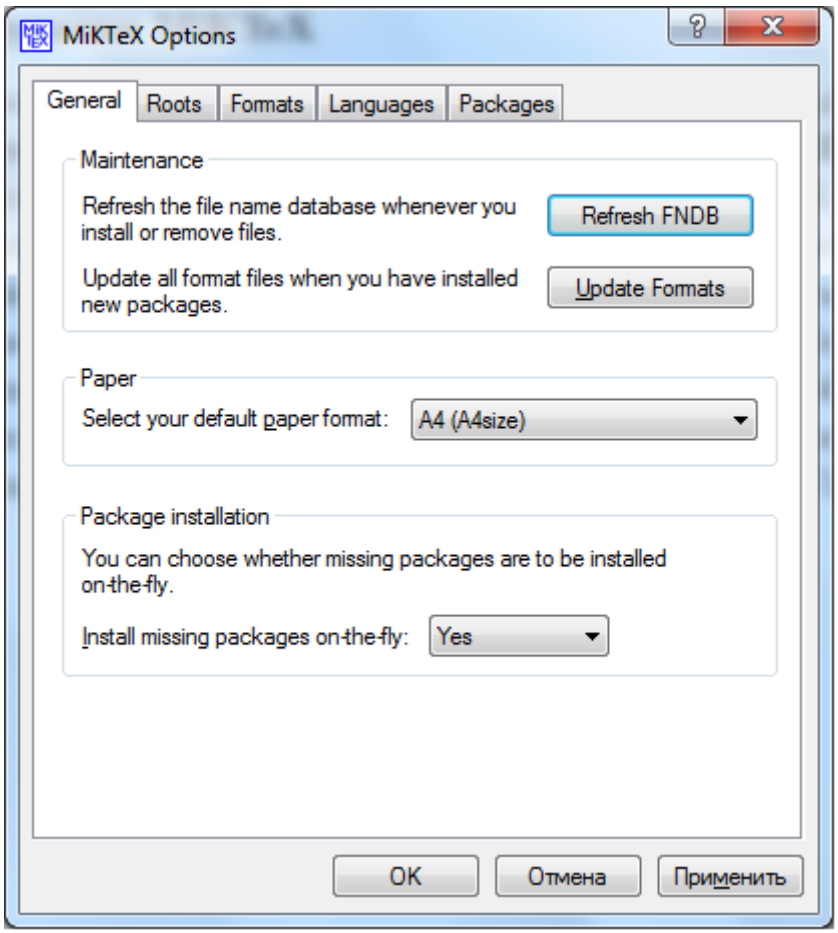
Перейдіть на закладку **Formats** і виберіть форматні файли:

- **latex** і **pdflatex** для перших кроків роботи з LaTeX;
- **tex** і **pdftex**, якщо будете створювати документи з використанням пакета макрокомманд **plain_tex**.

Рекомендують спочатку інших форматних файлів не створювати. Для під'єднання чи від'єднання форматних файлів потрібно відповідно встановити чи зняти для нього атрибут **exclude** (див. третій стовпчик у цьому вікні). Після вибору форматних файлів натисніть кнопку **Застосувати**. Якщо ніякого додаткового вибору не здійснено, то і кнопку натискати не потрібно.

Перейдіть на закладку **Languages** і виберіть мови, для яких буде підключено файли (правил) перенесення. Після цього натисніть кнопку **Застосувати**. Якщо ніякого додаткового вибору не здійснено, то і кнопку натискати не потрібно.

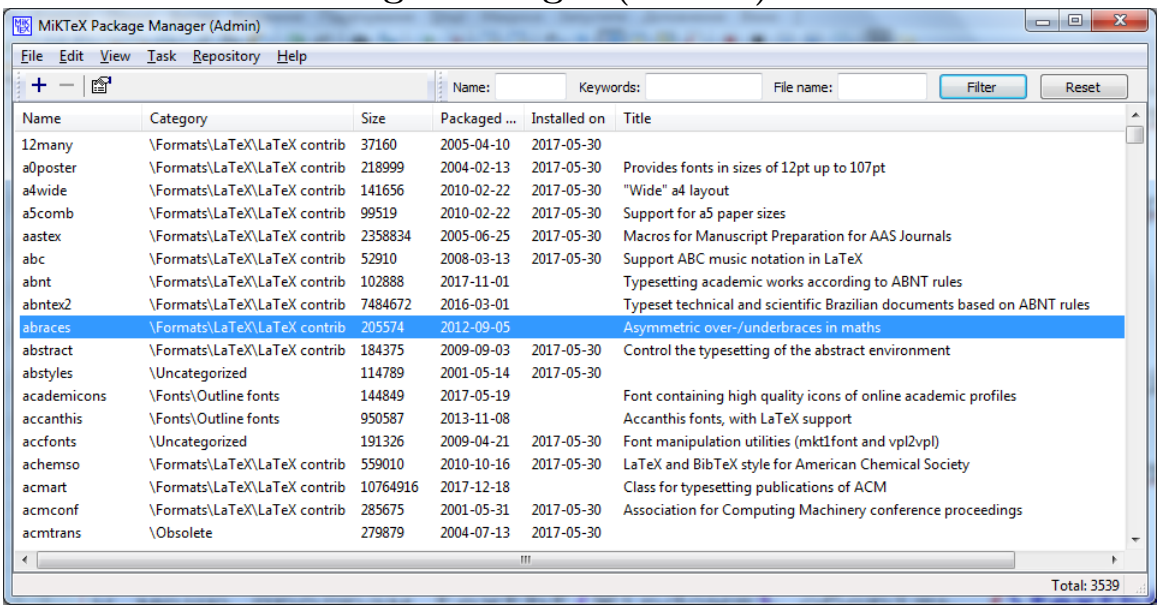
Перейдіть на закладку **Packages** і виберіть пакети для додаткового встановлення з глобальної мережі чи репозитарію.



1.3 Оновлення та додавання пакетів у MiKTeX

Пакети, які слід додати чи оновити, можна знайти в репозитаріях як в комп'ютері, так і в глобальній мережі.

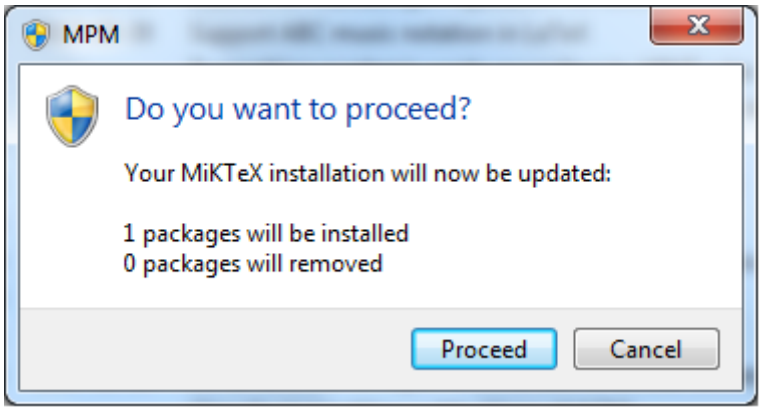
Для того, щоб додати (оновити) з репозитарію у меню програм textbfWindows оберіть (**Пуск\Програми\MiKTeX 2.9\Maintenance(Admin)**). З'явиться вікно **MiKTeX Package Manager (Admin)**.



В цьому вікні подано перелік пакетів, які можна ходяться додати (оновити).

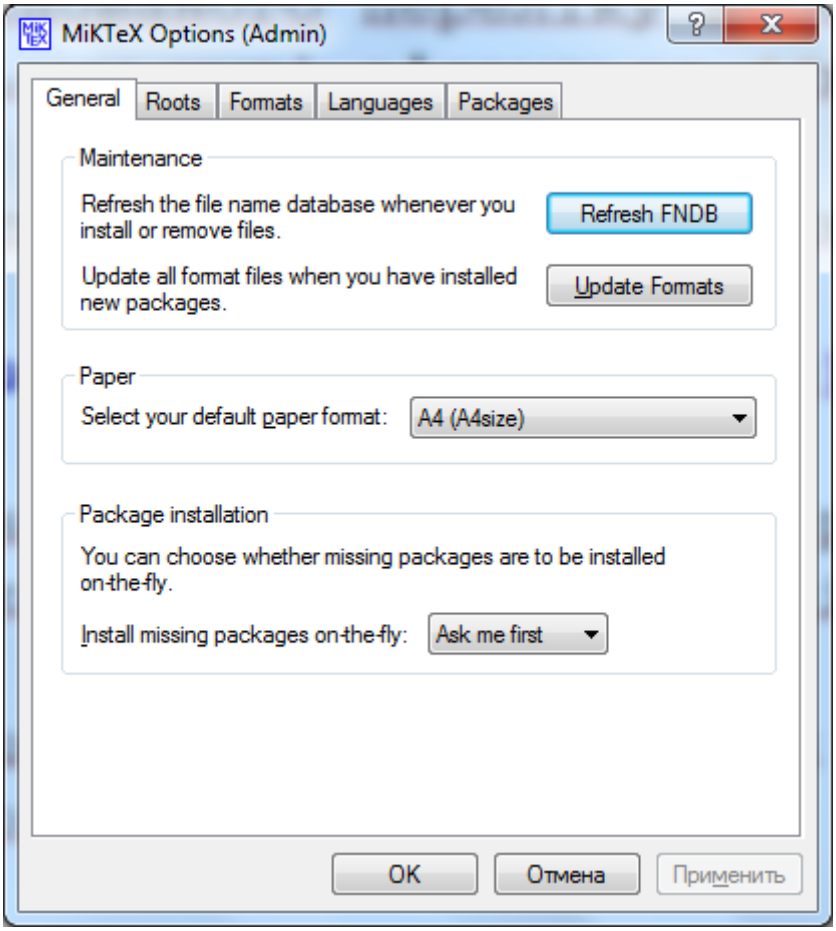
Обираєте з запропонованого переліку необхідні пакети. Якщо певний пакет ще не додано у **MiKTeX**, то у лівому верхньому куті з'явиться «жирний» хрестик.

У відповідь на натискання на цей хрестик система зпропонує встановити той чи інший пакет.



Після натискання на кнопку «Proceed» розпочнеться процес встановлення обраного пакету. Обрати можна кілька пакетів, тоді всі ці пакети встановлюватимуться послідовно.

По закінченні процесу встановлення слід натиснути кнопку «Close» і оберіть в меню програм textbfWindows (**Пуск\Програми\MiKTeX 2.9\Setting(Admin)**).



Для оновлення бази даних імен файлів слід натиснути кнопку «Refresh FNDB».

ПІсля оновлення бази даних слід натиснути кнопку «Update Formats».

Певна кількість пакетів складається з файлів **README**, ***.ins** та ***.dtx**.

У файлі **README** зазвичай описіють путь до тек, де слід встановити необхідні для роботи пакету файли.

Файли ***.ins** містять файли, які є необхідними для роботи пакету.

Файли ***.dtx** містять докумантацію до пакету.

Файли ***.ins** та ***.dtx** необхідно попередньо розпакувати.

1. розпакування ***.ins**-файлів можна здійснити одним з таких способів:
   - у командному рядку виконати таку команду **latex *.ins**, тут і далі замість * записують ім'я пакету;
   - за допомогою програми **Texworks**, викликавши з цієї програми файл **latex *.ins**.
2. розпакування ***.dtx** задля отримання документації на пакет можна здійснити одним з таких способів:
   - у командному рядку виконати таку команду **latex *.dtx**;
   - у командному рядку виконати таку команду **pdflatex *.dtx**;
   - за допомогою програми **Texworks**, викликавши з цієї програми файл **latex *.dtx**.

Після розпакування вище згаданих фійлів і розташування отриманих файлів у теках, що вказано у файлі **README**, необхідно оновити бази даних імен файлів. Таке оновлення здійснюється за процедурою, яку описано вище.

2 Структура документа

Документ LaTeX складається з трьох частин: класу документа, преамбули та власне тексту.

2.1 Клас документа

Вхідний документ LaTeX завжди починається преамбулою, яка, в свою чергу, відкривається заголовком:

\documentclass[options]{class},

де:

class — клас документу, визначає загальну структуру та особливості форматування документу;

options — необов'язковий аргумент, дозволяє змінювати значення ряду параметрів та певних правил форматування, які визначаються за замовчуванням для певного класу.

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

В LaTeX для оформлення документів розроблено такі базові стилі (на їх основі розроблено значну кількість інших, похідних стилів):

article (стаття) — стиль оформлення наукових статей, звітів чи коротких документів. Він не має поділу документу на окремі глави. Титульна сторінка розміщується вгорі першої сторінки. Стиль є найбільш прийнятним для формування більшості документів;

report (доповідь) — стиль для достатньо великих технічних документів (дипломних робіт, дисертацій тощо). Від попереднього відрізняється можливістю поділу документу на глави, а титульний аркуш формується на окремій сторінці;

book (книга) — стиль для видання книжок. Документ формується зважаючи на те, що в кінцевому варіанті він буде друкуватися на двох сторонах аркушу;

letter (лист) — стиль для написання ділових листів. Містить всі елементи для створення добре відформатованого листа: адреса, дата, підпис тощо. В преамбулі перед текстом листа оголошують ім'я, адресу та інші атрибути відправника;

beamer (презентація) — створення електронних презентацій з можливістю використання стандартних стилів оформлення, застосування анімаційних ефектів та кнопок управління переглядом.

Кожен з перерахованих стилів має свої особливості, однак всі вони побудовані за однією схемою. Кожен має ряд стильових опцій (необов'язкових параметрів, значення яких наведено в табл. 2.1 [2]). Їх значення можуть бути змінені в преамбулі документу.

Таблиця 2.1 — Стильові опції та їх можливі значення

Параметр стилю	**Можливі значення**	**Значення за замовчуванням**
Стандартний розмір шрифту	10pt,11pt,12pt	10pt
Розмір паперу	a4paper, a5paper, b5paper, letterpaper, legalpaper	a4paper
Орієнтація сторінки	landscape – альбомна	Книжна
Наявність титульної сторінки	titlepage — так, notitlepage — ні	titlepage
Положення номера у винесених формулах	leqno — номер ліворуч від формули	Праворуч
Одно- чи двосторонній друк	oneside — односторонній, twoside — двосторонній	oneside, але для класу book — twoside
Розміщення тексту в колонках	onecolumn — одна колонка, twocolumn — дві колонки	onecolumn
Сторінка, з якої може починатися нова глава	openright — тільки з правого боку розвороту, opennay — з будь-якого	Для стилю book — openright, для інших — opennay
Стиль виводу титульної сторінки	titlepage	Команда maketitle за наявності цього параметру друкує титульну сторінку на окремому аркуші і для документу в стилі article.

2.2 Преамбула

Після заголовку розташовується преамбула. Вона завантажує стилеві файли, що є необхідними для налаштування параметрів документу. Може бути розміщена безпосередньо після заголовку, міститися в окремому файлі, завантажуватися як стилевий файл. Складається з двох частин: стилевих файлів, що забезпечують налаштування документа, та налаштування параметрів сторінки.

Наведемо приклад преамбули.

```
\usepackage{animate}
\usepackage{amsmath}
\usepackage{graphicx,color}
\usepackage{cite}
\usepackage{cmap}
\usepackage[cp1251]{inputenc} % Друкування вихідного тексту.
\usepackage[ukrainian]{babel} % Підтримка української мови.
```

```
\usepackage{tikz}
\usetikzlibrary{datavisualization}
\usetikzlibrary{datavisualization.formats.functions}
\usepackage{indentfirst} % Відступ у першому рядку абзацу
\usepackage{verbatim} % Можна записувати довгі коментарі
на кілька рядків
\usepackage{pdfpages} %
%\usepackage[unicode]{hyperref}
%\usepackage[pdftex, unicode, bookmarks]{hyperref} %
Додає посилання в тексті та створює закладинки
\frenchspacing % Відстань між окремими реченнями така ж як і
між словами в реченні

% Налаштування параметрів сторінки
\oddsidemargin 0.mm % Непарна сторінка
\evensidemargin 0.mm % Парна сторінка
\topmargin 0.mm % Відстань між верхнім краєм аркуша і верхнім
колонтитулом мінус 1 дюйм.
\textwidth 160.mm % Ширина тексту
\textheight 240.mm % Висота тексту
\headheight 0.mm % Висота верхнього колонтитулу
\headsep 5.mm % Вертикальна відстань між
верхнім колонтитулом і тілом текстової сторінки
\footskip 0pt % Вертикальна відстань між  нижнім краєм тексту і
нижнім краєм нижнього колонтитула
%\pagestyle{headings}
%\def\thepage{} % Номера сторінок не ставити
\renewcommand{\refname}{Перелік посилань} % Для article
%\renewcommand{\bibname}{Перелік посилань} % Для book
 та report
\numberwithin{equation}{section} % Номера рівнянь
будуть 1.1 3.6
\numberwithin{table}{section} % Номера таблиць
будуть 1.1 3.6
\makeatletter
\renewcommand{\@biblabel}[1]{#1.\hfill}
\makeatother
%\parindent = 1.5cm % Відступ у першому рядку абзацу 1.5cm
%\graphicspath{{zu/ch1}}
```

Охарактеризуємо пакети, що містяться в преамбулі:

inputenc — для вибору кодування тексту. Для кодування Windows — 1251;
babel — пакет для визначення мов документу. Англійська мова використовується за замовчуванням;
indentfirst — відступ у першому рядку абзацу;
graphicx — відповідає за вставку графічних файлів;
color — відповідає за колір;
amsmath — відповідає за математичну нотацію;
cite — відповідає за цитування літератури;
animate — дозволяє автоматично програвати послідовність слайдів. Для того, щоб ця можливість спрацювала, необхідно використовувати Adobe Reader;
cmap — додає можливість пошуку й копіювання тексту в pdf-документі;
tikz — пакет для програмування векторної графіки;
verbatim — дозволяє подати текст буквально так, як він набраний у вихідному документі: з пробілами та розбиттям тексту рядками, й записувати довгі коментарі;
pdfpages — дозволяє маніпулювати pdf-сторінками зовнішніх pdf-документів;
hyperref — відповідає за роботу перехресних посилань команд LaTeX під час створення гіперпосилань в документі та закладинок; цей пекет бажано завантажувати до завантаження пакету babel;
frenchspacing — встановлює таку ж відстань між окремими реченнями, як і між словами в реченні: після розділових знаків пробіли не збільшуються, як це й прийнято в Україні та континентальній Європі (за замовчанням ці пробіли збільшуються у відповідності до англо-саксонської традиції).

Розглянемо команди, які найчастіше застосовують для налаштування параметрів сторінки:

\oddsidemargin — за одностороннього набору величина лівого поля задається цим параметром. При цьому поле відраховується не від самого краю аркуша: попередньо робиться відступ в один дюйм;
\evensidemargin — цим параметром задаються розміри лівого поля сторінок з парними номерами;
\topmargin — відстань між верхнім краєм аркуша і верхнім колонтитулом мінус 1 дюйм;
\textwidth — визначається ширина тексту на сторінці;
\textheight — визначається висота тексту на сторінці;
\headheight — висота верхнього колонтитулу;

\headsep — вертикальна відстань між верхнім колонтитулом і тілом текстової сторінки;

\footskip — вертикальна відстань між нижнім краєм тексту і нижнім краєм нижнього колонтитула.

Налаштування параметрів сторінки також можна здійснити, використовуючи пакет **geometry**, як показано нижче.

```
\usepackage[left=20mm, top=20mm, right=15mm, bottom=20mm]
{geometry}
```

або

```
\usepackage{geometry}
\geometry{a4paper}
\geometry{left =35mm, right =15mm, top =20mm, bottom =20mm}
\geometry{headheight =2ex, headsep =10mm, footskip =10mm}.
```

Тут:

left — ліве поле;
right — праве поле;
top — верхнє поле;
bottom — нижнє поле.

Опишемо певні корисні команди, що використовують під час обробки та форматування тексту. **\pagestyle** визначає стиль оформлення сторінок документу та може мати один з таких обов'язкових аргументів:

empty — номери сторінок не виводяться;
plain — номери сторінок виводяться внизу посередині рядка й при цьому колонтитулів не буде;
headings — номери сторінок виводяться вгорі сторінки (одночасно виводиться і назва відповідного розділу);
myheadings — номери сторінок виводяться вгорі сторінки.

Натомість команда **\thispagestyle** дозволяє поміняти стиль оформлення окремої сторінки. Вона аналогічна до **\pagestyle**, але стосується тільки тієї сторінки, на яку потрапляє текст, що її оточує.

Для автоматичного визначення кількості сторінок поточного документу доцільно скористатися пакетом **lastpage**. Він визначає мітку **lastpage** на останній сторінці документу, а команда **\pageref{lastpage}** дозволяє отримати номер його останньої сторінки.

Заголовок списку літератури генерується відповідно до стилю підключеного документу (найчастіше «Література»). За необхідності він може бути змінений командою \\ для документу зі стилем оформлення *стаття*:

\renewcommand {\refname}{Список першоджерел}

для документу зі стилями оформлення *книжка* чи *звіт*:

\renewcommand{\bibname}{Перелік джерел посилань}.

Для нумерації рівнянь всередині розділу використовують команду

\numberwithin{equation}{section}.

Для нумерації таблиць всередині розділу використовують команду

\numberwithin{table}{section}.

Щоб у бібліографічному списку номери джерел були не в квадратних дужках, це необхідно визначити таким набором команд:

\makeatletter
\renewcommand{\@biblabel}[1]{#1.\hfill}
\makeatother

У назвах таблиць та рисунків системні написи «Табл.» та «Рис.» можна замінити стандартними командами:

\renewcommand {\tablename}{Таблиця}

або

\renewcommand{\figurename}{Рисунок}.

Відступ у першому рядку абзацу можна визначити командою

\parindent = 1.5cm.

За допомогою команди **\graphicspath{{path1, path2, ..., pathn}}** визначають ті директорії, в яких розташовано графічні файли, що призначені для вставки в документ.

2.3 Розбиття вихідного файлу на частини

На обробку великого документу на малопотужному комп'ютері LaTeX може витратити значиний час. Під час редагування кількох розділів немає сенсу переробляти знову й знову весь документ, більшу частину якого вже успішно зформатували. Часто буває зручно розбити великий вихідний текст на кілька частин, що зберігаються в різних файлах [3]. Попри те, скільки використовують вхідних файлів, один з них є кореневим; це той файл, чиє ім'я отримує компілятор як вхідний параметр [6, 8].

LaTeX пропонує два способи розбиття вхідного файлу на частини. Розпочнемо з найпростішого.

Команда

\input{file}

призводить до того, що вміст файлу file LaTeX обробляє точно так нібито воно було переписано в цей файл, який містить цю команду й розташовано у тому

місці, де він знаходиться. Ім'я файлу **file** можна вказати разом з розширенням чи без нього; у останньому випадку зчитується файл **file.tex**. LaTeX зчитує файл, який вводять, від початку й до кінця або до першої команди **endinput**. Якщо файл не знайдено, LaTeX фіксує помилку й вимагає вказати ім'я іншого файлу. Команда **input** може знаходитися у будь-якому місці вхідного файлу. У свою чергу файл, який вставляють, також може містити команди **input**.

Крім зручності роботи з невеликими файлами команда **input** дозволяє легко додати один і той же текст у кілька документів. Наприклад, кореневий файл з текстом цієї книжки в момент редагування розділу міг би мати такий вигляд

```
\input{preamble}
\begin{document}
%\input{intro} % \chapter{Вступ}
%\input{install} % \chapter{Інсталяція MikTeX}
%\input{structure} % \chapter{Структура документу}
\input{textwrite} % \chapter{Набирання тексту}
\dotfill
\end{document}
```

Файл preamble.tex, який зчитується першим, починається з **documentclass** і містить всі необхідні декларації. Його можна використовувати у тестових документах для перевірки прикладів, які потім додаватимуться до книжки. Наступні три команди **input** «закоментовано», що дозволило «відключити» великий фрагмент тексту, який складається з вступу і двох перших розділів. Залишаючи «незакоментованим» якійсь один файл, можна швидко обробити по частинах увесь документ. Однак такий спосіб має недолік, тому що збиває нумерацію сторінок и робить неможливими посилання на «відключені» розділи. Зрозуміло, після завершення редагування всіх розділів можна прибрати знаки коментаря % й пропустити через компілятор увесь текст цілком [3, 6].

Кожну команду **input** варто розташовувати в окремому рядку.

Другий спосіб подрібнення вхідного файлу дозволяє підтримувати правильну нумерацію сторінок й посилань під час всієї роботи. У цьому випадку замість **input** використовують команду **include{file}**, а в преамбулу необхідно додати декларацію

includeonly{file-list},

де file-list містить список файлів, що перелічено через кому, пробілу після коми не має бути. Правило йменування файлів тепер інше: розширення імені файлу опускають, а зчитується файл з розширенням tex; однак, якщо його не знайдено, компілятор не фіксує помилку, а друкує попередження. Файли з розширеннями, які відрізняються від tex, за допомогою цієї командт підключати не можна.

Команда **include** насамперед викликає перехід на нову сторінку (якщо поточна сторінка не є порожньою), а потім зчитує файл file. Якщо файлу file немає

у списку file-list, то команда **include** просто викликає перехід на нову сторінку (якщо поточна сторінка не є порожньою).

На відміну від **input** команда **include** не має знаходитись раніше команди **begin{document}**, через те, що викликає перехід на нову сторінку, а в преамбулі можуть знаходитися тільки декларації, які не здійснюють реальних дій. Щоб «відключити» певний файл, тепер не треба вилучати команду **include** з його іменем; досить вилучити ім'я файлу зі списку file-list.

Тоді текст файлу не буде зчитаний, але вся інформація про встановлені в ньому мітки, нумерації сторінок, рисунків, таблиць тощо буде вилучена з файлу file.aux, тому наступна частина вхідного файлу буде оброблена так, якщо б file.tex був дійсно зчитаний і оброблений. Файл file.aux оновлюється кожного разу, коли файл file.tex обробляється LaTeX'ом, тобто його ім'я присутнє у списку файлів в **includeonly**.

Через те, що фрагмент тексту, що зчитується командою **include**, завжди починається з нової сторінки, то цю команду зручно використовувати для додавання розділів, які, як правило, починаються с з нової сторінки. Якщо обробляти файли за порядком, в якому їх перелічено в командах **include**, то текст всього документу збереже наскрізну нумерацію сторінок, розділів, формул тощо. При зміні вмісту будь-якого розділу слід обробити знову всі наступні розділи, щоб відновити правильну нумерацію перехресних посилань, хоча цю операцію можна відкласти до закінчення всієї роботи.

Існує два обмеження на вміст файлу, що вказано в аргументі команди **include**. По-перше, команди **include** не можуть бути вкладені одна в одну, тому файл, який додають, не може містити команду **include**. По-друге, всередині файлу, який додають, не можна визначати нові лічильники за допомогою **newcounter**. Оскільки декларація **newcounter** має глобальну область дії, найбільш прийнятним місцем для її розташування є преамбула кореневого вхідного файлу [3, 6].

3 Набирання тексту

Текст можна набирати, не турбуючись про кількість пробілів між словами — зайві пробіли ігноруються.

Порожній рядок означає кінець абзацу, який можна також задати командою \par або \\. Розбиття абзацу на рядки, перенесення слів і вирівнювання тексту відбуваються автоматично.

Вхідний файл документу, що готують до публікації засобами LaTeX'а, містять інформацію двох типів: текст і розмітку (команди форматування). Текст здебільшого вводять безпосередньо з клавіатури, хоча існують символи, вводити які необхідно певним чином, тому що вони можуть бути відсутні на клавіатурі, або LaTeX використовує їх як керуючі символи.

Через натискання відповідних клавіш вводять літери латинської абетки як маленькі, так і великі. Крім цього вводять літери української абетки, арабські цифри, та спеціальні символи, які є на клавіатурі.

Символи $, #, %, & можна набирати за допомогою керуючих символів \$, \#, \%, \&. Будь-який символ можна також набрати за допомогою команди \char{i}, або \symbol{i} просто вказавши номер символа в кодуванні ASCII.

У багатьох мовах використовують літери з додатковими значками, які розташовують над або під літерою. Такі значки називають діакритичними знаками. Команди для набору літер спеціального вигляду наведено в табл. 3.1, 3.2 [?].

Таблиця 3.1 — Діакритичні знаки в LaTeX

Набрано	Отримано	Набрано	Отримано
\'{a}	á	\u{a}	ă
\`{a}	à	\v{a}	ǎ
\^{a}	â	\H{a}	a̋
\"{a}	ä	\c{a}	ą
\~{a}	ã	\d{a}	ạ
\={a}	ā	\b{a}	a̲
\.{a}	ȧ	\t{aa}	a͡a
\k{a}	ą	\r{a}	å
\C{a}	a̧	\f{a}	ȃ
		\U{a}	ă

Якщо виникає необхідність виділити частину тексту, то використовують символи групування { і }. Текст, що розташовано між символами групування, називається групою. Команди й змінні, що розташовано всередні групи, є локальними щодо тексту поза символів групування, а їх дія обмежена группою. Після виходу з групи відновлюються початкові значення змінних, що задано до входу в группу.

Таблиця 3.2 — Літери різних абеток

Набрано	Отримано	Набрано	Отримано
\oe	œ	\OE	Œ
\ae	æ	\AE	Æ
\aa	å	\AA	Å
\o	ø	\O	Ø
\l	ł	\L	Ł
\i	ı	\j	ȷ
\ss	ß		

3.1 Переноси в словах

Зазвичай TEX не робить переносів, коли від слова відділяються дві останні літери. Це відповідає нормам англійської орфографії, але для української мови таке обмеження є недоречним. Тому під час роботи з українськими текстами варто встановити такий режим, за якого перенесення двох останніх літер є дозволеним. Такий режим можна встановити за допомогою зміни параметру **righthyphenmin**. Значення цього параметру — ціле число, що дорівнює найменшій кількості літер в слові, яке можна переносити на наступний рядок. Якщо написати у файлі

righthyphenmin=2

то під час обробки всіх абзаців, які розташовано після цієї команди, переноси двох останніх літер будуть дозволені. Якщо зустрінеться абзац англійського тексту, то перед його початком введіть команду

righthyphenmin=3

щоб не зробити неправильних переносів у англійських словах.

Іноді все ж таки слова переносяться неправильно. Щоб здійснити одноразовий переніс у слові, місце переносу позначають \-.

Якщо таке слово зустічається в тексті неоднаразово, то можна скористатися командою **hyphenation** [6].

3.2 Тире та лапки в LaTeX

TEX вирізняє кілька видів рисочок: дефіс, знак «мінус», знак переносу, коротке тире, довге тире. Коротке тире має ширину (приблизно) як латинська літера «N» (en-dash), довге — як латинська літера «M» (emdash). В українській мові використовують лише одне тире — довге, а як символ переносу використовують дефіс. Розстановка тире та дефісів регулюється правилами орфографії та пунктуації української мови.

В документі можуть використовуватися три різні типи тире:

- одинарне тире (дефіс) як з'єднувальний знак у складних словах. Ніколи не відділяється пробілами від попереднього та наступного слова;

– подвійне (коротке) тире як знак границі *від* і *до*;

— потрійне (звичайне) тире. Замінює собою пропущене слово.

Ситуація з лапками виглядає складніше. Для типографського набору базовими лапками є запозичені з французької мови «ялинки». Додатковими для набору цитати в цитаті, є запозичені з німецької мови „лапки“. При використанні лігатур TEX'а, «ялинки» набираються кутовими дужками: << і >>, а „лапки“ набираються двома поруч розташованими комами ,, і двома поруч розташованими гравісами (grave accent) ``. В рукописному тексті в українській мові використовують польські „лапки”. За допомогою тих же лігатур вони набираються двома поруч розташованими комами ,, і двома поруч розташованими апострофами '' . Всі способи набору рисочок та лапок зведено до табл. 3.3 [2].

Таблиця 3.3 — Тире та лапки в LATEX

Назва	**TEX**	**Вигляд**
дефіс	-	-
тире	---	—
мінус	$-$	−
ялинки ліві	<<	«
ялинки праві	>>	»
лапки ліві	``	“
лапки праві	``	“
лапки ліві порп.	''	”
лапки праві порп.	''	”

Такі розділові знаки, як крапка, кома, три крапки, двокрапка, знак оклику та питання, ставлять поруч (без пробілу) до попереднього тексту і окремо до наступного.

3.3 Одиниці довжини в LATEX

Визначимося з одиницями довжини, які використовують в LATEX. TEX базується на метричній системі мір. Всі одиниці вимірювання перераховуються в масштабні пункти (sp), в яких і здійснюються обчислення. Для спрощення комп'ютерних обчислень Дональд Кнут дещо змінив значення типографських пунктів та ввів точну таблицю перерахунку одиниць вимірювань (табл. 3.4)

3.4 Пробіли, інтервали та бокси

Для TEX'а окремий символ шрифта є одним з найпростіших боксів. **Бокси** (рамки) — це об'єкти прямокутної форми, для яких визначені висота, ширина, глибина і базова лінія. Група символів, з'єднуючись по горизонталі, вирівнюється по базовій лінії. Відстань між базовими лініями сусідніх рядків або

Таблиця 3.4 — Одиниці довжини в LaTeX

pt	пункт = 1/73 in ≈ 0,35 мм (американський стандарт)
pc	піка = 12 pt
bp	великий пункт (DTP-пункт) 1/72 in
sp	масштабний пункт 1/65536 pt
mm	мм
cm	см = 10 mm
in	дюйм = 25,4 mm
dd	пункт Дідо ≈ 1,07*pt* (європейський стандарт)
cc	цицеро = 12 dd
em	приблизно ширина літери «M» поточного шрифту
ex	приблизно висота літери «x» поточного шрифту

міжрядковий пробіл чи інтервал називається **інтерліньяжем**. Інтервал між рядками формується системою самостійно, виходячи зі стилю документу.

Як стандартний інтерліньяж LaTeX задає інтервал, який приблизно на 20% перевищує розмір шрифта. Такий інтерліньяж називають «через один інтервал». Зміни міжрядкового інтервалу здійснюють кількома способами.

1. Використовуючи пакет **setspace**. Цей пакет підтримує три значення інтерліньяжу:

   - **singlespacing** — один інтервал (за замовчанням);
   - **onehalfspacing** — півтори інтервали;
   - **doublespacing** — два інтервали.

   Крім цього пакет **setspace** вводить нові оточення **singlespacing**, **onehalfspacing** та **doublespacing**, які дозволяють змінити інтерліньяж всередині тексту.

2. **linespread** ставиться у преамбулі документу з обов'язковим параметром — величиною міжрядкового інтервалу. Стосується всього документу. Наприклад **linespread{2.0}** задає подвійний міжрядковий інтервал.

3. **renewcommand{\baselinestretch}{x}** дозволяє змінити міжрядковий інтервал в межах групи. **x** задає відношення нової величини міжрядкового інтервалу до стандартної величини, що визначається стилем документу [2, 5].

Сторінка тексту — це крупний бокс з вертикальних боксів, які складаються в свою чергу з горизонтальних боксів. Бокси можна створювати такими командами:

- **makebox[width][position]{text}** — бокс заданої ширини з вирівняним рядком тексту;

- **\mbox{text}** — бокс з рядком тексту;
- **\framebox[width][position]{text}** — бокс заданої ширини з вирівняним рядком тексту і оточений рамкою;
- **\fbox{text}** — бокс з рядком тексту в рамці;
- **\fcolorbox{colbox}{colbgd}{text}** — бокс з рядком тексту в кольоровій рамці;
- **\parbox[position]{width}{text}**— бокс з абзацем тексту;
- **\raiseboxdistance[above][below]{text}** — зсунутий бокс;
- **\hbox{text}** — бокс з рядком тексту в поточному режимі;
- **\vbox{{text 1}\{text 2}}** — створюється вертикальний бокс з боксів у фігурних дужках.

Всі команди, крім двох останніх, здійснюють перехід в горизонтальний режим (побудова горизонтального списку для абзацу). Рамка довкола бокса є лінією товщиною **\fboxrule**, що відділена від тексту на відстань **\fboxsep**. Аргументи в квадратних дужках не є обов'язаковими. У списку прийнято такі позначення:

width — ширина боксу; якщо її не задано, ширина боксу дорівнює ширині тексту **text**;

position — розташування тексту **text** всередині боксу; набуває значень: **c** — в центрі (за замовчуванням), **l** — притиснутий ліворуч, **r** — притиснутий праворуч;

text — текст, що розташовано всередині боксу; може містити команди, крім команд завершення рядка (типу **\par, \begin** тощо);

distance — зсув тексту відносно базової лінії;

above, below — фіктиві розміри боксу над і під базовою линією для верстки;

colbox — колір рамки боксу;

colbgd — колір тла.

Команди **\framebox**, **\fbox** та **\fcolorbox** виділяють текст рамкою тільки в одному рядку.

Для виділення тексту, який складається з кількох рядків, рамкою, в тому числі й кольоровою з фоном певного кольору, можна за допомогою такаго набору команд:

\fbox{%

\parbox{width}{text}%
}

Тут замість команди **\fbox** можуть бути використані команди **\framebox**, **\fcolorbox**.

Бокси один з одним поєднуються клеєм. Клей — це порожній простір, який має властивості стискання та розширення. Клей задається трьома параметрами: номінальною величиною, стисканням і розширенням. Клей ніколи не стиснеться менше дозволеної величини, але здатний розширюватися. Після визначення правильної (з точки зору LaTeX'а) величини клея бокс стає жорстким і більше не змінюється.

Можна задавати різні пробіли командами, що наведено в табл. 3.5. Останні два пробіли в цій таблиці можливо використовувати тільки в математичному режимі. Для верстки можна послуговуватися пробілами різної ширини.

Таблиця 3.5 — Пробіли в LaTeX

TeX	Назва	Ширина
\quad	тонкий мат. пробіл	1 em
\qquad	подвійний мат. пробіл	2 em
\enskip	n-пробіл	0.5 em
\␣	звичайний пробіл	залежить від шрифта
~	нерозривний пробіл	дорівнює ␣
\thinspace, \,	тонкий пробіл	0.16667 em (1/6 em)
\negthinspace, \!	від'ємний тонкий пробіл	-0.16667 em
\:	середній пробіл	2/9 em
\;	товстий пробіл	5/18 em
\/	пробіл після курсиву	
\@	пробіл після знаків пунктуації	

Задавати точні горизонтальні проміжки пробілами не можна — ширина пробілу трохи змінюється під час вирівнювання рядка. Для цього використовують команду **\hspace{size}** — горизонтальний пробіл довжини **size**. Якщо такий пробіл має працювати на початку чи в кінці рядка, використовуйте варіант команди «з зірочкою» — **\hspace*{size}**.

Для того, щоб визначити додаткові вертикальні інтервали, можна використовувати команди **\vspace{size}** і **\vspace*{size}**, де параметр **size** визначається подібно до попередньої команди. Якщо перша з вказаних команд не завжди утворює додаткові інтервали, наприклад, на початку сторінки, то друга позбавлена цього недоліку. Для того, щоб визначити вертикальні інтервали, слугують ще три команди зсуву на фіксовану відстань: **\smallskip, \medskip, \bigskip**. Розміри цих інтервалів залежать від класу документу та опцій в цих класах.

Іноді виникає необхідність заповнити текст пробілами до кінця рядка чи порожніми рядками до кінця сторінки. Для цього використовують клей **\fill** нульової довжини й значної розтягливості. В горизонтальному режимі з цим

клеєм можна використовувати команду **hfill**, яка є макрокомандою **hfill** = **hspace{****fill}**. Аналогічну роботу у вертикальному режимі виконує команда **vfill**.

Інструкції `\vfill` та `\hfill` додають пробіли нескінченої довжини по вертикалі та горизонталі.

Оточення **minipage** є боксом фіксованої ширини й використовують його для формування шапки з підписами. Основна властивість боксу полягає в тому, що його не можна розрізати і не можна частинами перенести зі сторінки на сторінку.

Якщо ліворуч від боксу **minipage** вказати команду `\hfill`, то бокс буде притиснений до правої межі тексту.

Команда `\textwidth` визначає ширину тексту, що дозволяє вказувати частину від нього, якщо потрібно. Аналогічно можна використовувати й висоту тексту `\textheight`.

Для створення лінійок використовують команду **rule**, яка дозволяє створювати чорні прямокутники. Вона визначена як
rule[distance]{width}{thickness}, де **thickness** — товщина лінії.

Наведемо кілька прикладів:

```
Впишіть слово \rule{5cm}{0.2pt}, яке позначає одиницю часу.
```
Впишіть слово _____, яке позначає одиницю часу.
```
Не намагайтеся прочитати слово \rule{3cm}{12pt}.
```
Не намагайтеся прочитати слово ■.
```
Широка лінія посередині рядка \rule[2pt]{3cm}{4pt}.
```
Широка лінія посередині рядка ■.
```
Широка лінія під рядком \rule[-3pt]{3cm}{4pt}.
```
Широка лінія під рядком ■.

3.5 Шрифти в LaTeX

В LaTeX 2_ε визначені три основних сім'ї (**family**) шрифтів: з засічками (roman), рублений (sans serif) и моноширинний (monospace або typewriter). Перемикання між ними здійснюється трьома командами: **rmfamily**, **sffamily,** **ttfamily**.

Всередині кожної шрифтової сім'ї можна вибрати насиченість (**series**) залежно від товщини шрифту: \mdseries (middle, середній), \bfseries (**bold, жирний**). Накреслення шрифту (**shape**) залежить від його форми й вибирається командами: \upshape (normal, прямий), \itshape (*italic, курсив*), \em (*виділений*).

Всі перелічені команди є «деклараціями» й діють до закінчення групи символів чи оточення. Для кожної команди визначена парна команда, яка друкує тільки свій аргумент вибраним текстом: \textbf для \bfseries, \textmd для

\mdseries тощо (табл. 3.6—3.11).

Розміри шрифтів в документі визначаються класом документа. Стандартні класи LaTeXa (**article, report, book**) дозволяють встановлювати базовий розмір шрифта (основного шрифта для тексту документа), що дорівнює 10, 11 і 12 пунктам. При цьому визначаються не тільки базовий розмір шрифта, але й розмір шрифта заголовків всіх рівнів, підписів, колонтитулів, зносок тощо (табл. 3.9).

Пакет **extsizes** дозволяє у стандартному класі документа, що використовують, скористатися в преамбулі командою типу **\usepackage[14pt]{extsizes}**, яка замінює базовий розмір шрифта документа (в прикладі — 14pt). Як стандартний інтерліньяж LaTeX задає такий інтервал, що приблизно на 20 % перевищує розмір шрифта. Цей інтерліньяж часто називають «через один інтервал». Зміни інтерліньяжу зручно робити за допомогою пакету **setspace**. Він підтримує три значення для інтерліньяжу: через один, через півтора і через два інтервали.

Наведеними в табл. 3.6 командами можна змінювати розмір шрифту відносно розмірів базового шрифта. Всі перелічені команди є «деклараціями» й діють до закінчення групи символів чи оточення. В табл. 3.6 вказані розмір шрифту (чисельник дробу) та інтерліньяж (знаменник дробу) в пунктах. В табл. 3.7 подано вигляд шрифтів різного розміру.

Таблиця 3.6 — Розміри шрифтів в LaTeX

Команди LaTeX'у	**Розмір базового шрифта**				
	10pt	11pt	12pt	14pt	Приклад
\tiny	5/6	6/7	6/7	7/8	tiny
\scriptsize	7/8	8/10	8/10	9/11	scriptsize
\footnotesize	8/10	9/11	10/12	11/14	footnotesize
\small	9/11	10/12	11/14	12/15	small
\normalsize	10/12	11/14	12/15	14/18	normalsize
\large	12/14	12/14	14/18	17/20	large
\Large	14/18	14/18	17/22	20/25	Large
\LARGE	17/22	17/22	20/25	25/30	LARGE
\huge	20/25	20/25	25/30		huge
\Huge	25/30	25/30			Huge

Якщо за окремих причин необхідно шрифт більшого розміру, можна скористатися командами зміни розмірів довільного боксу з пакету **graphicx**. Текст має бути набрано векторними шрифтами.

Отриманий текст не може перевищувати ширину рядка. Крім того, ці команди зручно використовувати для зміни розмірів рисунків та таблиць. Команда **\resizebox{ !}{2cm}{Великий}** дозволяє отримати літери висотою 2 см. Знак оклику як перший аргумент вказує на те, що ширина має бути обчислена з висоти з врахуванням збереження співвідношення сторін вихідного боксу.

Таблиця 3.7 — Розміри шрифтів в LaTeX залежно від класу

Шрифт	**Розмір (article, book)**	**Розмір (beamer)**
\tiny	малесенький	малесенький
\scriptsize	дуже маленький	дуже маленький
\footnotesize	досить маленький	досить маленький
\small	маленький	маленький
\normalsize	нормальний	нормальний
\large	великий	великий
\Large	більший	великий
\LARGE	дуже великий	
\huge	величезний	
\Huge	гігантський	

Таблиця 3.8 — Абсолютні розміри шрифтів у стандартних класах в LaTeX

Розмір	**10pt (за замовч.)**	**опція 11pt**	**опція 12pt**
\tiny	5pt	6pt	6pt
\scriptsize	7pt	8pt	8pt
\footnotesize	8pt	9pt	10pt
\small	9pt	10pt	11pt
\normalsize	10pt	11pt	12pt
\large	12pt	12pt	14pt
\Large	14pt	14pt	17pt
\LARGE	17pt	17pt	20pt
\huge	20pt	20pt	25pt
\Huge	25pt	25pt	25pt

Таблиця 3.9 — Сім'ї шрифтів в LaTeX

\rmfamily	шрифт з засічками, схожий на TimesNewRoman
\sffamily	шрифт без засічок, схожий на Arial
\ttfamily	`шрифт типу друкарської машинки`

Таблиця 3.10 — Насиченість шрифтів в LaTeX

\mdseries	середня насиченість
\bfseries	**напівжирна насиченість**

Весь текст, який розміщений за символом %, важається коментарем, і тому не виводиться під час друкування. Символ відсотка можна вивести за допомогою команди \%, а символ ~ формує нерозривний пробіл.

Зазвичай текст вирівнюють за шириною. Однак у певних випадках виникають ситуації, коли текст необхідно вирівняти у інший спосіб. Для цього використовують відповідні оточення або команди.

центр Оточення **center** або команду **\centering** використовують для вирівнювання вмісту по центру.

Таблиця 3.11 — Накреслення шрифтів в LaTeX

\upshape	прямий
\itshape	*курсив*
\slshape	*нахилений*
\scshape	Капітель

праворуч Оточення **flushright** або команду **flushright** використовують для вирівнювання вмісту по правому боку.

ліворуч Оточення **flushleft** або команду **flushleft** використовують для вирівнювання вмісту по лівому боку.

4 Додавання графічних об'єктів

Найбільш поширеним способом додавання в документ растрової та векторної графіки є використання пакету **graphics** чи його «розширеної» версії **graphicx**. Обидва визначають основну команду додавання рисунків `\includegraphics`. Реалізація цієї команди майже однакова, відмінності існують лише серед необов'язкових параметрів. Однак використанню розширеної версії надається перевага.

Компілятор pdfLaTeX не має вбудованих засобів обробки мови **Postscript** й тому не може вставляти в документ **eps**-файли. Його орієнтовано на використання драйвера pdftex і він може залучати до документу **pdf-, jpg-, png**-файли.

Файли з зображеннями варто створювати поза LaTeX, використовуючи спеціалізовані графічні редактори або систему комп'ютерної математики Maple. В документ можна вставляти як векторні картинки, так і растрові. За можливістю краще використовувати векторні формати тому, що вони дозволяють себе масштабувати без погіршення якості зображення.

У найпростішому випадку растрова графіка вставляється в документ командою `\includegraphics[options]{file}`, де параметри **options** визначають деякі додаткові умови вставлення рисунку (наведено окремі параметри):

scale — збільшення чи зменшення рисунку;

width — зміна ширини рисунку;

height — зміна висоти рисунку;

keepaspectratio — збереження співвідношення висоти і ширини рисунку під час перетворень;

angle — повернення рисунку на потрібний кут (в градусах) проти годинникової стрілки;

file — ім'я файлу, що вставляють в документ. Можна використовувати ряд графічних форматів: **eps, png, jpg**. Якщо графічний та вихідний файли знаходяться в різних каталогах, то необхідно задати повне ім'я графічного файлу, або означити в преамбулі директорії, де розташовано графічні файли за допомогою команди **\graphicspath{{path1, path2, . . . , pathn}}.**

Якщо ширина чи висота рисунку явним чином не задана, то вона обчислюється автоматично, зважаючи на збереження масштабу.

Команда `\includegraphics[options]{file}` може використовуватися як для вставки окремих рисунків, так і для вставки невеликих графічних зображень безпосередньо в текст.

4.1 Побудова графіків функцій в LaTeX

До документу можна залучати графіки, що створено безпосередньо засобами LaTeX. Зі всіх графічних можливостей розглянемо лише два пакети:

tikz — пакет для програмування векторної графіки;

pgfplots — доповнення до пакету tikz для побудови графіків, які базуються на табличних даних.

Для використання можливостей цих пакетів вони мають бути підключені в преамбулі документу:

```
\usepackage{tikz}
\usepackage{pgfplots}
```

Власне графік функції формується в межах оточення tikzpicture:

```
\begin{tikzpicture}
Параметри графіка
\end{tikzpicture}
```

Всі команди для побудови графіка «Параметри графіка» задаються в оточенні **axis** та команди `\addplot`. Основні параметри оточення **axis** наведено в табл. 4.1. Можливі параметри команди `\addplot` наведено в табл. 4.2. Типи маркерів, що можна використати для побудови графіків наведено в табл. 4.3.

Таблиця 4.1 — Значення параметрів для оточення axis

Параметр	**Призначення**
xmin=значення	Початкове значення x для побудови графіка
xmax=значення	Кінцеве значення x для побудови графіка
ymin=значення	Початкове значення y для побудови графіка
ymax=значення	Кінцеве значення y для побудови графіка
Ytick={y1,y2,y3,...}	Координати для відображення написів за віссю y
yticklabels={напис1,напис2,напис3,...}	Написи за віссю y
Xtick={y1,y2,y3,...}	Координати для відображення написів до осі x
xticklabels={напис1,напис2,напис3,...}	Написи до осі x
ymax=значення	Явне задання діапазону для побудови графіка за віссю ординат
xmax=значення	Явне задання діапазону для побудови графіка за віссю абсцис
xlabel=напис	Підпис за віссю x
ylabel=напис	Підпис за віссю y
height=значення	Висота області побудови графіка
width=значення	Ширина області побудови графіка

Більш гнучкою у використанні є команда `\addplot+`.

Графічні можливості пакетів проілюструємо на ряді прикладів.

Таблиця 4.2 — Параметри команди \addplot

Параметр	Призначення
xmin=значення	Початкове значення x для побудови графіка
only marks	Відображаються тільки маркери точок
mark=none	Відображається тільки ламана лінія, що поєднує точки
mark size=розмір	Розмір маркера
color=значення	Колір маркера
thick	Товста лінія для графіка товщиною 0.8pt
very thick	Дуже товста лінія для графіка товщиною 1.2pt
solid	Суцільна лінія
doted	Лінія у вигляді крапок
densely doted	Лінія у вигляді крапок зі зменшеною відстанню між ними
loosely doted	Лінія у вигляді крапок зі збільшеною відстанню між ними
dashed	Лінія у вигляді пунктиру
densely dashed	Лінія у вигляді пунктиру зі зменшеною відстанню
loosely dashed	Лінія у вигляді пунктиру зі збільшеною відстанню
draw=колір	Колір лінії графіка

Таблиця 4.3 — Можливі типи маркерів для побудови графіків (для команди \addplot)

Параметр	Вигляд маркера
mark=*	
mark=x	
mark=square	
mark=square*	
mark=triangle	
mark=triangle*	
mark=diamond	
mark=diamond*	
mark=pentagon	
mark=pentagon*	

4.1.1 *Побудова графіка функції за точками*

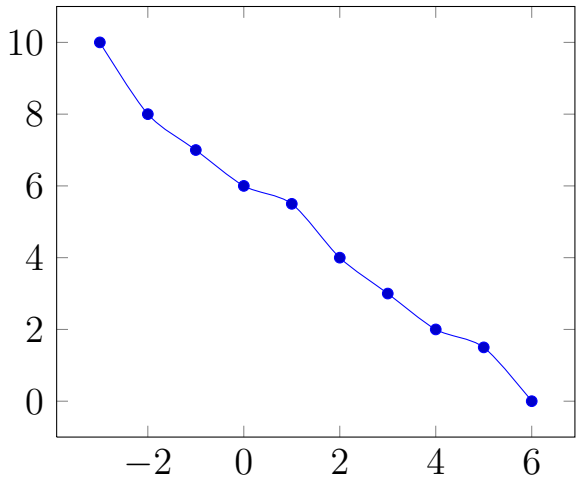
Побудуємо графік за точками, що задані своїми координатами. Точки на цьому графіку з'єднуються прямими лініями.

```
\begin{tikzpicture}[title=Графік за точками]
\begin{axis}
%задаються координати точок для побудови графіка
%координати точок задаються у форматі: (x,y)
```

```
\addplot coordinates {
( -3, 10 )
( -2, 8 )
( -1, 7 )
( 0, 6 )
( 1, 5.5 )
( 2, 4 )
( 3, 3 )
( 4, 2 )
(5,1.5)
(6,0)};
\end{axis}
\end{tikzpicture}
```

Результат компіляції:

Графік за точками

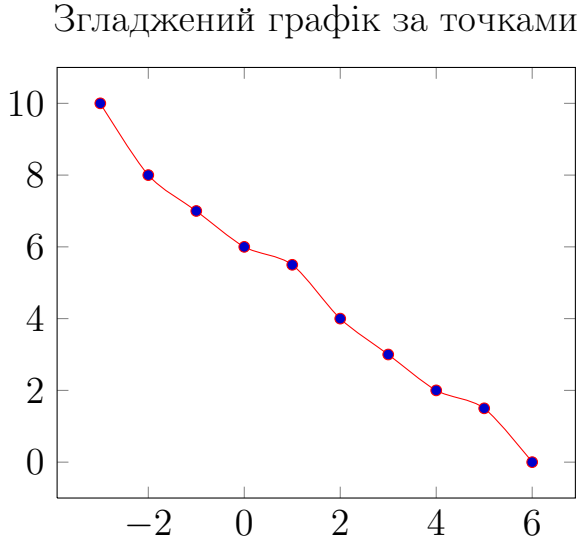


Для побудови згладженого графіка з'єднання точок використовують опцію [smooth]:

```
\begin{tikzpicture}[title=Згладжений графік за точками]
\begin{axis}
%задаються координати точок для побудови графіка
%координати точок задаються у форматі: (x,y)
\addplot +[smooth, draw = red] coordinates {
( -3, 10 )
( -2, 8 )
( -1, 7 )
( 0, 6 )
( 1, 5.5 )
```

```
( 2, 4 )
( 3, 3 )
( 4, 2 )
(5,1.5)
(6,0)};
\end{axis}
\end{tikzpicture}
```

Результат компіляції:



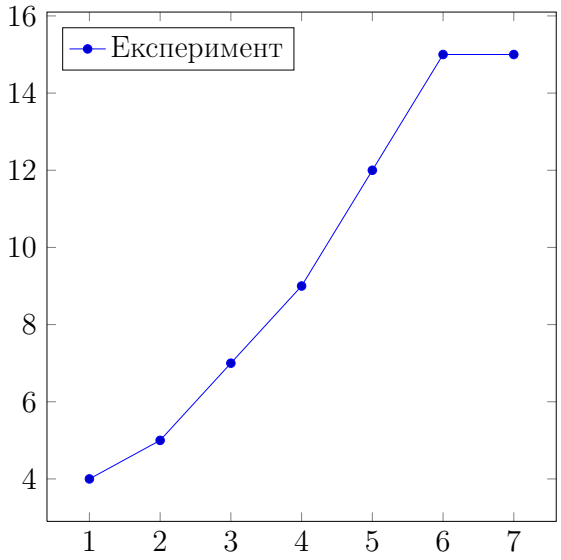
4.1.2 *Побудова графіка функції за даними з файлу*

Дані для побудови графіка функції можна задати в окремому файлі.

```
\begin{tikzpicture}[scale=0.9]
% задано розмір графіка height=10cm,width=10cm
\begin{axis}[height=10cm,width=10cm,
legend pos={north west},title=Графік за даними з файлу]
%дані читаються з файлу zzz.txt
\addplot file {zzz.txt};
% до графіка додається легенда: Експеримент
\addlegendentry{Експеримент}
\end{axis}
\end{tikzpicture}
```

Результат компіляції:

Графік за даними з файлу



Дані читаються з файлу zzz.txt, який містить координати точок у два стовчики, як це показано на рис. 4.1.

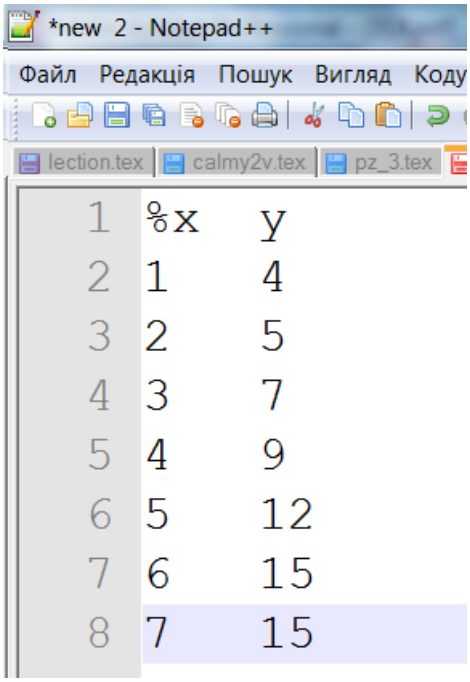


Рисунок 4.1 — Координати точок у фійлі zzz.txt

Файл може бути підготовлений за допомогою будь-якого зовнішнього редактора. До цього можна додати необхідні коментарі — рядок, що починається символом % або # під час побудови графіка буде проігнорований.

Команда `\addlegendentry{Експеримент}` додає легенду до графіка. Місце розташування легенди на графіку визначається параметром legend pos={місце}, який знаходиться в необов'язкових параметрах оточення axis. Легенда на графіку може роташовуватися:

north west — у верхньому лівому куті;
north — вгорі;
north east — у верхньому правому куті (за замовчуванням);
east — праворуч;
south east — у нижньому правому куті;
south — внизу;
south west — у нижньому лівому куті;
west — ліворуч.

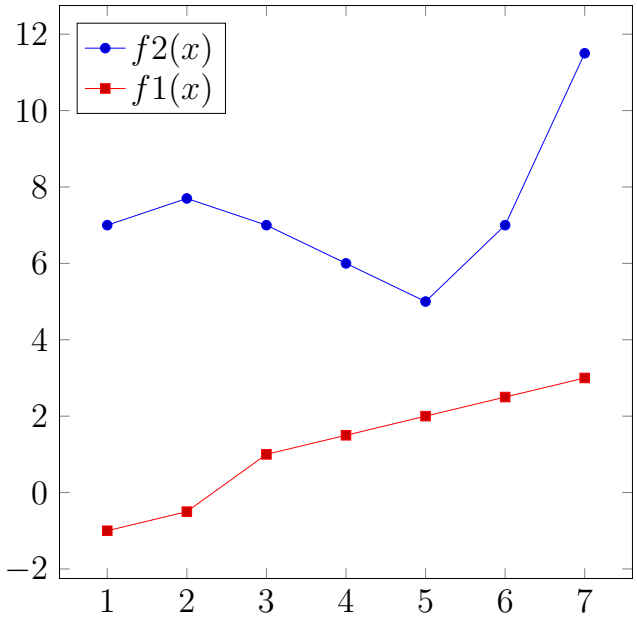
Параметр **scale** оточення **tikzpicture** дозволяє задати масштаб області для побудови. У прикладі вище він складає 90 % (scale=0.9) від реально заданих розмірів (height=10cm, width= 10cm).

Якщо у файлі міститься кілька колонок даних, то команда `\addplot` table дозволяє будувати графіки функцій для вказаних наборів даних. Наприклад:

```
\begin{tikzpicture}[scale=0.9]
\begin{axis}[height=10cm,width=10cm,legend pos={north},
title=Графік за даними з файлу у кілька колонок]
%задання колонок даних для побудови двох графіків
%дані читаються з файлу zzz2.txt
\addplot table[x=x,y=f2(x)] {zzz2.txt};
\addplot table[x=x,y=f1(x)] {zzz2.txt};
% вивід легенди до першого графіка
\addlegendentry{$f2(x)$}
% вивід легенди до другого графіка
\addlegendentry{$f1(x)$}
\end{axis}
\end{tikzpicture}
```

Результат компіляції:

Графік за даними з файлу у кілька колонок



Для побудови графіків сформовано файл **zzz2.txt**. Перший рядок файлу з даними містить назви відповідних стовпчиків. Назви використовуються для вибору відповідних наборів даних під час побудови, як це показано на рис. 4.2.

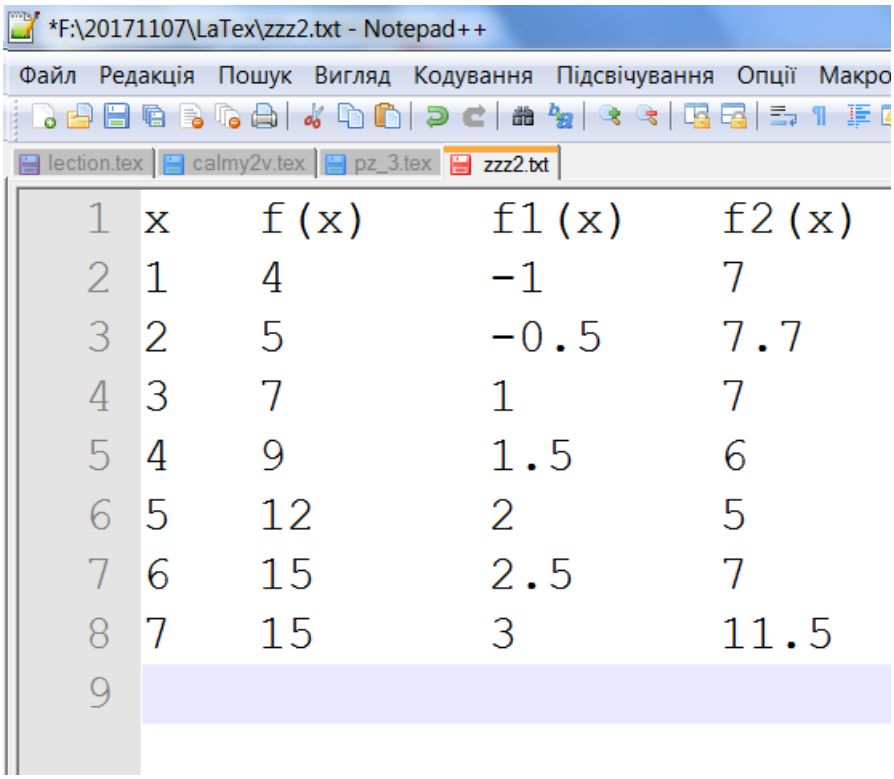


Рисунок 4.2 — Координати точок у чотири стовпчики у фійлі zzz2.txt

Побудова кожного з графіків здійснюється командою:
\addplot table[x=x,y=f2(x)] {zzz2.txt};

Вибір набору даних, що відображаються на графіку, здійснюється за допомогою параметрів **[x=x,y=f2(x)]** — дані беруться з першого та четвертого стовпчиків файлу. Аналогічним чином будується другий графік з параметрами **[x=x,y=f1(x)]**.

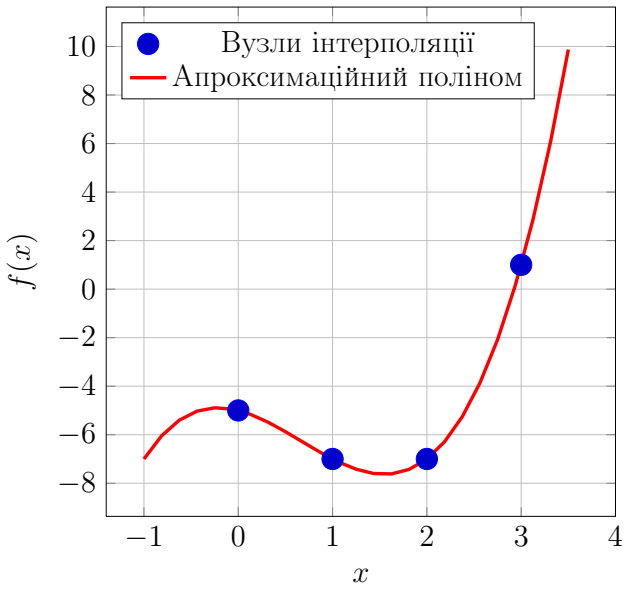
4.1.3 *Побудова графіка за заданою функцією*

У прикладі будуються графіки двох функцій ($f(x) = x^3 - 2x^2 - x - 5$ та точок, що отримано експериментально) в одній системі координат.

```
\begin{tikzpicture}[scale=0.8]
\begin{axis}[xmin=-1.4,xmax=4,
xlabel=$x$,
ylabel={$f(x)$},
height=10cm,width=10cm, legend pos={north west},
grid = major,title=Графік за функцією]
% побудова графіка першої функції
\addplot +[domain=-1:3.5,draw=blue,samples=4,only
marks] coordinates {
( 0, -5 )
( 1, -7 )
( 2, -7 )
( 3, 1 )};
% легенда до першого графіка
\addlegendentry{Вузли інтерполяції}
% побудова графіка другої функції
\addplot+[domain=-1:3.5,thick,mark=none,draw=red]
{\a*x^3 +\b*x^2 + \c*x + \d};
% легенда до другого графіка
%\addlegendentry{Апроксимаційний поліном}
\end{axis}
\end{tikzpicture}
```

Результат компіляції:

Графік за функцією



Для побудови використано команду:

```
\addplot +[domain=-1:3.5,draw=blue,mark size=5pt,samples=4,only
marks] coordinates {
( 0, -5 )
( 1, -7 )
( 2, -7 )
( 3, 1 )};
```

тут: **domain=-1:3.5** — діапазон побудови графіка цієї функції;
samples=4 — кількість точок, за якими будується графік;
draw=blue — точки мають блакитний колір;
mark size=5pt — розмір точки складає 5pt.

Іноді по якійсь осі необхідно відобразити більше знаків, ніж у стандартному відображенні TEX. Для цього в параметрах оточення **axis** необхідно задати **yticklabel style={/pgf/number format/fixed, /pgf/number format/precision=4,}**

4.1.4 *Побудова графіка функції, яку задано параметрично*

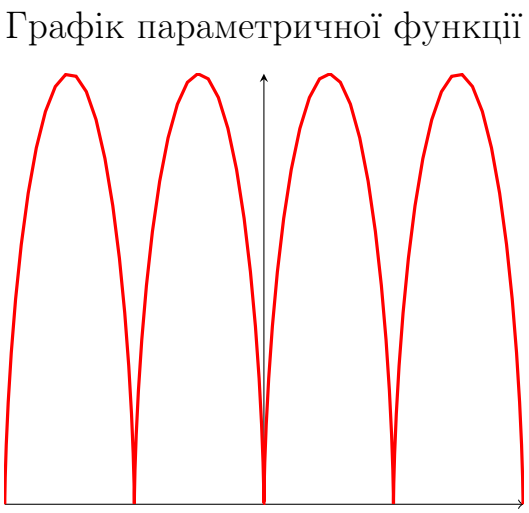
У цьому прикладі будується графік циклоїди, рівняння якої задано параметрично:

$$x = t - \sin t$$
$$y = 1 - \cos t$$

```
\begin{tikzpicture}
\begin{axis}[axis equal, axis lines=middle,
```

```
axis line style={->},
tick style={color=black},
xtick=\empty, ytick=\empty ,title=Графік параметричної функції]

\addplot+[mark=none,red, very thick,
samples=100, domain=-4*pi:4*pi]
({x-sin(deg(x))}, {1-cos(deg(x))});
\end{axis}
\end{tikzpicture}
```

Результат компіляції:

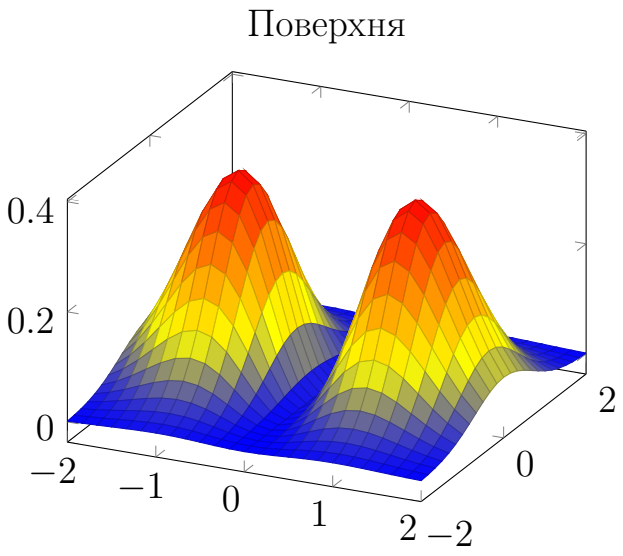


4.1.5 *Побудова поверхні*

Для побудови тривимірної поверхні використовують функцію `\addplot3`

```
\begin{tikzpicture}
\begin{axis}[title=Поверхня]
\addplot3[surf][domain=-2:2]
{x^2*exp(-x^2-y^2)};
\end{axis}
\end{tikzpicture}
```

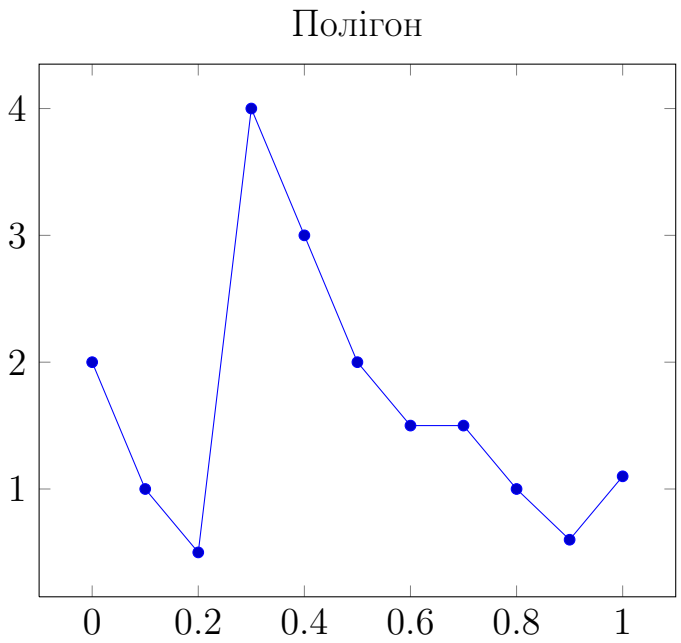
Результат компіляції:

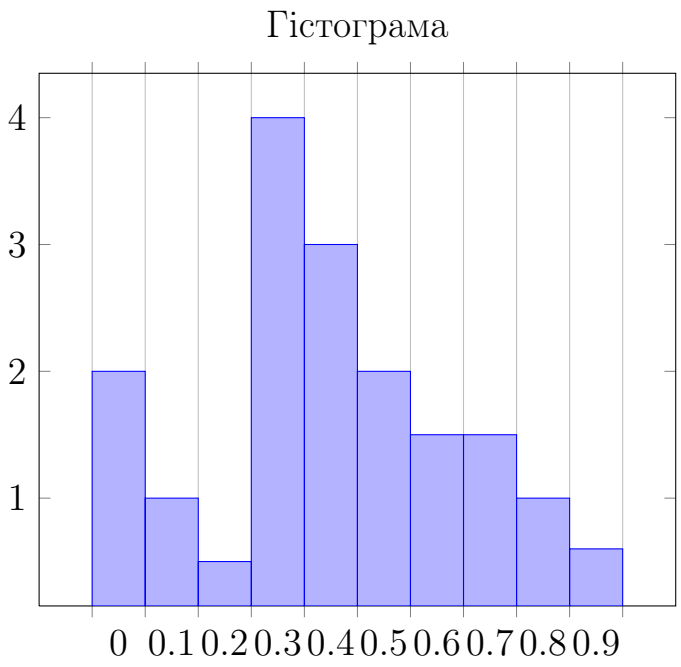


4.1.6 *Побудова полігону та гістограми*

```
 \begin{tikzpicture}
\begin{axis}[
%xtick=data,
%xticklabel interval boundaries, x tick label
style= {anchor=north}, width=10cm,
title=Полігон]
\addplot
coordinates {(0,2) (0.1,1) (0.2,0.5) (0.3,4) (0.4,3) (0.5,2)
(0.6,1.5) (0.7,1.5) (0.8,1) (0.9, 0.6) (1,1.1)};
\end{axis}
\end{tikzpicture}

\begin{tikzpicture}
\begin{axis}[ybar interval,
%xtick=data,
%xticklabel interval boundaries, x tick label
style= {anchor=north}, width=10cm,title=Гістограма]
\addplot
coordinates {(0,2) (0.1,1) (0.2,0.5) (0.3,4) (0.4,3) (0.5,2)
(0.6,1.5) (0.7,1.5) (0.8,1) (0.9, 0.6) (1,1.1)};
\end{axis}
\end{tikzpicture}
```

Результат компіляції:



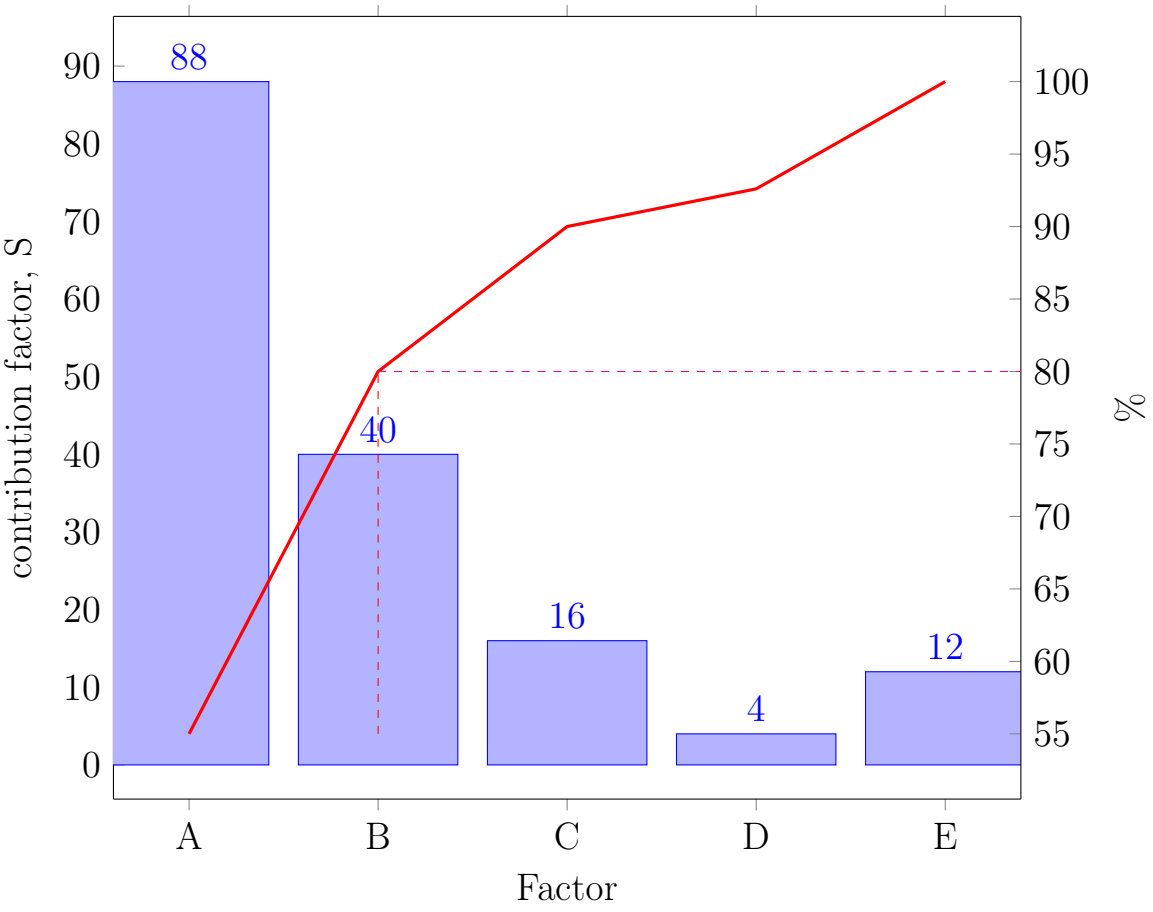


Спорідненою з полігоном є діаграма Парето, яка є одним з семи інфтрументів якості.

Для побудови діаграми Парето можна використати такий код:

```
\begin{tikzpicture}
  \begin{axis}[scale only axis,  axis y line*=left,
    ybar, bar width=60pt, width=12cm,
enlarge x limits=0.1,
    xlabel={Factor},
ylabel={contribution factor, S},
    symbolic x coords={A,B,C,D,E},
    xtick=data,
    nodes near coords,  %nodes near coords align={horizontal},
```

```
    ]
    \addplot coordinates {(A,88) (B,40) (C,16) (D,4) (E,12)};
  \end{axis}

\begin{axis}[scale only axis,  axis y line*=right,
width=12cm,
axis x line=none, ylabel=\%,
symbolic x coords={A,B,C,D,E}]
\addplot[red, very thick]
coordinates {(A,55) (B,80) (C,90) (D,92.6) (E,100) };
\draw[magenta, dashed] (100,250) -- (500,250) ;
\draw[red, dashed] (100,0) -- (100,250) ;
\end{axis}
\end{tikzpicture}
```



4.2 Інтелектуалні мапи

Інтелектуалні мапи — це візуалізація думок. Вони допомагають вчитися, генерувати нові ідеї та запам'ятовувати великий обсяг інформації.

Часто зустрічаються й інші синонімічні переклади терміну **mindmap**:

- мапи асоціацій;

- мапи розуму;
- мапи думок;
- мапи пам'яті;
- ментальні мапи;
- діаграма зв'язків
- асоціатива мапа.

Всі ці терміни обозначають спосіб фіксації процесу мислення, який найбільш сожий на те, як зароджуються й еволюціонують думки та ідеї у нашому мозку та відображають основну ідею техніки — побудову «дерева асоціацій», у якому:

- в центрі розташовано базовий образ, від якого відходять кілька гілок — вирішення, безпосередньо пов'язані з базовим образом;
- від інших може відходити нескінченна кількість малих гілочок, які будуть відносно першообразу вторинними, третинними і т. д.

Метод mind map — техніка візуалізації, яка допомагає вивчити нову інформацію чи вирішити певну задачу, це інструмент візуального відображення інформації, що дозволяє ефективно структурувати й обробляти її.

Інтелектуальна мапа є деревоподібною схемою, на якій зображені слова, ідеї, задачи чи інші поняття, що пов'язані гілками, які відгалужуються від центрального поняття чи ідеї. Гілки позначають і пояснюють ключовими словами чи образами.

Цей інструмент сьогодні активно використовують і рекомендують експерти з розвитку різних напрямків. Він відноситься до розряду нейро-візуального планування.

За концептуальну базу взято структуру побудови нейроних мереж мозку, її досконала ефективність у галузі роботи з інформацією. Коли передача даних прямує від центрального вузла далі по мережі побудованих зв'язків, і кожна порція сигналу надходить до потрібного вузла, для кінцевої реакції. Сьогодні цей принцип активно використовують в галузях підвищення ефективності будь-яких дій, у будь-якій царині, від керування бізнесом, побудови роботи колективу, генерації ідей, планування подій, до організації хатніх справ, відпочинку чи навчання.

Інтелектуальні мапи — спосіб зображення процесу загального системного мислення за допомогою схем. Також може розглядатися як зручна техніка альтернативного запису. Інтелектуальні мапи використовують для створення, візуалізації, структурування і класифікації ідей, а також як засіб для навчання, організації, вирішення завдань, ухвалення рішень, під час написання статей.

Інтелектуальна мапа — це інструмент візуального мислення, який має багато переваг у порівнянні з традиційними техніками ведення нотаток.

Інтелектуальні мапи

- Застосовують для демонстрації набору ідей, які, як правило, надходять з середини.
- Використовують для організації широкого спектру задач і концепцій, а тому мають різноманітні галузі застосування.
- Містять основне слово, фразу чи зображення в центрі схемы, звідки в усі боки відгалужуються пов'язані ідеї.
- Для пояснення теми використовують одну батьківську гілку і кілька дочірніх.

4.2.1 Як використовувати інтелектуальні мапи

Якщо ви задаєтесь запитанням, навіщо складати інтелектуальні мапи, подивіться на цей список головних переваг майндмепінгу:

Інформація про структуру інтелектуальної мапи

Інтелектуальні мапи є графічним поданням інформації, яке передає відношення між окремими ідеями і концепціями. Незалежно від того, наскільки складним чи обширним є предмет, інтелектуальна мапа привносить порядок у хаос і допомогає Вам побачити повну картину.

Інтелектуальні мапи покращують пам'ять

Інтелектуальні мапи використвують такі збудники пам'яті, як кольори і зображення. Ваш мозок запам'ятовує їх легше, ніж простий текст. Крім того, інтелектуальні мапи допомогають Вам пов'язати нові елементи інформації з існуючими знаннями.

Інтелектуальні мапи розвивають креативність

Існує дуже мало технік, які так сприяють розвитку Ваших творчіх здібностей, як майндмепінг. Інтелектуальні мапи використовують зображення і ключові слова для створення нових асоціацій у Вашому мозку, які Ви можете розуміти з надзвичайною швидкістю.

Інтелектуальні мапи полешують співробітництво

Якщо традиційні паперові інтелектуальні мапи добре підходять для самостійної розробки ідей, то онлайн інтелектуальні мапи дозволяють здійснювати мозковий штурм з іншими людьми і сумісно планувати в режимі реального часу, неважливо, де ви знаходитесь чи який пристрій використовуєте.

4.2.2 Кому потрібні інтелектуальні мапи?

Творчі особистості

Дизайнери, письменники і маркетологи можуть використовувати інтелектуальні мапи для мозкового штурму і візуалізації ідей.

Менеджери

Менеджери можуть використовувати інтелектуальні мапи для створення стратегій, планів проєктів, успішних нарад тощо.

Консультанти

Консультанти можуть використовувати інтелектуальні мапи для успішного спілкування з клієнтами і для показу презентацій.

Вчителі

Вчителі можуть використовувати інтелектуальні мапи і шаблони для розвитку критиченого мислення і співробітництва в класі.

Студенты

Студенти можуть використовувати інтелектуальні мапи для створення більш ефективних нотаток та поліпшення пам'яті і розуміння.

Всім

Кожен може використовувати інтелектуальні мапи для розкриття свого творчого потенціалу, вирішення проблем і візуалізації цілей!

Мозговой штурм

Майндмепінг сприяє вільному потоку ідей і наштовхує на нові думки через асоціації. Це ідеальний інструмент для креативного мозкового штурму наодинці чи в команді.

Створення нотаток

На конференції чи в лекційній залі — створювати нотатки в інтелектуальній мапі ефективніше і потім їх легко можна переглянути і запам'ятати.

Планування

Ви можете використовувати інтелектуальні мапи, щоб створювати плани проєктів, розробляти бізнес-стратегії, планувати заходи і багато іншого. Планування з інтелектуальними мапами є простим, візуальним і навіть не відчувається як робота!

Керування нарадою

Готуйте і розповсюджуйте інформацію для зустрічей, створюйте швидкі презентації, протоколи зустрічей і призначайте задачі — і все в рамках однієї інтелектуальної мапи.

Керування ідеями

За допомогою інтелектуальної мапи Ви легко генеруєте, захоплюєте і керуєте ідеями. Збирайте думки учасників, голосуйте і коментируйте теми, а також розставляйте пріоритети — і це все в одній мапі.

Керування знаннями

Інтелектуальні мапи дозволяють організаціям зберігати, обмінюватися і керувати своїми колективними знаннями централізовано і надійно. Завдяки їх інтуїтивно зрозумілому формату, пошук і оновлення інформації не викличе ускладнень.

4.2.3 Як створити інтелектуальну мапу самостійно?

Інтелектуальні мапи будуть ефективними й дадуть відповідний результат, якщо створювати їх за певними рекомендаціями. Для створення інтелектуальної мапи знадобиться аркуш паперу, кольорові фломастери чи олівці та трішки натхнення. Якщо малювання не твоя сильна сторона — не переймайся, адже основна мета інтелектуальної мапи зовсім в іншому.

1. Визначся з основною темою інтелектуальної мапи. Вигадай до неї візуальний образ і розташуй її в центрі аркуша. Не записуй тему словами і не використовуй цифр — звільни свій мозок від шаблонів та ввімкни творчість.

2. Намалюй кілька гілок, які беруть початок з центру. Це можуть бути назви розділів прочитаної книжки чи кілька варіантів стратегії розвитку компанії, які пов'язані спільною темою — центром.

3. Від основних гілок промальовуй додаткові гілки, які позначатимуть менш значущі речі. На цьому етапі вже краще зменшити розмір зображень, щоб спершу в очі впадала основна інформація.

4. Застосовуй різні кольори, шрифти та малюй символічні зображення, які допоможуть тобі краще запам'ятати інформацію та передусім фокусуватися на важливих темах.

5. Не розписуй поняття — використовуй символи. Намагайся використовувати зображення, які викликатимуть у тебе асоціації. Це може наштовхнути на нові ідеї чи думки.

Етапи побудови інтелектуальної мапи можуть різнитися залежно від цілей та завдань уроку, заняття, однак загальний алгоритм побудови такий:

Розпочніть з центру: Щоб створити інтелектуальну мапу, розпочніть з визначення предмету Вашої мапи в центрі чистого аркушу пареру чи цифрового холста. Предметом може бути одне ключове слово, зображення, запитання чи проблема, яку Ви намагаєтесь вирішити.

Додайте гілки: Гілки — це лінії, які проводять від центру і символізують основні категорії чи ключові ідеї, що мають відношення до Вашого предмету. Більшість мап має від чотирьох до шести гілок, але Ви можете додати стільки, скільки Вам потрібно.

Напишить ключове слово в кожній гілці: Використавуйте ключові слова і короткі фрази, а не довгі речення. Це зробить Вашу мапу компактною і легкою для огляду, і Ви зможете побачити Ваші найважливіші ідеї з першого погляду.

Додайте ідеї: Кожна мапа, яку Ви створюєте, може викликати асоціації у Вашому мозку і пробуджувати новіе ідеї, які потім можна буде додати як дочірні гілки. На кількість дочірніх гілок чи ієрархічни рівнів, які Ви можете створити, обмерень немає.

Додайте візуальні елементи: Збагатіть мапу, додаши кольри, іконки та зображення в окремі теми.

Етапи побудови інтелектуальної мапи

	Етапи	Особливості
1.	Обмірковування ідеї	• здійснюють мозковий штурм: озвучують будь-які слова, ідеї, образи, асоціації, що спадають на думку • обирають базовий центральний образ (об'єкт вивчення)
2.	Деталізація змісту інтелектуальної мапи	• будують центральний образ • обмірковують ключові слова • будують головні гілки • записують ключові слова (по одному слову на кожній гілці)
3.	Доповнення мапи новими ідеями	• будують розширені гілки від головних гілок • там, де це можливо, додають: – іконки – малюнки – значки – символи • прослідковують зв'язки між окремими образами, асоціаціями • активно застосовують різні кольори, шрифти, розміри, товщину слів і гілок
4.	Редагування	• застосовують мозковий штурм для з'ясування невикористаних асоціацій • записують нововиявлені асоціації • інтелектуальну мапу переглядають, щоб виявити можливі: – повтори – описки – помилки – дублювання гілок • перевіряють можливість відтворення візуальних образів, що зображені на інтелектуальній мапі

4.2.4 TikZ бібліотека інтелектуальної мапи

Для того щоб намалювати інтелектуальну мапу ми спочатку створюємо середовище для малювання. Ми завантажуємо пакет **TikZ** та бібліотеку **mindmap** і входимо у середовище **tikzpicture**, використовуючи стиль **mindmap**. Загальний вигляд мінімального коду у цьому випадку має бути таким:

```
\documentclass{standalone}

% Required package
\usepackage{tikz}
\usetikzlibrary{mindmap}  % Mindmap drawing library

\begin{document}
\begin{tikzpicture}[mindmap]
% We will draw the mind map here
\end{tikzpicture}
\end{document}
```

4.2.5 Концептуальний вузол і концептуальний кольор

Інтелектуальна мапх має характеристики діаграми у вигляді дерева; вона має кореневий вузол та дочірні вузли (вузли-нащадки), які з'єднані з їхніми батьками ієрархічно. Основна відмінність між ними полягає в тому, що в інтелектуальних мапах корінь зазвичай знаходиться в центрі сторінки, а його дочірні вузли розкидані у всіх напрямках, а не спрямовані в одному напрямку, як це здійснено у діаграмі у вигляді дерева. У **TikZ** інтелектуальні мапи форматують поверх діаграм у вигляді дерева. Щоб створити свою інтелектуальну мапу, ми маємо розпочинати зі створення нашого кореневого вузла. Кожна з наших ідей буде відгалуженням від цього кореня.

Щоб створити вузол, ми просто вводимо
\node [concept] {наша коренева концепція} ;.
Ми використовуємо стиль концепції, щоб вказати, що ми малюємо вузли для нашої концепції концептуальної мапи, аби ми могли вносити зміни в стиль, який впливає на всі вузли. Однією з таких стилевих опцій є концептуальний колір. Наприклад, давайте розпочнемо з блакитного відтінку.

```
\begin{tikzpicture}[mindmap, concept color = blue!30]
\node [concept] {Наша основна концепція };
\end{tikzpicture}
```

Результат компіляції

4.2.6 Стилізація дочірніх вузлів

Наступне, що потрібно зробити, це додати до нашого кореневого вузла кілька дочірніх вузлів для розгалуження. Ми можемо легко це зробити, використовуючи ключове слово у дочірньому вузлі. У кожному дочірньому вузлі ми створимо ще один вузол рекурсивним чином. Синтаксис для дочірніх вузлів не міститиме бекслешів:

child {node {node content}}

Ці дочірні вузли також мають бути елементами, на які впливає стиль концепції. Щоб уникнути повторення, ми можемо вказати середовищу змінити кожен вузол, щоб зробити їх концептуальними вузлами, додавши команду
every node/.style = concept
в параметри нашої **tikzpicture**. Тоді ми можемо прибрати параметр стилю в нашому кореневому вузлі, оскільки він нам більше не потрібен.

```
\begin{tikzpicture}[mindmap, concept color = blue!30,
every node/.style = {concept}]
\node {Наша основна концепція}
child {node {Перша ідея}};
\end{tikzpicture}
```

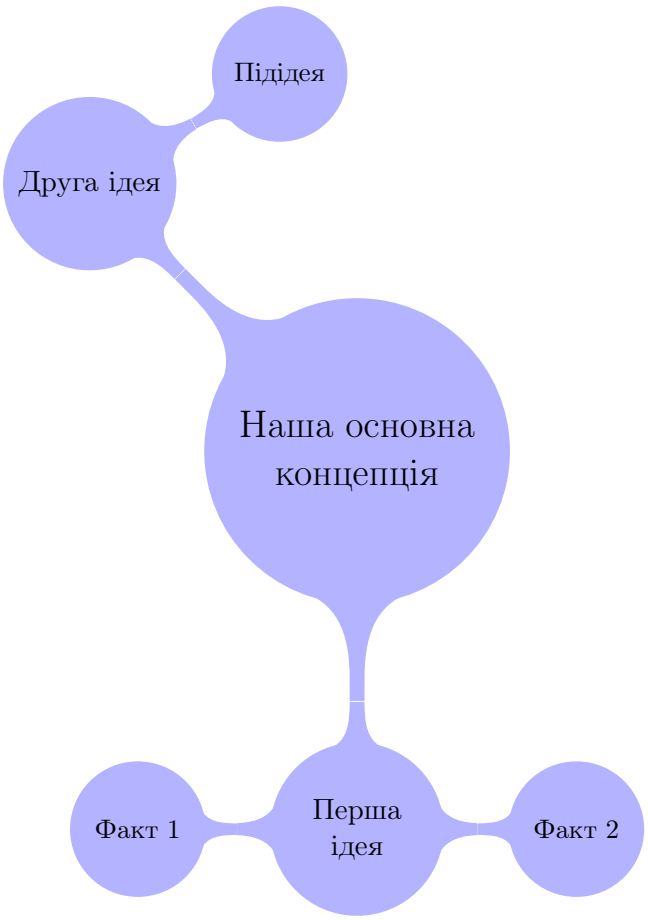
Результат компіляції

4.2.7 Розташування дочірніх вузлів за допомогою опції Grow

Коли ми додаємо нові дочірні вузли до нашої інтелектуальної мапи, нам варто буде подумати про їхнє розташування. Ми можемо визначити їхню конкретну позицію, використовуючи опцію **Grow**. Таким чином, ми матимемо свободу розташовувати їх як нам зручно. Ми можемо використовувати набір ключових слів (**up, down, left, right, north, south, east, west, north east тощо**) або кути з опцією Grow.

```
\begin{tikzpicture}[mindmap, concept color = blue!30,
every node/.style = {concept}]
\node  {Наша основна концепція}
child [grow = down]
{node {Перша ідея}
child [grow = left] {node {Факт 1}}
child [grow = right] {node {Факт 2}}}
child [grow = north west]
{node {Друга ідея}
child [grow = 30]
{node {Підідея}}};
\end{tikzpicture}
```

Результат компіляції



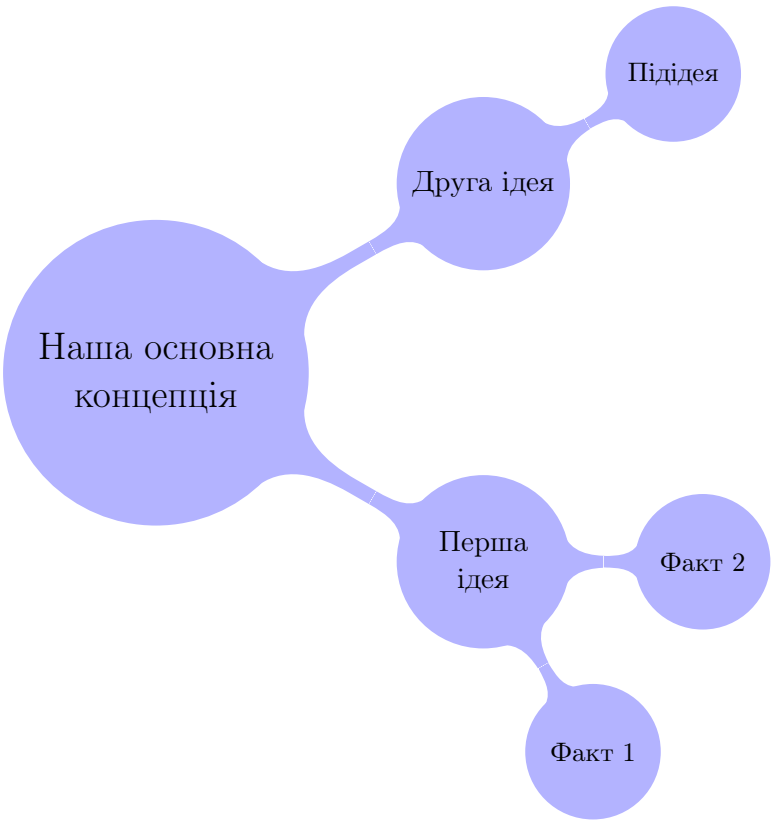
4.2.8 Циклічний варіант опції Grow

Незважаючи на те, що ми маємо опцію (**Grow**), ми можемо впоратися з розташуванням набагато легше, використовуючи **grow cyclic** в нашому TikZ середовищі. Ця опція дозволить розташувати дерево по колу і залишити нашу кореневу концепцію в центрі.

Нам потрібно видалити інші опції **Grow**, щоб побачити ці ефекти.

```
\begin{tikzpicture}[mindmap, concept color = blue!30,
every node/.style = {concept}, grow cyclic]
\node  {Наша основна концепція}
child {node {Перша ідея}
child {node {Факт 1}}
child {node {Факт 2}}}
child {node {Друга ідея}
child {node {Підідея}}};
\end{tikzpicture}
```

Результат компіляції



4.2.9 Стилізація різних рівнів схеми інтелектуальної мапи

Далі ми розглянемо опції стилів різних рівнів наших вузлів. TikZ бібліотека **mindmap** інтелектуальних мап надає нам опції стилів п'яти різних рівнів нашої схеми інтелектуальної мапи:

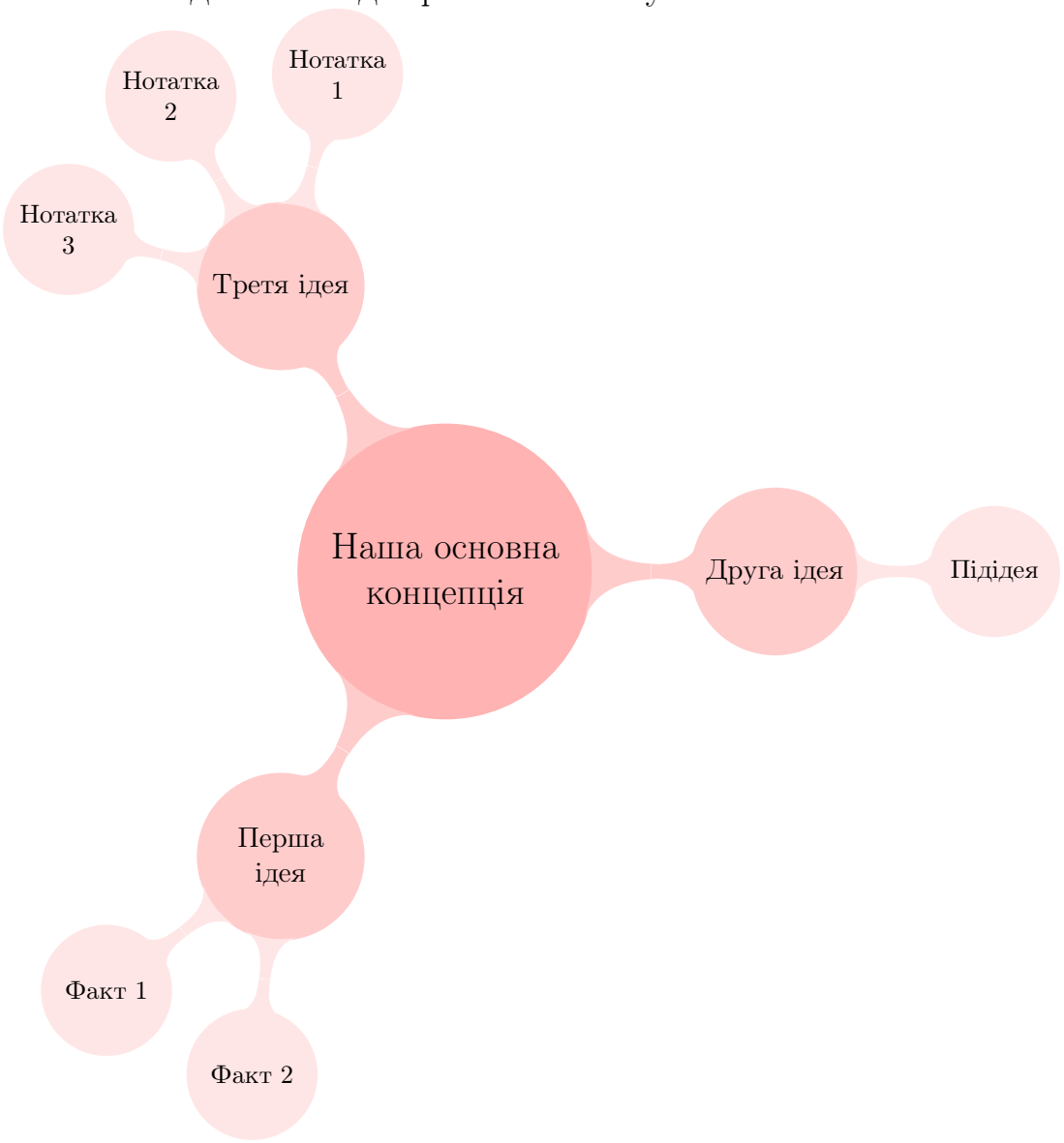
root concept, level 1, level 2, level 3 and level 4.

Немає варіантів для рівня 5+. Вузли пешого рівня (level 1) є дочірніми щодо нашої інтелектуальної мапи, другого рівня (level 1) є дочірніми щодо першого рівня і так далі. Ми можемо змінити параметри стилю, такі як колір концепції, відстань між рівнями або кутову відстань для кожного дочірнього вузла, використовуючи стиль додавання. Нижче наведений код показує різні варіанти на практиці з кількома додатковими дочірніми вузлами.

```
\begin{tikzpicture}[mindmap, concept color = red!30,
every node/.style = {concept}, grow cyclic,
level 1/.append style = {concept color = red!20,
level distance = 4.5cm, sibling angle = 120},
level 2/.append style = {concept color = red!10,
level distance = 3cm,  sibling angle = 45}]
\node  {Наша основна концепція}
child {node {Перша ідея}
child {node {Факт 1}}
```

```
child {node {Факт 2}}
child {node {Друга ідея}
child {node {Підідея}}}
child {node {Третя ідея}
child {node {Нотатка 1}}
child {node {Нотатка 2}}
child {node {Нотатка 3}}};
\end{tikzpicture}
```

З чого виходить така діаграма інтелектуальної мапи



У нашому прикладі ми змогли вибрати світліший колірний тон для кожного рівня. Ще одна зміна була реалізована за допомогою команди визначення відстані між рівнями. Відстань між рівнями є відстанню між центром батьківського вузла та центром дочірнього вузла.

У нашому випадку відстань до першого рівня складає 4,5 см, а відстань до другого рівня — 3 см. Ми також визначаємо кутову відстань між дочірніми вузлами на рівні; для вузлів першого рівня — 120 градусів, а для вузлів другого

рівня — 45 градусів. Для першого рівня існує емпіричне правило — цей кут між дочірніми вузлами на рівні складає 360 градусів; цей параметр розташовував би кожен вузол по колу способом навкруг кореневого вузла.

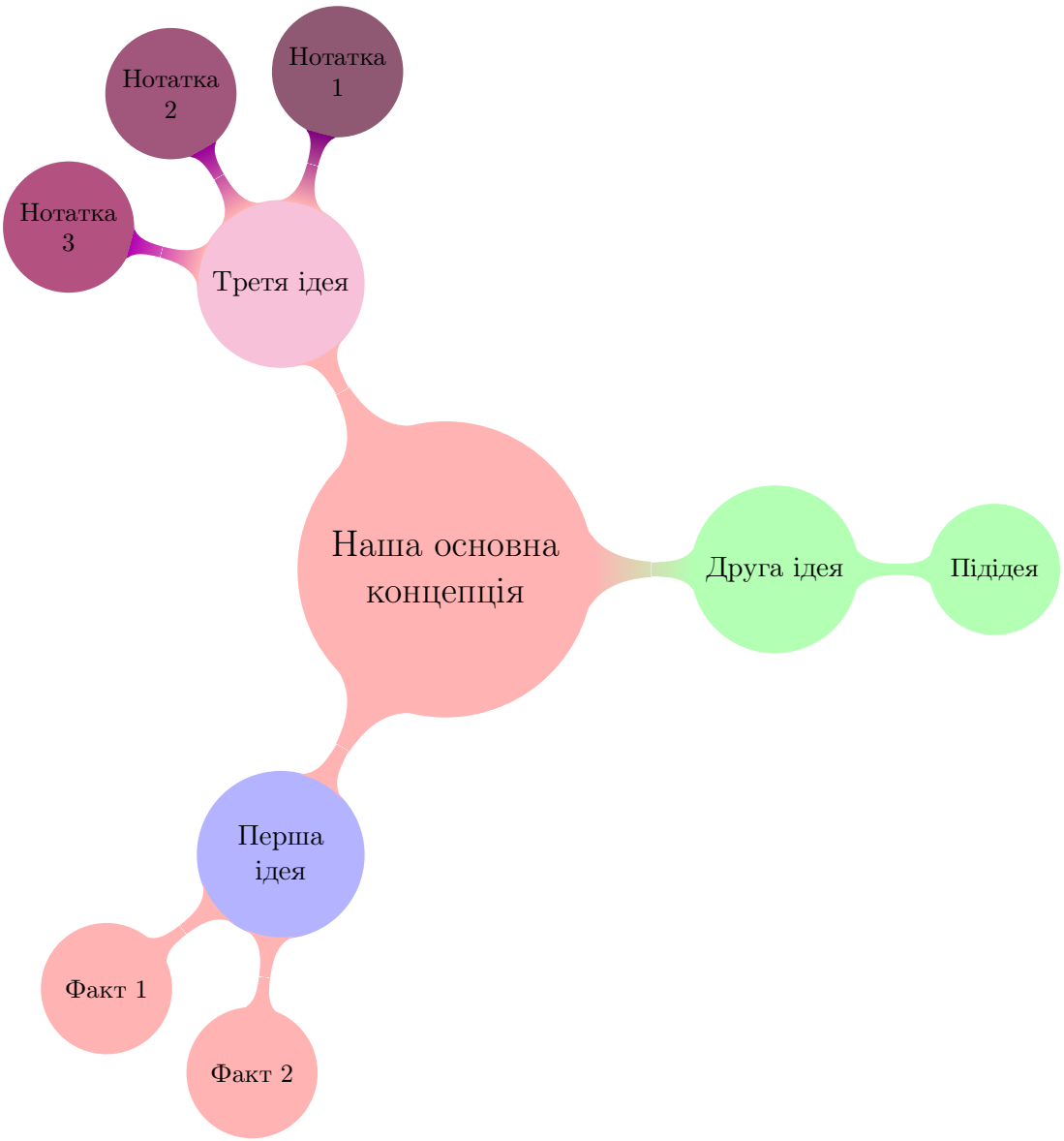
4.2.10 Кольорування вузлів по-різному

Інший спосіб розфарбувати вузли — надати кожній ідеї першого рівня власний колір. Для досягнення цього ми можемо додати різні варіанти кольорової концепції до кожного з наших вузлів першого рівня замість того, щоб встановлювати загальне правило для всіх вузлів рівня. Наш код, що наведено нижче, випробовує різні варіанти забарвлення дочірніх вузлів. Зверніть увагу, що ми прибрали частину концепції кольору зі стилів першого і другого рівня.

Нагадаємо, що пакет **xcolor** має 19 кольорів: red , green , blue , cyan , magenta , yellow , black , gray , darkgray , lightgray , brown , lime , olive , orange , pink , purple , teal , violet and white .

```
\begin{tikzpicture}[mindmap, concept color = red!30,
every node/.style = {concept}, grow cyclic,
level 1/.append style = {level distance = 4.5cm,
sibling angle = 120},
level 2/.append style = {level distance = 3cm,
sibling angle = 45}]
\node {Наша основна концепція}
child {node [concept color = blue!30] {Перша ідея}
child {node {Факт 1}}
child {node {Факт 2}}}
child [concept color = green!30] {node {Друга ідея}
child {node {Підідея}}}
child {node [concept color = magenta!30] {Третя ідея}
child [concept color = magenta!50!black] {node {Нотатка 1}}
child [concept color = magenta!60!black] {node {Нотатка 2}}
child [concept color = magenta!70!black] {node {Нотатка 3}}};
\end{tikzpicture}
```

Результат компіляції



У нашому першому дочірньому вузлі ми змінили стиль вузла всередині нашої дочірньої команди, яка змінила лише рівень 1 і не змінила рівень 2. У нашому другому дочірньому вузлі ми змінили стиль дочірньої команди, яка також змінила всі її дочірні вузли. Нарешті, у нашому третьому дочірньому вузлі ми визначаємо у кожному з вузлів різний колір. Ви можете змінювати колір кожного вузла, використовуючи ці опції. Зверніть увагу, що коли ми змінили колір дочірньої команди, існує перехід між кольорами, якого немає, коли ми змінили колір лише вузла або всього рівня.

4.2.11 Додавання зв'язків до діаграми інтелектуальної мапи в LaTeX

Іноді вузли в інтелектуальній мапі, які належать різним батькам, можуть мати зв'язок між собою. Ми маємо опцію для подання цих зв'язків використовуючи TikZ. Найпростіший спосіб з'єднати два вузли — це провести між ними лінію. Бібліотека **mindmap** має опції для з'єднання концепцій для обробки таких типів з'єднань.

Щоб підключити наші вузли, спочатку ми маємо дати їм імена, додавши їхні імена в дужках між стилем вузла та вмістом, а потім викликати функцію малювання з цими іменами, використовуючи опцію підключення концепції. Щоб імена були простими та короткими, я назвав кореневу концепцію **root** та використав числа для кожного рівня та вузла. Наприклад, назва «Перша ідея» — 1, «Факт 1» — 11, «Факт 2» — 12.

Потім ми викличемо нашу функцію малювання. Припустимо, існує зв'язок між Фактом 1 першої ідеї та Нотаткою 3 третьої ідеї. Лінія міститьметься в порожньому просторі і не перекриватиметься жодним вузлом. Однак, якщо ми спробуємо зв'язати Факт 2 Першої ідеї з Нотаткою 1 Третьої ідеї, наша лінія перетне вузли Першої ідеї та Третьої ідеї та ускладнить їх читання. Щоб вирішити цю проблему, ми можемо намалювати наші лінії на фоновому рівні.

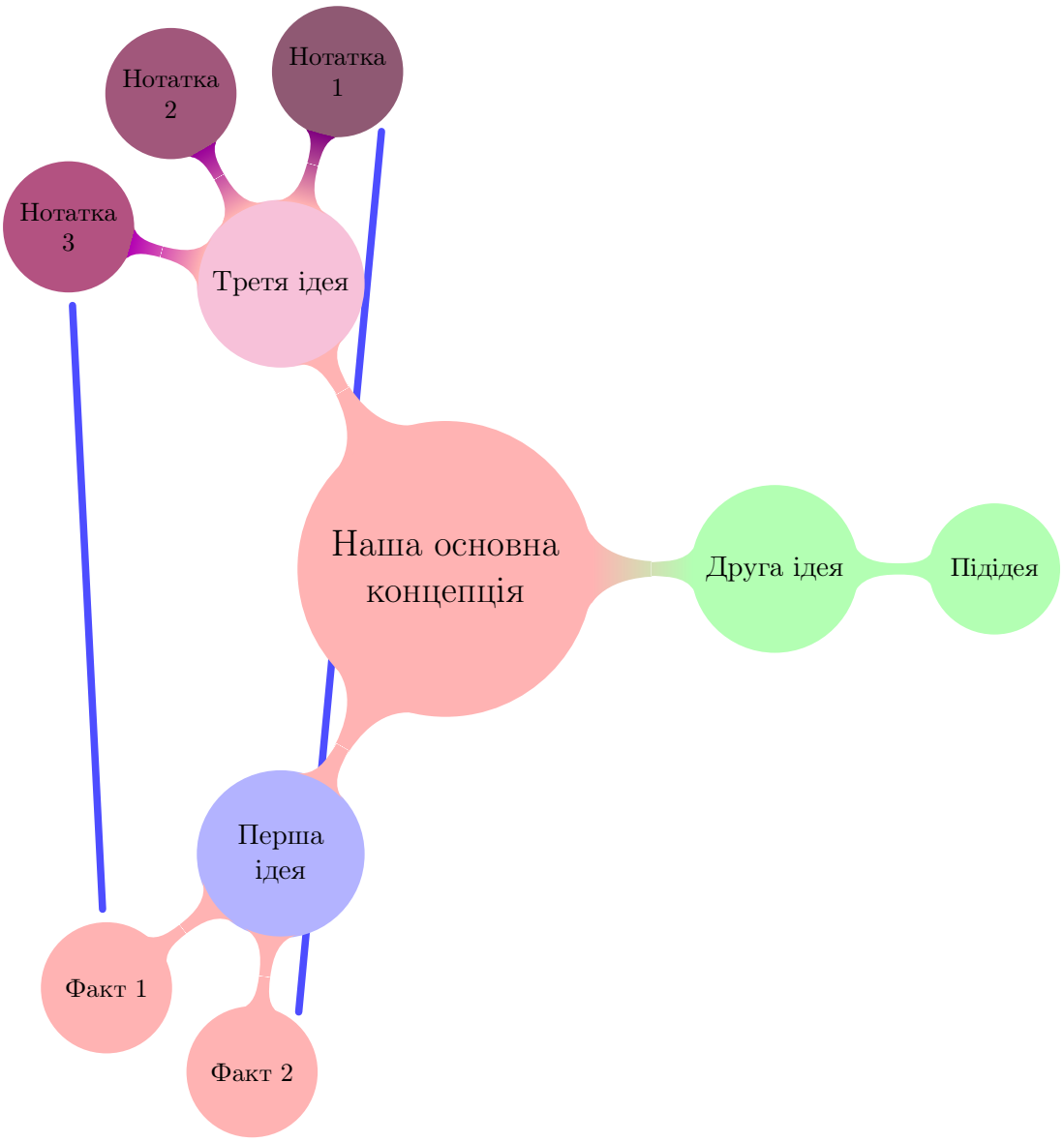
4.2.12 Фоновий рівень (тло) в TikZ

Щоб використовувати тло, нам потрібно додати бібліотеку **backgrounds**, яка дозволить скористатися командою **pgfonlayer**. Ми можемо зробити зв'язок більш помітним, якщо провести лінію з **north east** Факту 2 до **south east** Нотатки 1. У наступному коді всі вузли поіменовані, а зв'язки між вузлами намальовані на фоновому рівні. Крім того, ми додали опцію стилю, щоб зробити наші лінії зв'язку блакитними.

```
\documentclass{standalone} % Required package
\usepackage[cp1251]{inputenc} % Друкування вихідного тексту.
\usepackage[ukrainian]{babel} % Підтримка української мови.
\usepackage{graphicx}
\usepackage{xcolor}
\usepackage{indentfirst} % Відступ у першому
\frenchspacing % відстань між окремими реченнями така ж як і
%                 між словами в реченні
\usepackage{pdfpages}
\usepackage{tikz} % Підключення пакету tikz
\usetikzlibrary{mindmap, backgrounds}
% Визначення необхідних бібліотек з пакету tikz
\usetikzlibrary{mindmap, backgrounds}
% Налаштування параметрів сторінки
\usepackage[left=20mm,top=20mm,right=20mm,bottom=20mm]{geometry}
\parindent = 1cm

\begin{document}
\begin{tikzpicture}[mindmap, concept color = red!30,
every node/.style = {concept},    grow cyclic,
```

```
level 1/.append style = {level distance = 4.5cm,
sibling angle = 120},
level 2/.append style = {level distance = 3cm,
sibling angle = 45},
concept connection/.append style = {color = blue!70}]
\node (root) {Наша основна концепція}
child {node [concept color = blue!30] (1) {Перша ідея}
child {node (11) {Факт 1}}
child {node (12) {Факт 2}}}
child [concept color = green!30]
{node (2) {Друга ідея}
child {node (21) {Підідея}}}
child {node [concept color = magenta!30] (3) {Третя ідея}
child [concept color = magenta!50!black]
{node (31) {Нотатка 1}}
child [concept color = magenta!60!black]
{node (32) {Нотатка 2}}
child [concept color = magenta!70!black]
{node (33) {Нотатка 3}}};
\begin{pgfonlayer}{background}
\draw [concept connection]    (11) edge (33)
(12.north east) edge (31.south east);
\end{pgfonlayer}
\end{tikzpicture}
 \end{document}
```

Результат компіляції



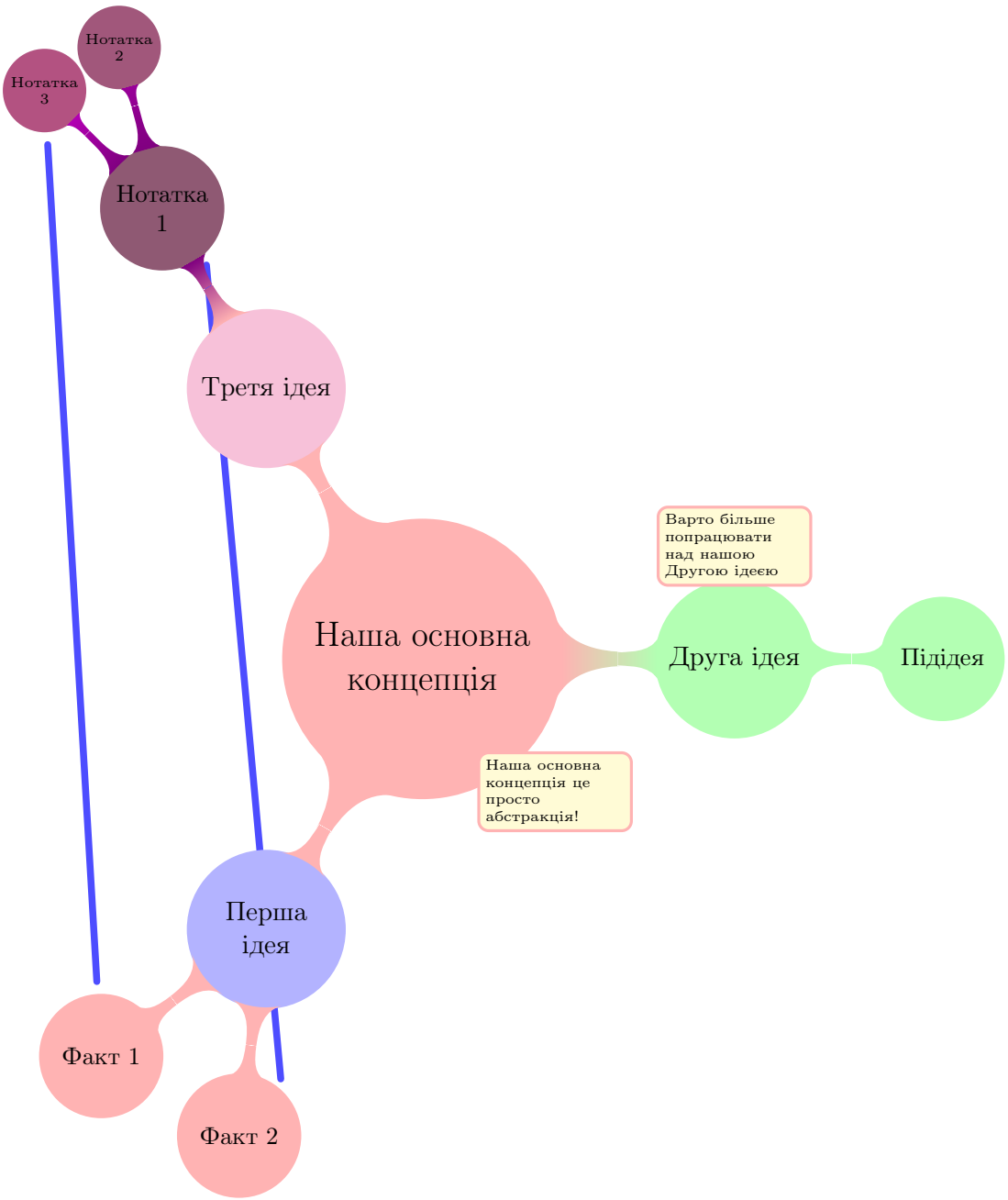
4.2.13 Анотування діаграми інтелектуальної мапи в LaTeX

Бібліотека **mindmap** з TikZ також надає нам можливість вставляти анотації всередину нашої діаграми. Ми можемо їх стилізувати за допомогою ключового слова **annotation** та вставити на діаграму, використовуючи позиції наших вузлів. Використовуючи бібліотеку **calc**, ми можемо вільно розташовувати свої анотації. Наступний код показує два вузли анотацій на схемі.

```
\documentclass{standalone} % Required package
\usepackage[cp1251]{inputenc} % Друкування вихідного тексту.
\usepackage[ukrainian]{babel} % Підтримка української мови.
\usepackage{graphicx}
\usepackage{xcolor}
\usepackage{indentfirst} % Відступ у першому
\frenchspacing % відстань між окремими реченнями така ж
               % як і  між словами в реченні
```

```
\usepackage{pdfpages}
\usepackage{tikz} % Підключення пакету tikz
% Визначення необхідних бібліотек з пакету tikz
\usetikzlibrary{mindmap, backgrounds, calc}
% Налаштування параметрів сторінки
\usepackage[left=20mm,top=20mm,right=20mm,bottom=20mm]{geometry}
\parindent = 1cm
\begin{document}
\begin{tikzpicture}[mindmap, concept color = red!30,
every node/.style = {concept}, grow cyclic,
level 1/.append style = {level distance = 4.5cm,
sibling angle = 120},
level 2/.append style = {level distance = 3cm,
sibling angle = 45},
concept connection/.append style = {color = blue!70},
every annotation/.append style = {fill = yellow!20,
text width = 2cm}]
\node (root) {Наша основна концепція}
child {node [concept color = blue!30] (1) {Перша ідея}
child {node (11) {Факт 1}}
child {node (12) {Факт 2}}}
child [concept color = green!30] {node (2) {Друга ідея}
child {node (21) {Підідея}}}
child {node [concept color = magenta!30] (3) {Третя ідея}
child [concept color = magenta!50!black] {node (31) {Нотатка 1}
child [concept color = magenta!60!black] {node (32) {Нотатка 2}}
child [concept color = magenta!70!black] {node (33) {Нотатка 3}}}};
 \begin{pgfonlayer}{background}
\draw [concept connection] (11) edge (33)
(12.north east) edge (31.south east);
\end{pgfonlayer}
\node[annotation] at ($(2.north) + (0,0.5)$){Варто більше
    попрацювати над нашою Другою ідеєю};
\node[annotation] at ($(root.south east) + (.5,-.5)$){Наша основна
концепція це просто абстракція!};
 \end{tikzpicture}
\end{document}
```

Результат компіляції



Давайте проаналізуємо наші доповнення. В першу чергу ми визначаємо колір та ширину тексту наших вузлів анотацій. Потім ми додаємо вузли, але, вставляючи, ми використали обчислення в системі координат для їх розташування.

($(2.north) + (0,0.5)$) означає додавання 0,5 по осі y до верхнього кінця нашого вузла Другої ідеї.

Наша друга анотації використовує подібне обчислення — ($ (root.south east) + (.5, -. 5) $), це розташоване на 0,5 одиниці праворуч і на 0,5 одиниці вниз від **south east** кінця кореневого вузла **root**. Зверніть увагу, що обчислення відбувається між дужками та знаками долара.

4.2.14 Додавання додаткового ізольованого вузла

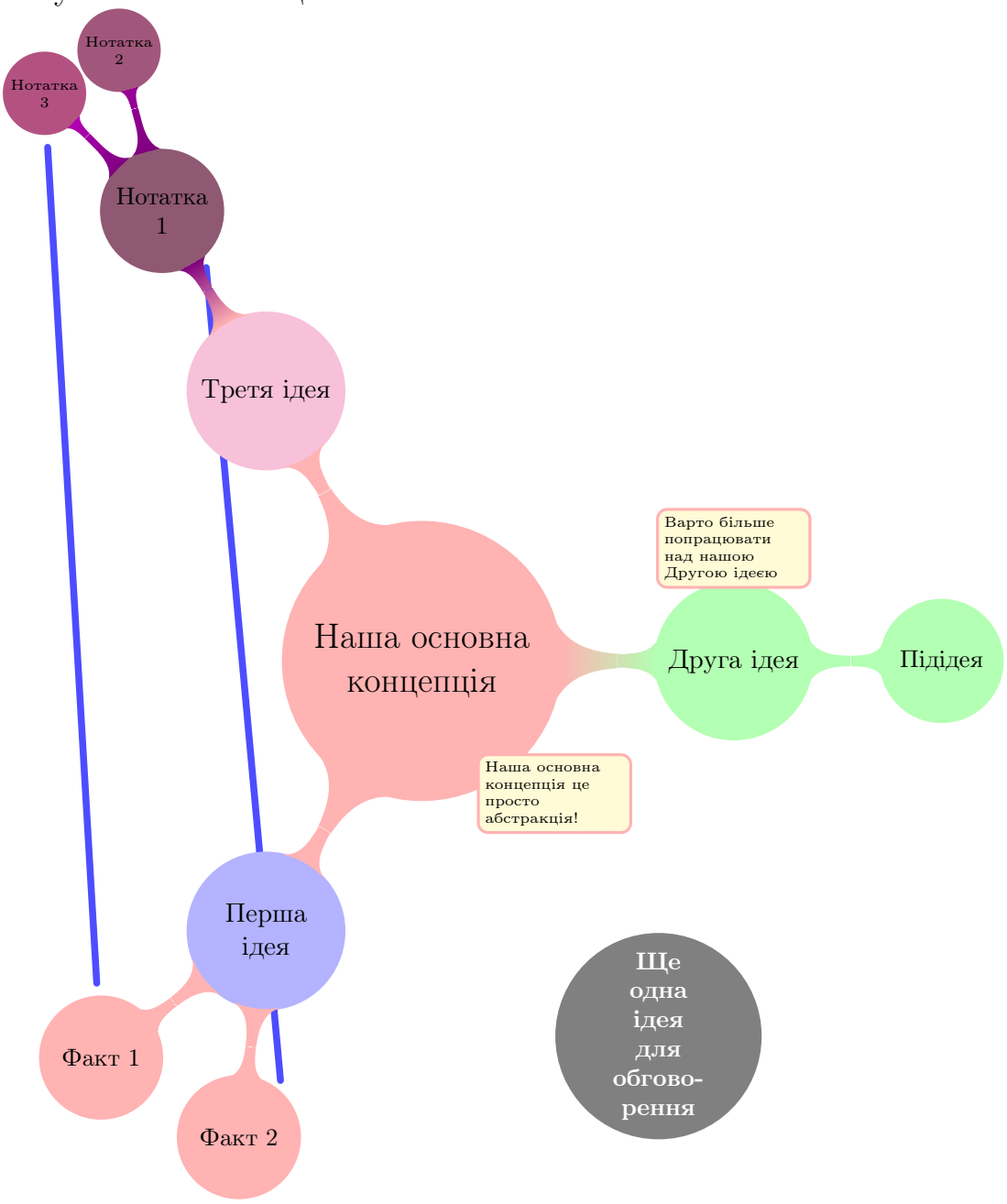
Інший варіант, який нам надає бібліотека **mindmap**, — це створення додаткових концепцій, які не пов'язані з нашими кореневими концепціями. Ми

можемо вставити ці вузли, використовуючи ключове слово **extra concept**, і розмістити їх за допомогою системи відносного позиціонування, яку ми вже обговорювали. Давайте створимо додатковий концептуальний вузол, який має чорне тло і жирний білий текст, розташований на **south east** нашого кореневого вузла.

```
\documentclass{standalone} % Required package
\usepackage[cp1251]{inputenc} % Друкування вихідного тексту.
\usepackage[ukrainian]{babel} % Підтримка української мови.
\usepackage{graphicx}
\usepackage{xcolor}
\usepackage{indentfirst} % Відступ у першому
\frenchspacing % відстань між окремими реченнями така ж
               % як і  між словами в реченні
\usepackage{pdfpages}
\usepackage{tikz} % Підключення пакету tikz
% Визначення необхідних бібліотек з пакету tikz
\usetikzlibrary{mindmap, backgrounds, calc}
% Налаштування параметрів сторінки
\usepackage[left=20mm,top=20mm,right=20mm,bottom=20mm]{geometry}
\parindent = 1cm
\begin{document}
\begin{tikzpicture}[mindmap, concept color = red!30,
every node/.style = {concept}, grow cyclic,
level 1/.append style = {level distance = 4.5cm,
sibling angle = 120},
level 2/.append style = {level distance = 3cm,
sibling angle = 45},
concept connection/.append style = {color = blue!70},
every annotation/.append style = {fill = yellow!20,
text width = 2cm}]
\node (root) {Наша основна концепція}
child {node [concept color = blue!30] (1) {Перша ідея}
child {node (11) {Факт 1}}
child {node (12) {Факт 2}}}
child [concept color = green!30] {node (2) {Друга ідея}
child {node (21) {Підідея}}}
child {node [concept color = magenta!30] (3) {Третя ідея}
child [concept color = magenta!50!black] {node (31) {Нотатка 1}
child [concept color = magenta!60!black] {node (32) {Нотатка 2}}
child [concept color = magenta!70!black] {node (33) {Нотатка 3}}}};
```

```
 \begin{pgfonlayer}{background}
\draw [concept connection] (11) edge (33)
(12.north east) edge (31.south east);
\end{pgfonlayer}
\node[annotation] at ($(2.north) + (0,0.5)$){Варто більше
    попрацювати над нашою Другою ідеєю};
\node[annotation] at ($(root.south east) + (.5,-.5)$){Наша основна
концепція це просто абстракція!};
\node[extra concept] at ($(root.south east) + (2,-4)$) {Ще одна
ідея для обговорення. };
 \end{tikzpicture}
\end{document}
```

Результат компіляції



У вузли крім тексту можна додавати графічні файли.

4.3 Схеми алгоритмів

Часто для пояснення роботи програм чи систем необхідно оформлювати їх у вигляді блок-схем.

Пакет Tikz має багато можливостей зробити це.

Принцип побудови таких схем складається з

- побудови вузлів (початок та кінець, опис дії, блок порівняння);
- побудови напрямку потоку від одного блоку до іншого.

Наведемо два приклади побудови найпростішого алгоритму.

Приклад 1

```
\begin{tikzpicture}[>=latex]
\node[draw,ellipse] (start) at (0,0) {Початок};
\node[draw,rectangle] (do1) at (0,-2) {Дія 1};
\node[draw,diamond] (if) at (0,-4) {Умова};
\node[draw,rectangle] (yes) at (2,-5) {Дія 3};
\node[draw,rectangle] (no) at (-2,-5) {Дія 2};
\node[draw,rectangle] (do4) at (0,-7) {Дія 4};
\node[draw,ellipse] (stop) at (0,-9) {Закінчення};

\draw[->] (start) -- (do1);
\draw[->] (do1) -- (if);
\draw[->] (if) -| (yes);
\draw[->] (if) -| (no);
\draw[->] (yes) |- (0,-6);
\draw[->] (no) |- (0,-6);
\draw[->] (0,-6) -- (do4);
\draw[->] (do4) -- (stop);
\end{tikzpicture}
```

Тут команда **node** визначає вузол певної форми:

ellipse у вигляді еліпсу,

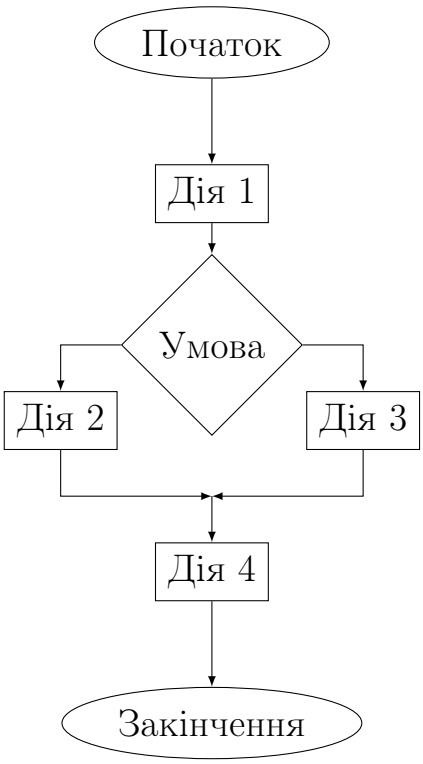
rectangle у вигляді прямокутника,

diamond у вигляді ромбу

з ім'ям (name), який розташовано у точці (x,y), з текстом {text}

Вузли між собою з'єднуються стрілочками за допомогою команди **draw**.

Результат компіляції



Друга можливість вимагає визначення елементів блок-схеми з наступною її побудовою. Це визначення може бути як всередині оточення **tikzpicture**, так і поза ним.

Стилі блоків визначаються командою **tikzstyle{block_shape}**. Тут використано такі види блоків:

блок прийняття рішень decision у вигляді ромба **diamond**;

блок виконання block у вигляді прямокутника **rectangle**;

блок початку/закінчення cloud у вигляді еліпса **ellipse**;

перехід від однієї дії до іншої line у вигляді стрілочки →.

Всі ці блоки мають своє налаштування, яке розташоване в квадратних дужках.

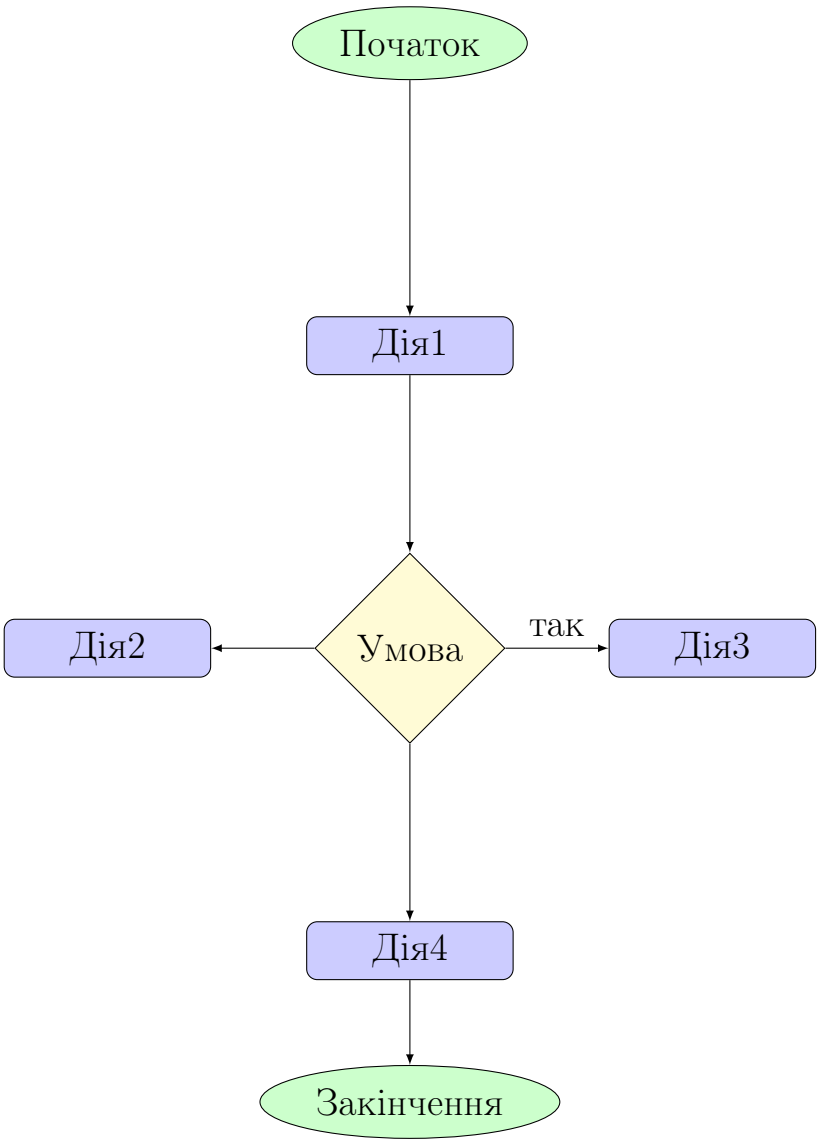
Вузли визначаються командою **node[block_shape] (block_name) {text};**

З'єднання між вузлами визначаються командою
path [line] (block_name_1) - - (block_name_2);

```
 \begin{tikzpicture}[node distance = 3cm, auto,>=latex]

 % Визначення стилів блоків
\tikzstyle{decision} = [diamond, draw, fill=yellow!20,
    text width=4.5em, text badly centered, node distance=4cm,
inner sep=0pt]
\tikzstyle{block} = [rectangle, draw, fill=blue!20,
```

```
    text width=5em, text centered, rounded corners,
node distance=4cm, minimum height=2pt]
\tikzstyle{line} = [draw,->]
\tikzstyle{cloud} = [draw, ellipse,fill=green!20,
node distance=2cm, minimum height=2em]

    % Розташування вузлів
    \node [cloud] (start) {Початок};
\node [block, below of=start] (do1) {Дія1};
     \node [decision, below of=do1] (if) {Умова};
\node [block, right of=if] (do3) {Дія3};
    \node [block, left of=if] (do2) {Дія2};
   \node [block, below of=if] (do4) {Дія4};
\node [cloud, below of=do4] (stop) {Закінчення};

% Малювання з'єднань
     \path [line] (start) -- (do1);
    \path [line] (do1) -- (if);
\path [line] (if) --  (do2);
  \path [line] (if) -- node[above right= 2pt] {ні} (do2);
  \path [line] (if) -- node[above ] {так} (do3);
\path [line] (if) -- (do4);
\path [line] (do4) -- (stop);
\end{tikzpicture}
```

Результат компіляції



4.4 Плаваючі об'єкти

Плаваючі об'єкти (floats) — це об'єкти, які не можуть бути розірвані під час переходу зі сторінки на сторінку, а мають бути розташовані на одній з них цілком. LaTeX 2_ε відносить до плаваючих об'єктів рисунки (figure) і таблиці (table), крім того, залишає користувачу можливість самому визначити додаткові плаваючі об'єкти. Якщо на такому об'єкті відбувається розрив сторінки, LaTeX 2_ε переносить його повністю в інше місце відповідно до вказівок користувача. Таким місцем може бути початок наступної сторінки.

Під час виклику оточення вказують тип плаваючого об'єкту й способи його розташування на сторінці: `\begin{float}[loc]`. Значення параметру loc наведено нижче:

h (here) — в поточній позиції тексту, де розташовано плаваючий об'єкт. Це значення не можна використовувати під час верстки у дві колонки з плаваючим об'єктом шириною у дві колонки;

t (top) — у верхній частині текстової сторінки;

b (bottom) — у нижній частині текстової сторінки. Це значення не можна використовувати під час верстки у дві колонки з плаваючим об'єктом шириною у дві колонки;

p (page of floats) — на окремій сторінці, яка містить лише плаваючі об'єкти.

За замовчуванням, якщо параметр loc не задано, використовується спосіб розташування tbp, який дозволяє розміщувати плаваючі об'єкти у верхній чи нижній частинах сторінки чи на окремій сторінці. Плаваючий об'єкт всередині абзацу не буде опрацьовуватися до завершення верстки абзацу.

В стандартному LaTeX 2_ε контейнер, який містить плаваючий об'єкт, помічається як оточення figure чи table. Припускається, що перше містить рисунок, а друге — таблицю. Обдва ці оточення підтримують підпис (назву), що визначається командою `\caption{}`, з автоматичною нумерацією в рамках відповідного оточення. Плаваючий об'єкт figure чи table може містити дві і більше команди, які задають підпис, що сформує кілька підписів з послідовною нумерацією. Тіло оточення форматується як звичайний абзац.

Якщо треба розташувати рисунок з підписом по центру сторінки, можна скористатися командами `\centering`, `\centerline` або оточенням center. Якщо необхідно розташувати два рисунки поруч в межах одного плаваючого об'єкту з одним підписом, можна скористатися оточенням minipage й отримати результат, як показано нижче.

```
\begin{figure}[ht]
\centering
\begin{minipage}{3cm}
\includegraphics[scale=0.58]{emblema.png}
\end{minipage}
\qquad
\begin{minipage}{3cm}
\includegraphics[scale=0.58]{emblema.png}
\end{minipage}
\caption{Емблема}
\end{figure}
```

Результат компіляції:

Якщо кожен рисунок повинен мати свій підпис, варто змінити визначення й отримати результат, як це показано нижче.

```
\begin{figure}[!h]
```

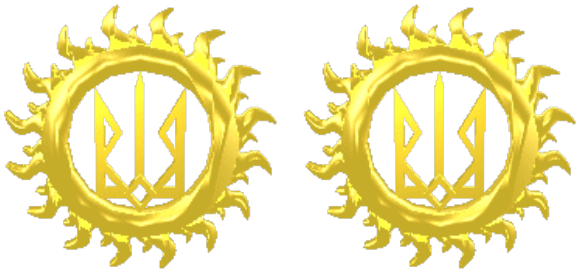


Рисунок 4.3 — Емблема

```
\begin{minipage}{0.45\textwidth}
\centering\includegraphics[scale=0.58]{emblema.png}
\caption{Емблема. Ліворуч}
\end{minipage}
\qquad
\begin{minipage}{0.45\textwidth}
\centering\includegraphics[scale=0.58]{emblema.png}
\caption{Емблема. Праворуч}
 \end{minipage}
\end{figure}
```

Результат компіляції:

Рисунок 4.4 — Емблема. Ліворуч

Рисунок 4.5 — Емблема. Праворуч

LaTeX 2$_\varepsilon$ налаштовано на американські правила набору тексту. Українізований стандартним чином (через babel) LaTeX 2$_\varepsilon$ не зовсім відповідвє цим вимогам. Підпис має вигляд: «Рис.», пробіл, номер рисунку, двокрапка і пробіл. Таблиця нумерується аналогічно. Задовільно налаштовувати формат підпису до рисунку і таблиці LaTeX 2$_\varepsilon$ майже не дозволяє. Змінити назву рисунку можна, якщо в преамбулі документу вказати:

```
\makeatletter
\renewcommand{\fnum@figure}{\textbf{Рисунок~\thefigure}}
\makeatother
```

Підпис набуде вигляду: «Рисунок», пробіл, номер рисунку, двокрапка і пробіл. Значно складніше з нумерацією, і для цього зазвичай використовують пакети розширення. У досить широких межах модифікувати формат підпису дозволяють пакети caption і ccaption. Вони легко дозволяють отримати формат підпису з крапкою і пробілом як розділовий знак. Для цього достатньо додати в преамбулу рядок:

usepackage[format=plain,labelsep=period]{caption}.

Цей пакет дозволяє змінювати формат підпису «на ходу», для чого параметри нового підпису можуть бути визначені командою: **captionsetup** наприклад: **captionsetupformat=plain,labelsep=colon**, — і після цієї команди для всіх наступних плаваючих об'єктів розділовим знаком стане двокрапка з пробілом. Якщо ж цю команду розташувати всередині плаваючого об'єкту до команди **caption**, то ця зміна формату діятиме лише для цього об'єкту.

4.5 Додавання фонових об'єктів

Вставити малюнок, щоб на його тлі був текст або малюнок, можна з використанням таких пакетів: **draftwatermark, ncctools, bophook, eso-pic, wallpaper**. Найкраще це робити за допомогою останнього.

Цей стильовий файл забезпечує такі команди:

CenterWallPaper Центрує фоновий малюнок на сторінці, дозволяє шкалювати відносно сторінки за шириною та висотою, підтримує, залишаючи співвідношення сторін незмінним.

ThisCenterWallPaper Як і попередня, але тільки для поточної сторінки.

TileWallPaper Дозволяє редагувати мозаїчні зображення з будь-яким співвідношенням сторін, яке не обов'язково зберігається.

ThisTileWallPaper Як і попередня, але тільки для поточної сторінки.

TileSquareWallPaper Дозволяє редагувати мозаїчні зображення, встановлюючи співвідношення сторін 1 : 1, користувач встановлює кількість мозаїк за шириною паперу.

ThisTileSquareWallPaper Як і попередня, але тільки для поточної сторінки.

ULCornerWallPaper Встановлює фонове зображення у верхньому лівому кутку сторінки, дозволяє масштабувати до частин ширини або висоти сторінки, залишає співвідношення сторін незмінним.

ThisULCornerWallPaper Як і попередня, але тільки для поточної сторінки.

\LLCornerWallPaper Встановлює фонове зображення у нижньому лівому кутку сторінки, дозволяє масштабувати до частин ширини або висоти сторінки, залишає співвідношення сторін незмінним.

\ThisLLCornerWallPaper Як і попередня, але тільки для поточної сторінки.

\URCornerWallPaper Встановлює фонове зображення у верхньому правому кутку сторінки, дозволяє масштабувати до частин ширини або висоти сторінки, залишає співвідношення сторін незмінним.

\ThisURCornerWallPaper Як і попередня, але тільки для поточної сторінки.

\LRCornerWallPaper Встановлює фонове зображення у нижньому правому кутку сторінки, дозволяє масштабувати до частин ширини або висоти сторінки, залишає співвідношення сторін незмінним.

\ThisLRCornerWallPaper Як і попередня, але тільки для поточної сторінки.

\ClearWallPaper Очищує фонове зображення.

Команди **\CenterWallPaper** та **\ThisCenterWallPaper** мають такий синтаксис:

\CenterWallPaper{<scaling>}{<filename>}

Параметр масштабування визначає розмір бажаного фонового малюнка у вигляді частки від ширини або висоти паперу, залежно від формату зображення, що зазначено другим параметром (якщо необхідне розширення не було задано), і співвідношення сторін паперу.

Команди **\TileWallPaper** та **\ThisTileWallPaper** мають такий синтаксис:

\TileWallPaper{<width>}{<height>}{<filename>}

Перші два параметри визначають потрібну ширину і висоту кожної мозаїки. Це може змінити співвідношення сторін зображення, яке зазначено третім параметром. Множина мозаїк, що необхідна для заповнення всієї сторінки, розміщена на кожній сторінці.

Команди **\TileSquareWallPaper** та **\ThisTileSquareWallPaper** мають такий синтаксис:

\TileSquareWallPaper{<number>}{<filename>}

У цьому випадку співвідношення сторін фіксується як 1 : 1, незалежно від співвідношення сторін зазначеного другим параметром. Перший параметр визначає, скільки мозаїк має бути розміщено по ширині сторінки. Як і вище, кількість мозаїк, що необхідні для заповнення сторінки, визначається автоматично.

Команди **\ULCornerWallPaper**, **\XXCornerWallPaper** та **\ThisXXCornerWallPaper** мають такий синтаксис:

\ULCornerWallPaper{<scaling>}{<filename>}

Параметр масштабування визначає розмір бажаного фонового малюнка у вигляді частки від ширини або висоти паперу, залежно від формату зображення, зазначеного другим параметром (якщо необхідне розширення не було задано), і співвідношення сторін паперу.

4.6 Створення анімаційних pdf–файлів за допомогою LaTeX

Анімаційні pdf–файли можна створити з двох складових:

1. З рисунків, що побудовані за допомогою пакетів LaTeX 'picture', PSTricks або pgf\TikZ, створюючи окремі елементи рисунку — кожен елемент це окремий фрейм.

2. З анімаційного gif рисунку. Попередньо з анімаційного gif рисунку утворюють набір файлів — кожен файл це окремий фрейм.

Крім цих файлів, в обох випадках має бути текстовий файл, який містить послідовність відображення фреймів.

Іноді в преамбулі може бути опис текстового файлу. Тоді, щоб цей файл з'явився в директорії, необхідно двічі запустити команду трансляції. Щоб трансляцію запускати один раз, цей файл має бути в директорії. Якщо цю інформацію розташовують в преамбулі, тоді вона має розпочинатися командою **\begin{filecontents}{fn.fm}**. Тут fn — ім'я файлу, fm — розширення файлу. Інформація закінчується командою **\end{filecontents}**.

Пакет **animate** містить команду \animategraphics[<options>]{<frame rate>} {<file basename>} {<first>} {<last>} для створення аніміційного pdf–файлу з аніміційного малюнку gif і оточення

\begin{animateinline}[<options>]{<frame rate>}
. . . . . надрукований матеріал . . . . . .
\newframe[<frame rate>]
. . . . . надрукований матеріал . . . . . .
\newframe*[<frame rate>]
. . . . . надрукований матеріал . . . . . .
\newframe
\multiframe{<number of frames>}{[<variables>]}{
. . . . . надрукований матеріал . . . . . .
}
\end{animateinline}

для створення аніміційного pdf–файлу з рисунків, що побудовано за допомогою пакетів LaTeX **picture, PSTricks** або **pgf\TikZ**.

Якщо **\animategraphics** збирає анімацію з графічних файлів або з багатосторінкового pdf-файла, оточення **animateinline** створює анімацію з залученого до нього надрукованого матеріалу.

Опції і в команді **animategraphics**, і в оточенні **animateinline** є однаковими.

Параметр **frame rate** визначає кількість кадрів за секунду анімації.

Команда **newframe** завершує кадр і починає наступний. Вона може використовуватись лише всередині оточення **animateinline**. Команда **newframe*** створює паузу анімації на певному кадрі.

Анімація будується покадрово за порядком залучення матеріалу. Однак, контроль порядку появи, накладання і повторення матеріалу доступний через опцію **timeline**.

Усі графічні файли послідовності повинні бути занумеровані. <**file basename**> — це крайня ліва частина імені файлу, яка є загальною для всіх членів послідовності. <**first**>, <**last**> — це, відповідно, номер першого і останнього файла послідовності. Імена файлів можуть бути занумеровані так: 0 . . . 99. Якщо вони починаються з нулів, то слід перевірити, чи всі вони мають однакову кількість цифр.

Анімація продовжується після клацання на неї. Обидві команди **newframe** мають аргумент, який змінює частоту кадрів всередині анімації.

Команда **multiframe** створює повторення картинок. Перший аргумент цієї команди задає кількість кадрів. Другий аргумент задає список визначених змінних через кому. Список може мати довільну довжину. У третьому аргументі змінні використовуються для параметризації картинок.

Формати файлів залежать від драйвера виведення і мають таку послідовність: **‘pdf’, ‘mps’, ‘png’, ‘jpg’, ‘jpeg’, ‘jbig2’, ‘jb2’, ‘jp2’, ‘j2k’, ‘jpx’**. Це для випадку, коли анімаційний pdf–файл утворюється з аніміційного малюнку **gif**.

Детальніше розглянемо опції пакету **animate**:

autoplay — Анімація починається після того, як сторінку відкрито. Також відновлює програвання з попередньо перерваної анімації.

autopause — Пауза в анімації, коли сторінку закрито замість зупинки та переходу до фрейму за замовчуванням.

autoresume — Відновлює попередньо призупинену анімацію, коли сторінку відкрито знову.

loop — Анімація перезапускається знову після досягнення кінця.

controls — Вставляє кнопки керування ( ) нижче вікна анімації. Ці кнопки означають:

- — зупинка та перший фрейм;
- — крок назад;
- — програвання назад;

 — програвання вперед;

 — крок вперед;

 — зупинка та останній фрейм;

 — зменшення швидкості;

 — швидкість за замовчуванням;

 — збільшення швидкості;

Обидві кнопки  заміщуються великою кнопкою  поки анімація програється.

palindrome — Анімація неперервно відбувається вперед і назад.

step — Анімація здійснюється покроково по клацанню мишки. Аргумент **frame rate** ігнорується.

poster[=first | none | last] — Специфікує, який фрейм відображається і друкується якщо анімація не є активною. Перший фрейм відображається за замовчуванням, нумерується як ‘0’.

buttonsize=<size> — Зміна висоти <size> кнопок керування. Висота за замовчуванням 1.44em.

buttonbg=<colour> — Колір тла для кнопок.

buttonfg=<colour> — Зміна кольору кнопок керування.

begin={<begin text>}

end={<end text>} (тільки ‘animateinline’.) <begin text> та <end text> вставляється в tex–файл на початку та кінці кожного фрейму.

Для встановлення оточення малювання використовується:
begin={\begin{pspicture}(. . . , . . .)(. . . , . . .)}, end={\end{pspicture}}
‘tikz-малюнки’ можуть визначати розміри графічних об’єктів. Межі об’єкту можна забезпечити невидимим об’єктом ‘rectangle’:
begin={ \begin{tikzpicture} \useasboundingbox (... , ...) rectangle (... , ...);
},
end={\end{tikzpicture}}

width=<width>
height=<height>
depth=<depth>
Змінює розміри вікна.

Опція **depth** визначає, як далеко від анімаційного вікна знаходиться наступний текст. Якщо хоча б одну опцію задано, інші змінюються пропорційно.

4.6.1 *Графічні елеменри: лінії та фігури*

Для покращення зображення графічного матеріалу використовують різні типи ліній. Одним з найважливіших параметрів лінії є її товщина.

Товщина лінії задається командою
/tikz/line width=<dimension> (початкова товщина 0.4pt)

Визначаємо лінію товщиною 5pt.

```
\tikz \draw[line width=5pt] (0,0) -- (1cm,1.5ex);
```

У pgf/tikz існує низка перевизначених стилів, що забезпечують більш природне визначення товщини лінії.

- /tikz/ultra thin

  Команда визначає лінію товщиною 0.1pt.

  ```
  \tikz \draw[ultra thin] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/very thin

  Команда визначає лінію товщиною 0.2pt.

  ```
  \tikz \draw[very thin] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/thin

  Команда визначає лінію товщиною 0.4pt.

  ```
  \tikz \draw[thin] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/semithick

  Команда визначає лінію товщиною 0.6pt.

  ```
  \tikz \draw[semithick] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/thick (style, no value)

  Команда визначає лінію товщиною 0.8pt.

  ```
  \tikz \draw[thick] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/very thick

  Команда визначає лінію товщиною 1.2pt.

  ```
  \tikz \draw[very thick] (0,0) -- (1cm,1.5ex);
  ```

- /tikz/ultra thick

  Команда визначає лінію товщиною 1.6pt.

  ```
  \tikz \draw[ultra thick] (0,0) -- (1cm,1.5ex);
  ```

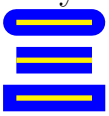
  

Команда /tikz/line cap=<type> (no default, initially butt) визначає, як лінії “закінчуються“.

Дозволені <type> є: round, rect, і butt. Вони мають такі ефекти:

```
\begin{tikzpicture}
\begin{scope}[line width=10pt, blue]
\draw[line cap=rect] (0,0 ) -- (1,0);
\draw[line cap=butt] (0,.5) -- (1,.5);
\draw[line cap=round] (0,1 ) -- (1,1);
\end{scope}
\draw[yellow, line width=2pt]
(0,0 ) -- (1,0) (0,.5) -- (1,.5) (0,1 ) -- (1,1);
\end{tikzpicture}
```

Результат компіляції:



Команда /tikz/line join=<type> визначає як лінії "з’єднують". Дозволені <type> є: round, bevel і miter. Вони мають такі ефекти:

```
\begin{tikzpicture}[line width=10pt]
\draw[red, line join=round] (0,0) -- ++(.5,1) -- ++(.5,-1);
\draw[blue, line join=bevel] (1.25,0) -- ++(.5,1) -- ++(.5,-1);
\draw[green, line join=miter] (2.5,0) -- ++(.5,1) -- ++(.5,-1);
\useasboundingbox (0,1.5); % enlarge bounding box
\end{tikzpicture}
```

Результат компіляції:

Під час використання з’єднання miter join і коли кут дуже гострий (малий кут), з’єднання miter може знаходитися дуже далеко від дійсної точки з’єднання. У цьому випадку, якщо виступ набагато більший ніж у <factor> разів за товщину лінії, з’єднання miter пересувається.

/tikz/miter limit=<factor>

```
\begin{tikzpicture}[line width=5pt]
```

```
\draw (0,0) -- ++(5,.5) -- ++(-5,.5);
\draw[miter limit=25] (6,0) -- ++(5,.5) -- ++(-5,.5);
\useasboundingbox (14,0); % make bounding box bigger
\end{tikzpicture}
```

Результат компіляції:

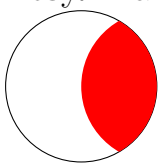
Опція /**tikz**/**clip** викликає всі підопції малювання для стискання поточної путі та розміру додаткових путів, які не є важливими для розміру картинки.

`\clip (a,b)`,

тут a — відстань від центру по горизонталі ($a > 0$ — праворуч від центру, $a < 0$ — ліворуч від центру); b — відстань від центру по вертикалі ($b > 0$ — вище центру, $b < 0$ — нижче центру) параметр b визначає відстань до кнопок controls.

```
\begin{tikzpicture}
\draw[clip] (0,0) circle (1cm);
\fill[red] (1,0) circle (1cm);
\end{tikzpicture}
```

Результат компіляції:



Це зазвичай принагідна ідея для застосування опцій clip тільки у першій команді в scope.

Якщо ви «бажате тільки clip» і не бажаєте нічого малювати, ви можете використати команду `\clip`, яка є скороченням для `\path[clip]`.

```
\begin{tikzpicture}
\clip (0,0) circle (1cm);
\fill[red] (1,0) circle (1cm);
\end{tikzpicture}
```

Результат компіляції:

draw В {tikzpicture} це є абревіатурою `\path[draw]`.

fill В {tikzpicture} це є абревіатурою `\path[fill]`.

filldraw В {tikzpicture} це є абревіатурою `\path[fill,draw]`.

Типи ліній можуть бути:
Суцільні
`\tikz \draw (0,0) -- (2cm,0);`
Точкові
`\tikz \draw[dotted] (0,0) -- (2cm,0);`
Ущільнені точкові
`\tikz \draw[densely dotted] (0,0) -- (2cm,0);`
Розріджені точкові
`\tikz \draw[loosely dotted] (0,0) -- (2cm,0);`
Пунктирні
`\tikz \draw[dashed] (0,0) -- (2cm,0);`
Ущільнені пунктирні
`\tikz \draw[densely dashed] (0,0) -- (2cm,0);`
Розріджені пунктирні
`\tikz \draw[loosely dashed] (0,0) -- (2cm,0);`
Штрих-пунктирні
`\tikz \draw[dashdotted] (0,0) -- (2cm,0);`
Ущільнені штрих-пунктирні
`\tikz \draw[densely dashdotted] (0,0) -- (2cm,0);`
Розріджені штрих-пунктирні
`\tikz \draw[loosely dashdotted] (0,0) -- (2cm,0);`
Штрих-пунктирні з подвійними крапками
`\tikz \draw[dashdotdotted] (0,0) -- (2cm,0);`
Ущільнені штрих-пунктирні з подвійними крапками
`\tikz \draw[densely dashdotdotted] (0,0) -- (2cm,0);`
Розріджені штрих-пунктирні з подвійними крапками
`\tikz \draw[loosely dashdotdotted] (0,0) -- (2cm,0);`

Окремим видом ліній є стрілки, їм притаманні всі опціє, що застосовують до ліній. Крім цього стрілки мають свої опції.

Стрілку у **tikz** зображують таким чином

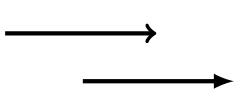
- стрілка праворуюч `->` або ліворуч `<-`;
- стрілка на обох кінцях лінії `<->`;
- подвійна стрілка праворуюч `->>` або ліворуч `<<-`;
- подвійні стрілкина обох кінцях лінії `<<->>`.

Замість знаків $<$ та $>$ можна використовувати опції **latex**,), *, o. Ця опція дещо змінює вигляд стрілки. Більший набір опцій міститься в бібліотеці **arrow**.

Наведемо кілька зображень стрілок.

Стрілка праворуоч

```
\tikz \draw[ultra thick,->] (0,0) -- (2cm,0);
\tikz \draw[ultra thick,-latex] (0,0) -- (2cm,0);
```



Подвійна стрілка ліворуоч

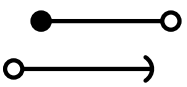
```
\tikz \draw[ultra thick,<<-] (0,0) -- (2cm,0);
```

Стрілка на обох кінцях лінії

```
\tikz \draw[ultra thick,latex-latex] (0,0) -- (2cm,0);
```

Специфічні символи на кінцях стрілок

```
\tikz \draw[ultra thick,*-o] (0,0) -- (2cm,0);
\tikz \draw[ultra thick, o-)] (0,0) -- (2cm,0);
```



Крім ліній у використовують великий набір фігур — найпростіші з них:

- ґратка;
- коло;
- еліпс;
- дуга кола;
- дуга еліпса;
- прямокутник;
- парабола;
- крива Без'є.

Ґратку створюють за допомогою команди

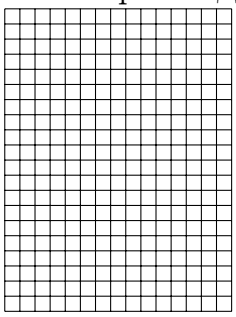
\draw[options] (x_b, y_b) **grid** (x_e, y_e)**;**

тут **options** — опції ліній (якщо потрібні) та крок ґратки **step** $=$ **h**, h — величина та розмірність кроку ґратки. За замовчуванням **step** дорівнює 1см.

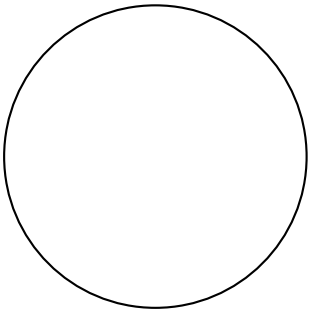
(x_b, y_b) — координати одного з кутів прямокутника, в якому розрашовано ґратку;

(x_e, y_e) — координати другого з кутів прямокутника, в який розрашовано по діагоналі щодо першого кута.

Наприклад, `\tikz \draw[step=0.2cm] (0,0) grid (3,4);`

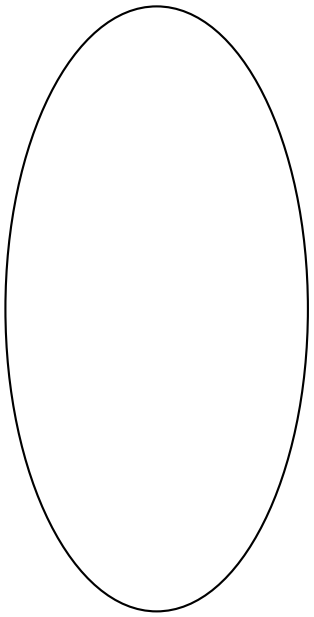


Коло створюють за допомогою команди
\draw[options] (x_b, y_b) circle (r);
тут **options** — опції ліній (якщо потрібні);
(x_b, y_b) — координати центру кола;
(r) — радіус кола.



Наприклад, `\tikz \draw[thick](0,0) circle(2);`

Еліпс створюють за допомогою команди
\draw[options] (x, y) ellipse (a and b);
тут **options** — опції ліній (якщо потрібні);
(x, y) — координати точки перетину осей еліпса;
(a and b) — довжини півосей еліпса.



Наприклад, `\tikz \draw[thick](0,0) ellipse (2 and 4);`

Дугу кола створюють за допомогою команди
\draw[options] (x, y) arc ($a_b : a_e : r$);
тут **options** — опції ліній (якщо потрібні);

(x, y) — координати центру кола;
$(a_b : a_e : r)$ — початковий кут дуги a_b, кіцевий кут дуги a_e, радіус дуги r. Кути задають в градусах, радіус — у одиницях довжини.

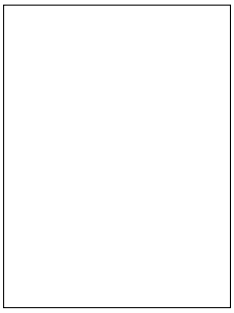
Наприклад, `\tikz \draw[thick](1,0) arc (-20:20:1);`

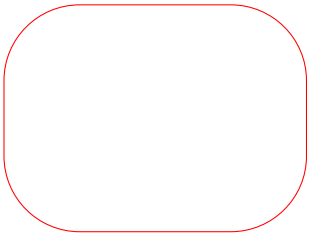
Дугу еліпса створюють за допомогою команди
\draw[options] (x, y) **arc** $(a_b : a_e : r_1$ *and* $r_2)$**;**
тут **options** — опції ліній (якщо потрібні);
(x, y) — координати точки перетину осей еліпса;
(a and b) — довжини півосей еліпса. $(a_b : a_e : r)$ — початковий кут дуги a_b, кіцевий кут дуги a_e, радіуси дуг r_1 та r_2. Кути задають в градусах, радіуси — у одиницях довжини.

Наприклад, `\tikz \draw[thick](1,0) arc (-20:20:2 and 1);`

Прямокутник створюють за допомогою команди
\draw[options] (x_b, y_b) **rectangle** (x_e, y_e)**;**
тут **options** — опції ліній (якщо потрібні) та заокруглення кутів прямокутника дугою кола з радіусом r **rounded corners** $=$ **r**.
(x_b, y_b) — координати одного з кутів прямокутника;
(x_e, y_e) — координати другого з кутів прямокутника, в який розрашовано по діагоналі щодо першого кута.

Наприклад, `\tikz \draw (0,0) rectangle (3,4);`





`\tikz \draw[red,rounded corners=1] (0,0) rectangle (4,3);`

Параболу створюють за допомогою кількох команд.

1. **\draw[options]** (x_b, y_b) **parabola** (x_e, y_e)**;**
   тут **options** — опції ліній (якщо потрібні);
   (x_b, y_b) — координати вершини параболи.

2. **\draw[options]** (x_b, y_b) **parabola[bend at end]** (x_e, y_e)**;**
   (x_e, y_e) — координати вершини параболи.

3. **\draw[options]** (x_b, y_b) **parabola bend** (x_h, y_h) (x_e, y_e)**;**
   (x_h, y_h) — координати вершини параболи.

Наприклад, `\tikz \draw (0,0) parabola (1,1);`

`\tikz \draw  (0,0) parabola[bend at end] (1,1);`

`\tikz \draw (0,0) parabola bend (.75,1.75) (1,1);`

Криві Без'є або криві Бернштейна-Без'є були розроблені у 60-х роках XX сторіччя незалежно одне від одного П'єром Без'є (Pierre Bezier) з автомобілебудівної компанії «Рено» та Полем де Кастельжо (Paul Faget de Casteljau) з компанії «Сітроен», де застосовували для проєктування кузовів автомобілів.

Ці криві названі так на честь французького математика П'єра Без'є, що вперше запропонував їх на початку 70-х років фірмі «Рено» для моделювання обводів кузова легкового автомобіля.

Криві Без'є використовують у комп'ютерній графіці для малювання плавних згинів (у таких застосуваннях, як Adobe Illustrator чи Macromedia Freehand), у CSS-анімації для опису процесу анімації тощо.

Відомо, що криві Без'є є фундаментальною основою практично всіх комп'ютерних додатків для комп'ютерної графіки.

Крива Без'є задається у параметричному вигляді, оскільки у загальному випадку геометричні контури не можуть бути описані у вигляді однозначної функції $y = f(x)$ [9].

Зазвичай у графічних пакетах застосовують кубічні криві Без'є, але деякі застосування, в тому числі Flash, працюють з квадратичними, фірма Microsoft використовує криві Без'є другого порядку свідомо через підвищення швидкодії.

Шрифти, що розроблено в Adobe, базуються на кубічних кривих Без'є й вони «якісніші» за шрифти TrueType, які базуються на квадратичних кривих [10].

Плоска крива Без'є третього порядку з чотирма опорними точками є окремим випадком, описується системою параметричних рівнянь (1) і є кубічним сплайном спеціального вигляду

$$\begin{aligned} x(t) &= (1-t)^3 x_1 + 3t(1-t)^2 x_2 + 3t^2(1-t)x_3 + t^3 x_4; \\ y(t) &= (1-t)^3 y_1 + 3t(1-t)^2 y_2 + 3t^2(1-t)y_3 + t^3 y_4. \end{aligned} \tag{1}$$

Параметризація здійснюється завданням декартових координат точки кривої як функцій деякого параметра.

Параметричний метод задання кривих має такі переваги:

- простіше обчислення координат точок;
- спрощення розрахунків при перетвореннях кривих;
- спрощення розрахунків, пов'язаних з підготовкою інформації для верстатів з числовим програмним управлінням. У геометричному моделюванні

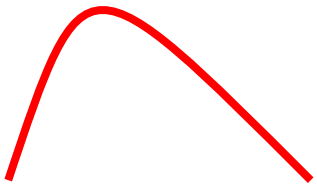
велике теоретичне і прикладне значення має розробка методів і алгоритмів проектування складних криволінійних обводів, що описуються параметричними поліномами. [11]

Криву Без'є другого порядку створюють за допомогою команди
draw[options] (x_b, y_b) **.. controls** (x, y) **..** (x_e, y_e)**;**
тут **options** — опції ліній (якщо потрібні);
(x_b, y_b) — координати початку кривої;
(x_e, y_e) — координати кінця кривої;
(x, y) — координати опорної точки, які впливають на форму кривої.

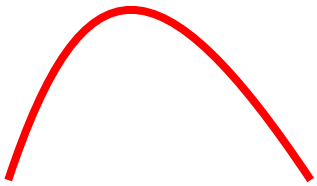
Криву Без'є третього порядку створюють за допомогою команди
draw[options] (x_b, y_b) **.. controls** (x_1, y_1) **and** (x_2, y_2) **..** (x_e, y_e)**;**
тут **options** — опції ліній (якщо потрібні);
(x_b, y_b) — координати початку кривої;
(x_e, y_e) — координати кінця кривої;
(x, y) — координати опорних точок, які впливають на форму кривої.

Наприклад,

```
\tikz \draw[line width=3pt,red] (0,0) .. controls (1,3) .. (4,0);
```



```
\tikz \draw[line width=3pt,red] (0,0) .. controls (1,3) \\ and (2,3).. (4,
```



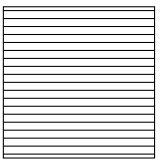
Для створення креслень часто використовують штрихування, різні форми якого містяться в бібліотеці **patterns**

Штрихування креслень створюють за допомогою команди

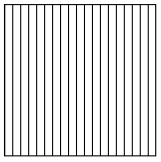
draw[pattern=image] (x_b, y_b) **shape** (x_e, y_e)**;**
тут (x_b, y_b) **та** (x_e, y_e) — координати форми, яку штрихують;
image — види штрихування;
shape — форма, яку штрихують.

Розглянемо види штрихування:

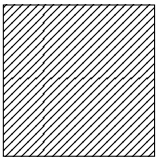
- горизонтальні лінії

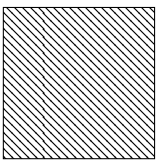
```
\tikz \draw[pattern=horizontal lines] (0,0) rectangle (2,2);
```


- вертикальні лінії

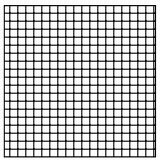
```
\tikz \draw[pattern=vertical lines] (0,0) rectangle (2,2);
```


- косі лінії

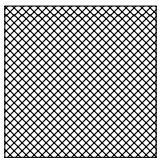
```
\tikz \draw[pattern=north east lines] (0,0) rectangle (2,2);
```


```
\tikz \draw[pattern=north west lines] (0,0) rectangle (2,2);
```


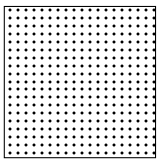
- сітка

```
\tikz \draw[pattern=grid] (0,0) rectangle (2,2);
```


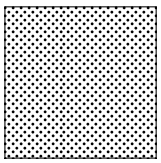
- коса сітка

```
\tikz \draw[pattern=crosshatch] (0,0) rectangle (2,2);
```


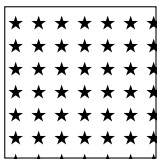
- точки

```
\tikz \draw[pattern=dots] (0,0) rectangle (2,2);
```


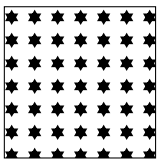
- косі точки

```
\tikz \draw[pattern=crosshatch dots] (0,0) rectangle (2,2);
```


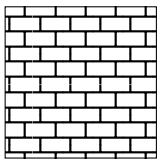
- п’ятикутні зірки

```
\tikz \draw[pattern=fivepointed stars] (0,0) rectangle (2,2);
```


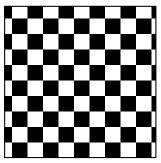
- шестикутні зірки

```
\tikz \draw[pattern=sixpointed stars] (0,0) rectangle (2,2);
```


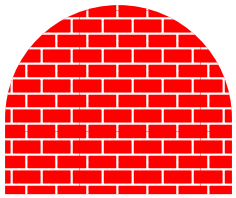
- брикети (цеглинки)

```
\tikz \draw[pattern=bricks] (0,0) rectangle (2,2);
```


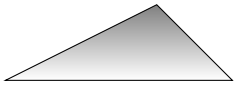
- шахматка

```
\tikz \draw[pattern=checkerboard] (0,0) rectangle (2,2);
```


Наприклад,

```
\begin{tikzpicture}
\def\mypath{(0,0) -- +(0,1) arc (180:0:1.5cm) -- +(0,-1)}
\fill [red] \mypath;
\pattern[pattern color=white,pattern=bricks] \mypath;
\end{tikzpicture}
```


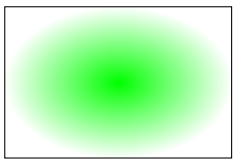
Подамо ефекти тіні [12, 13] з відповідними кодами.

```
\tikz \path[shade, draw](0,0)--(2,1)--(3,0)--cycle;
```


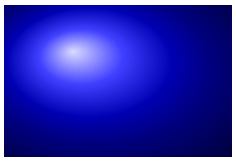
```
\tikz \shade[top color = green](0,0)--(2,1)--(3,0)--cycle;
```


```
\tikz \shade[top color = blue, bottom color=yellow](0,0) rectangle (3,2);
```


```
\tikz \shade[draw,shading = radial, inner color=green](0,0) rectangle (3,2
```



```
\tikz \shade[shading = rectangle, ball color=blue](0,0) rectangle (3,2);
```



```
  \tikz \shade[shading = ball, ball color=red](0,0) circle (0.5);
\tikz \shade[shading = ball, ball color=yelllow](2,0) circle (0.5);
\tikz \shade[shading = ball, ball color=green](3,0) circle (0.5);
```

4.6.2 *Опції timeline*

timeline = fn.fm
fn.fm – текстовий файл, який містить визначення порядку відображення підготованого матеріалу.

Вміст найпростішого файлу fn.fm

```
::0x0
::1x0
::2x0
::3x0
::4x0
::5x0
::6x0
::7x0
::8x0
::9x0
::10x0
::11x0
::12x0
::13x0
::14x0
::15x0,
```

де перше число у кожному рядку — номер фрейму.

Для утворення набору графічних файлів з анімаційного gif файлу необхідно виконати команду **convert fn.gif fn.png** з пакету ImageMagick

Для створення анімаційного pdf файлу бажано утворити окрему директорію, яка має містити: tex файл; fn.fm; набір графічних файлів. Утворення цього файлу вимагає завантаження таких стильових файлів, якщо вони не інстальовані в поточній версії програми:

- atbegshi.sty
- atenddvi.sty
- atveryend.sty
- auxhook.sty
- etexcmds.sty
- ifdraft.sty
- ifluatex.sty
- infwarerr.sty
- kvdefinekeys.sty
- kvsetkeys.sty
- ltxcmds.sty
- pdftexcmds.sty
- zref-abspage.sty
- zref-base.sty
- zref-lastpage.sty

Ці файли можуть знаходитись як у директорії, де знаходиться файл fn.tex, так і в LaTeX.

Для створення анімаційного pdf файлу необхідно виконати команду pdfelatex.exe fn1.tex в пакетному режимі, або завантажити цей файл з оболонки.

Розглянемо два приклади.

4.6.3 *Приклад № 1.*

Завдання: Побудуйте трикутник і проведіть навколо нього описуюче коло.

Цю побудову здійснюватимемо за таким алгоритмом:

- Малюємо зелений кружечок радіусом 2.5pt з координатами центру (0.5,0.5).
- Позначаємо цю точку літерою A чорного кольору.

- Малюємо зелений кружечок радіусом 2.5pt з координатами центру (4.5,0.5).
- Позначаємо цю точку літерою B чорного кольору.
- Малюємо зелений кружечок радіусом 2.5pt з координатами центру (3.5,3.5).
- Позначаємо цю точку літерою C чорного кольору.
- З'єднуємо точки AB відрізком червоного кольору.
- З'єднуємо точки BC відрізком червоного кольору.
- З'єднуємо точки AC відрізком червоного кольору.
- Проводимо висник зеленого кольору з середини відрізка AB.
- Проводимо висник зеленого кольору з середини відрізка BC.
- Проводимо висник зеленого кольору з середини відрізка AC.
- Малюємо червоний кружечок радіусом 3.5pt з координатами центру у точці перетину цих висників.
- Позначаємо цю точку літерою O чорного кольору.
- Малюємо коло блакитного кольору, яке проходить через вершини трикутника ABC.

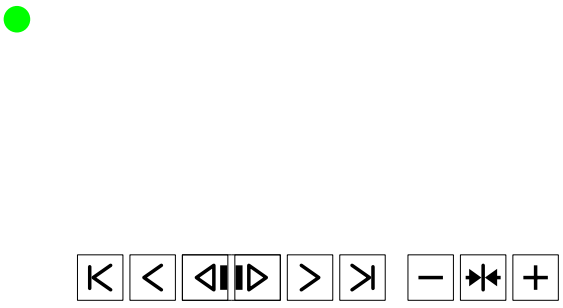
Втілення цього алгоритму наведено нижче.

```
\begin{animateinline}[
begin={
\begin{tikzpicture}[line cap=round,line join=round,x=1.0cm,y=1.0cm]
\clip(0.02,-1)  rectangle (4.96,4);
},
end={\end{tikzpicture}},
controls,timeline=test1.txt, scale=2]{2}
% Малюємо точку A
\fill [color=green] (0.5,0.5) circle (2.5pt);
% Позначаємо цю точку
\newframe
\begin{scriptsize}
\draw[color=black] (0.4,0.3) node {$A$};
\end{scriptsize}
% Малюємо точку B
\newframe
\fill [color=green] (4.5,0.5) circle (2.5pt);
```

```
% Позначаємо цю точку
\newframe
\begin{scriptsize}
\draw[color=black] (4.7,0.34) node {$B$};
\end{scriptsize}
% Малюємо точку C
\newframe
\fill [color=green] (3.5,3.5) circle (2.5pt);
% Позначаємо цю точку
\newframe
\begin{scriptsize}
\draw[color=black] (3.58,3.8) node {$C$};
\end{scriptsize}
% Малюємо відрізок AB
\newframe
\draw [color=red](0.5,0.5)-- (4.5,0.5);
% Малюємо відрізок BC
\newframe
\draw [color=red](4.5,0.5)-- (3.5,3.5);
% Малюємо відрізок AC
\newframe
\draw [color=red](3.5,3.5)-- (0.5,0.5);
% Малюємо висник до відрізку AB
\newframe
\draw [color=green](2.5,-1) -- (2.5,4);
% Малюємо висник до відрізку BC
\newframe
\draw [domain=0.02:4.96,color=green] plot(\x,{(2+\x)/3});
% Малюємо висник до відрізку AC
\newframe
\draw [domain=0.02:4.96,color=green] plot(\x,{(4-\x)});
\newframe
% Малюємо точку O -- центр перетину висників
\fill [color=red] (2.5,1.5) circle (3.5pt);
% Позначаємо цю точку
\newframe
\begin{scriptsize}
\draw[color=black] (2.6,1.78) node {$O$};
\end{scriptsize}
% Малюємо коло, що описане навкруг трикутника ABC
```

```
\newframe
\draw[color=blue](2.5,1.5) circle (2.24cm);
\end{animateinline}
```

Результат компіляції:



4.6.4 *Приклад № 2.*

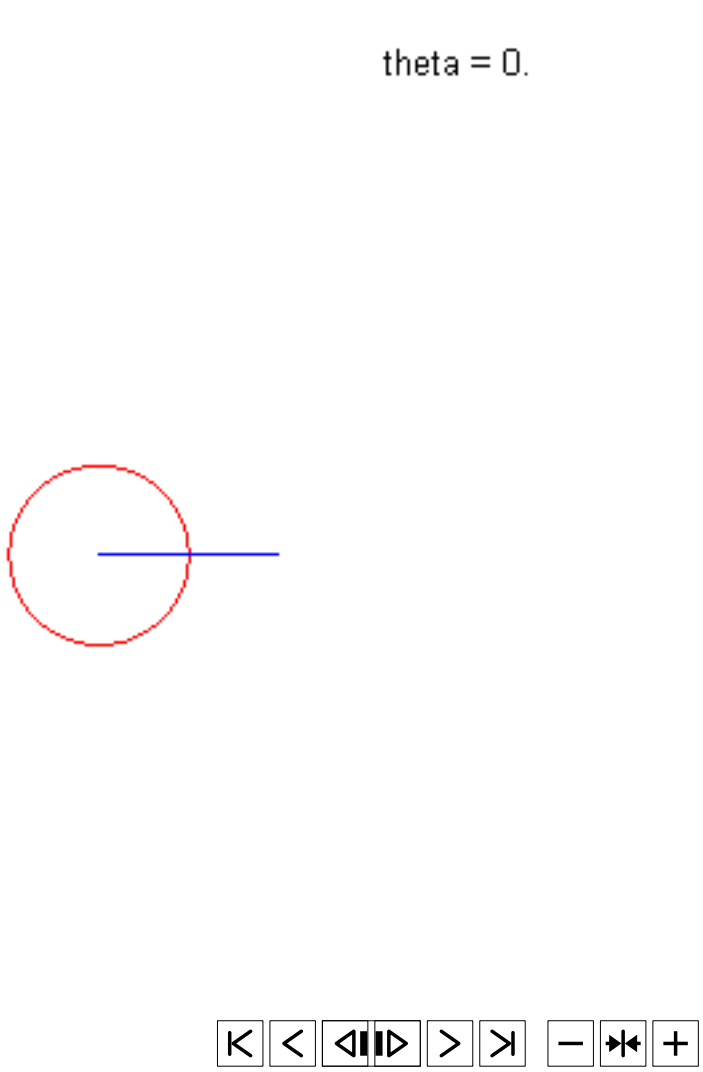
Завдання: створити анімаційний pdf-файл з анімаційного файлу.

Використовуючи систему комп'ютерної математики Maple, створюємо анімаційний рисунок.

Запам'ятовуємо цей рисунок як анімаційний gif файл. Анімаційний рисунок також можна отримати з іншої програми чи джерела.

За допомогою програми ImageMagick розбиваємо цей gif файл на окремі графічні файли.

```
\animategraphics[
  controls, loop,
  timeline=test2.txt
]{6}{aa-}{0}{24}
```



4.7 Побудова електричних та електронних схем

Іноді в LaTeX треба намалювати нескладну електричну схему. Це можна зробити за допомогою кількох пакетів. Хоча LaTeX й не може розглядатися як пакет для проектування електричних схем, але певні можливості в цій сфері він все ж таки має:

bloques — механізм для малювання схем силової електроніки;
eltex — електричні схеми з сіткою, яка спрощує розташування елементів;
circ — створення нескладних електричних схем засобами LaTeX і METAFONT;
circuits — візуалізація електричних схем засобами бібліотеки circuits;
circuitikz — візуалізація електричних схем засобами **TikZ**;
pst-circ — електричні схеми;
pst-osci — ілюстрації осцилограм.

Пакет **circ** є обв'язкою над спеціальними шрифтами, які дозволяють зобразити прості елементи типу резисторів, конденсаторів, транзисторів тощо. Нічого, крім LaTeX, для створення електронних схем з допомогою **circ** не потрібно.

В преамбулі треба завантажити стиль **circ**.

Трішки по-іншому працює пакет **pst-circ**.

Для створення рисунків з елементами електронних схем може також стати у нагоді пакет для MetaPost **makecirc**.

Наразі став популярним пакет **TikZ**, який має велику кількість бібліотек. Бібліотекою, яка придатна для малювання електричних та електронних схем є бібліотека **circuits**.

Іншим пакетом, що призначено для гарфічного зображення електричних та електронних схем, є пакет **circuitikz**, який базується на пакеті **TikZ**.

Далі покажемо, як здійснити побудову таких схем за допомогою цих пакетів.

4.7.1 *TikZ з бібліотекою circuits*

Для малювання електричних схем слугує бібліотека **circuits**. Її необхідно підключити в преамбулі документу за допомогою команди **usetikzlibrary**. Додатково як параметр оточення **tikzpicture** треба вказати **circuit ee IEC**.

\usetikzlibrary{circuits}
\usetikzlibrary{circuits.ee}
\usetikzlibrary{circuits.ee.IEC}
\usetikzlibrary{circuits.logic.IEC}

Елементи схем в цій бібліотеці є спеціальними типами вузлів й задаються як звичайні вузли TikZ командою **node**. Тут варто звернути увагу на те, що текст вузла, який задають у фігурних дужках, у радіоелементів завжди порожній. Позиційне позначення й номінал вводяться як параметр **info** в квадратних дужках. Парметр **point up** розгортає компонент на 90 градусів, а параметр **point down** — на 270 градусів.

```
\node (R) [resistor={info={$R$}}] at (2,2) {};
% горизонтальний резистор
\node (C1) at (3,0) [point up,capacitor={info = $C_1$,
info'= 100 пФ}] {}; %вертикальний конденсатор
```

Повний список всіх компонентів описано в документації до **TikZ**.

Вузол схеми задають спеціальним вузлом **contact**, який малює зафарбований кружечок: **node (p2) [contact] at (6,-2) ;**

Дроти між елементами задають як зазвичай в **TikZ** командою **draw**. Прямий дріт малюють командою **draw (R1) – (C1);**

А дріт під кутом 90 градусів малюють командою:

draw (R1) |- (C1);

або

draw (R1) -| (C1);

Відмінність полягає в тому, що у першому випадку дріт спочатку малюється вертикально, а потім горизонтально, а у другому випадку — навпаки.

До дротів можна застосувати ті ж параметри (стрілки, товщину лінії), як і до звичайних траєкторій. Наприклад, товстий дріт зі стрілкою:

\draw [thick,->] (R1) -| (C1);

Схеми можна поєднувати з будь-якою графикою **TikZ**.

Розглянемо приклад

```
\begin{document}

Опис RLC-контура

\begin{figure}[!h]
\begin{center}
\begin{tikzpicture}[circuit ee IEC] % обов'язково вказуємо
circuit ee IEC
\node (in) at (0,0) [contact] {}; % вхід - контакт
\node (L1) at (1,0) [inductor={info = $L_1$,
info'= 47 мкГн}] {}; % котушка -
% info - позначення на схемі , info' - номінал
\node (C1) at (3,0) [capacitor={info = $C_1$,
info'= 100 пФ}] {}; % конденсатор
\node (R) at (5,0) [resistor={info = $R_1$, info'= 2 Ом}] {};
% резистор
\node (out) at (6,0) [contact] {}; % вихідний зажим
\draw (in) -- (L1) -- (C1) -- (R) -- (out); % вихідний зажим
\end{tikzpicture}
\end{center}
\caption{Послідовний коливальний контур}
\end{figure}

\begin{figure}[!h]
\begin{center}
\begin{tikzpicture}[circuit ee IEC]
\node (R) [resistor={info={$R$}}] at (2,2) {}; % малюємо резистор
\node (p1) [contact] at (3,2) {}; % малюємо точку з'єднання R і C
\node (C) [point up, capacitor={info={$C$}}] at (3,1) {};
% конденстаор повернемо на 90 градусів
\node (p2) [contact] at (3,0) {}; % малюємо точку з'єднання
конденсатора із загальним дротом
```

```
\draw [-latex] (p1) -- (5,2); % з'єднуємо вузли схеми, R і C
\draw [latex-] (0,2) -- (R);
\draw (R) -- (p1) -- (C) -- (p2); % малюємо дроти
\draw [latex-] (0,0) -- (p2);
\draw [-latex] (p2) -- (5,0);
\node  at (0,1) {Вхід}; % підписуємо   Вхід
\node  at (5,1) {Вихід}; % і   Вихід

% тепер малюємо АЧХ як звичайний графік
\draw[xshift=60mm,-latex] (0,0) -- (4,0) node [anchor=west]
{$\omega$}; % вісь X
\draw[xshift=60mm,-latex] (0,0) -- (0,3) node [anchor=south]
{$K(\omega)$};
% вісь Y
\draw [very thick,xshift=60mm, y=2cm, x=1cm,
declare function={K(\w)=1/sqrt(1+\w^2);}] plot [domain=0:3,
samples=10,
smooth] (\x,{K(\x)}); % графік АЧХ
\end{tikzpicture}
\end{center}
\caption{RC --- ланцюг та його АЧХ}
\end{figure}
\end{document}
```

Результат компіляції наведено на рис. 4.6

4.7.2 *Пакет circuitikz*

Пакет **circuitikz** є подальшим розвитком бібліотеки **cirtuits**. Пакет містить значно більше компонентів:

- пасивні компоненти;
- транзистори;
- діоди;
- тиристори;
- логічні вентилі;
- трансформатори.

Пакет **circuitikz** не є сумісним з бібліотекою **circuits**. Використовувати їх разом не можна.

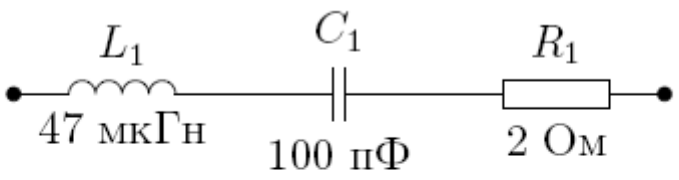


Рисунок 1 -- Послідовний коливальний контур

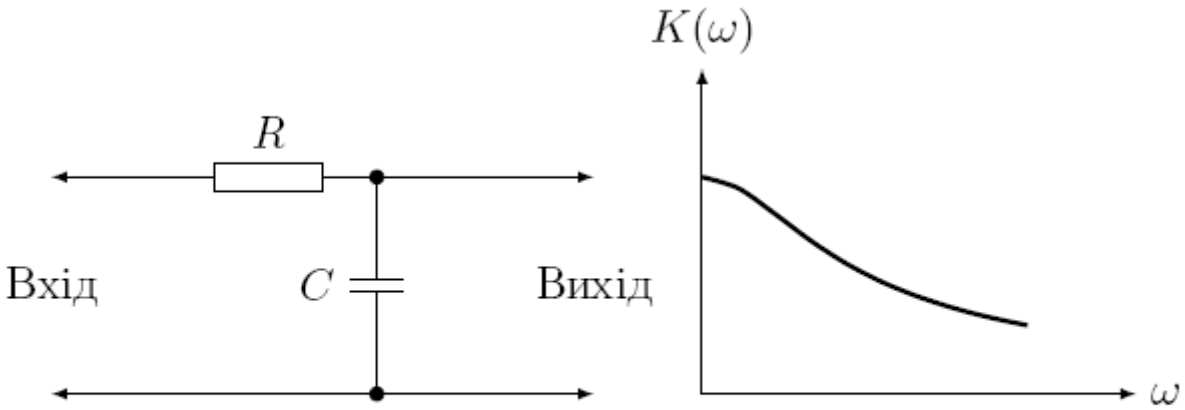


Рисунок 2 -- RC - ланцюг та його АЧХ

Рисунок 4.6 — Схеми, що побудовано за допомогою пакету TikZ з бібліотекою circuits

Для того, щоб позначення елементів на схемах були близькими до наших стандартів, пакет **circuitikz** підключається в преамбулі таким чином:

usepackage[**europeanresistors,americaninductors**]{**circuitikz**}

Пакет визначає спеціальне оточення, в якому треба розташовувати схеми:

```
\begin{circuitikz}
\end{circuitikz}
```

Ідеологія малювання схем відрізняється від бібліотеки **circuits**. Компоненти є не вузлами (nodes), а путями (paths). Щоб розташувати поодинокий компонент, треба створити путь командою **draw**. Поворот компонента здійснюється орієнтацією путі. Як приклад розглянемо схему тиристорного фазового регулятора та кілька схем, які використовують у вимірюваннях.

Схема тиристорного фазового регулятора

```
\begin{circuitikz}[european resistors, american inductors]
\draw (-1,0) to [sV,l=$U_c$] (-1,2) % джерело синусоїдальної
```

```
напруги
             to [R, l=$R_{\text{н}}$,i>_=$I_{\text{н}}$] (2,2)
 % резистор  зі спрямуванням струму
             to [Ty,l={VS},*-,n=VS] (2,0); % тиристор
\draw (VS.G) to [R,mirror,l=$R_{\text{y}}$,i<_=$I_{\text{y}}$]
++(4,0) |- (2,2);
\draw (VS.cathode) |- (-1,0);
\end{circuitikz}
```

Результат компіляції:

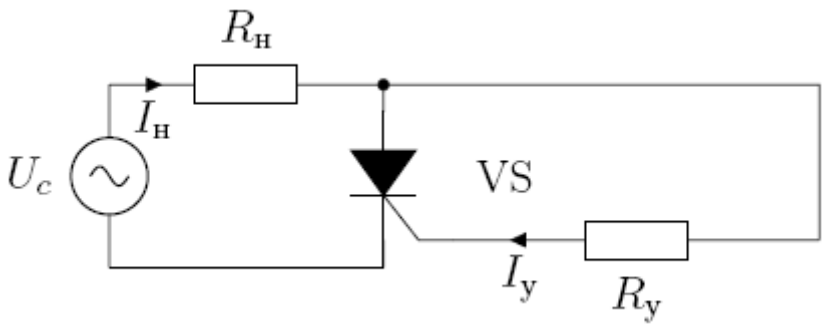


Схема транзисторного підсилювача змінного струму

```
\begin{figure}[h!]
  \begin{center}
\begin{circuitikz}[european resistors, american inductors]
% Малюємо нижню з'єднувальну лінію
      \draw (0,0) to[short,*- ] (8,0);
% Малюємо  заземлення
      \draw (2,0) to[short,*- ] node[ground] {} (2,-0.1);
% Малюємо резистори R1 та R2
      \draw (2,0) to[R=$R2$] (2,3)
      to[R=$R1$] (2,6);
% Малюємо роздільний конденсатор
  \draw (0,3) to[C=$C_P$,*-*] (2,3);
% Малюємо колекторний резистор
  \draw (4,4) to[R=$R_K$,*-*] (4,6);
% Малюємо емітерні резистор та конденсатор
  \draw (4,0) to[R=$R_E$,*-*] (4,2)
  to[short] (6,2) to[C=$C_E$, -*] (6,0);
% Малюємо навантаження
   \draw (4,4) to[C=$C_P$] (8,4) to[R=$R_H$] (8,0);
% Малюємо верхню з'єднувальну лінію
```

```
      \draw (2,6) to[short,-*] (10,6);
% Малюємо транзистор
  \draw (4,3) node[npn](npn1)  {VT1}
  (npn1.base) node[anchor=east] {}
  (npn1.collector) node[anchor=south] {}
  (npn1.emitter) node[anchor=north] {};
  \draw (2,3) -- (npn1.base);
  \draw (4,2) -- (npn1.emitter);
  \draw (4,4) -- (npn1.collector);
   \end{circuitikz}
  \end{center}
\end{figure}
```

Результат компіляції:

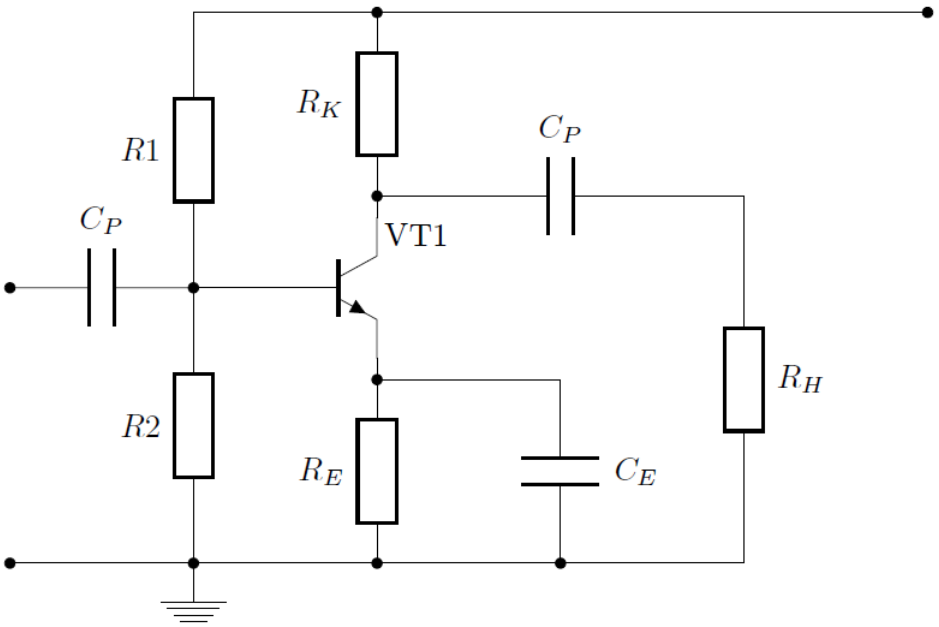
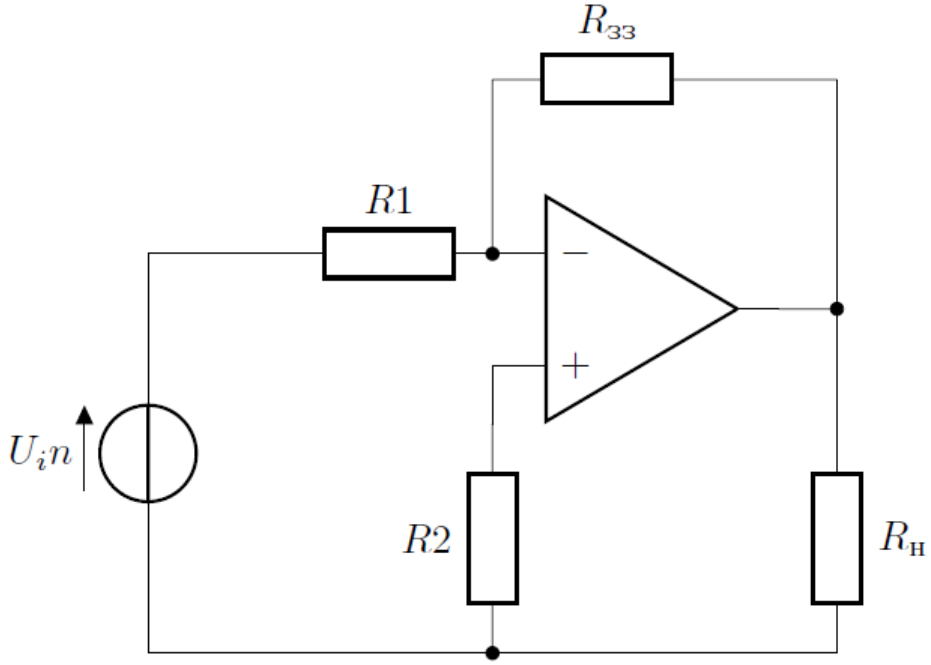


Схема підсилювача на операційному підсилювачі

```
\begin{figure}[h!]
  \begin{center}
\begin{circuitikz}[european
resistors, american inductors]
% Малюємо нижню з'єднувальну лінію
      \draw (0,0) to[short] (6,0);
% Малюємо резистор R2
      \draw (3,0) to[R=$R2$,*-] (3,2);
% Малюємо операційний підсилювач
\draw (4.3,3) node[op amp] (opamp) {}
```

```
  (opamp.+) %node[left] {$v_+$}
  (opamp.-) %node[left] {$v_??$}
  (opamp.out) %node[right] {$v_o$}
    ;
% З'єднуємо резистор R2 з неінвертуальним входом ОП
 \draw (3,2) |- (opamp.+);
 % Малюємо джерело сигналу
\draw (0,0)
      to[V,v=$U_in$] (0,3.48)
      to[short] (1,3.48);
 % Малюємо резистор R1
      \draw (1,3.48) to[R=$R1$] (3.2,3.48);
 % Малюємо резистори зворотного зв'язку та навантаження
 \draw (3,3.48) to[short,*-] (3,5) to[R=$R_{\text{зз}}$] (5,5)
to[short] (6,5) to[short] (6,2) to[R=$R_{\text{н}}$] (6,0);
  \draw (opamp.out) to[short,-*] (6,3);
  \end{circuitikz}
    \end{center}
\end{figure}
```

Результат компіляції:



Отже, для малювання схем в LaTeX можна використовувати бібліотеку **circuits** для TikZ чи пакет **circuitikz**.

Недоліком є те, що великі схеми уповільнюють компіляцію документа. На малювання схеми засобами TikZ витрачається багато часу.

5 Додавання таблиць

LaTeX має вбудовані засоби для побудови таблиць, а також може бути використано одне з таких оточень:

tabbing: створення таблиці в машинописному стилі;

tabular: універсальний засіб створення таблиць будь-яких стилів;

table: створення таблиці як «плаваючого» об'єкту;

array: створення таблиць в математичному стилі, зокрема векторів та масивів.

5.1 Оточення tabbing

Оточення використовується для створення відносно простих таблиць, якщо ширина колонки є відомою заздалегідь. Таблиця формується в ручному режимі: розбиття на колонки задається за допомогою керуючого символу \= (визначення позицій табуляції) у першому рядку таблиці. Далі порядково формуються комірки таблиці за допомогою команди \>. Перехід на наступний рядок задається командою \\. Вона може мати необов'язковий параметр для визначення додаткової відстані між рядками таблиці. Приклад: **[10pt]** задає додаткову відстань між рядками в 10pt.

Це оточення формує наповнення комірок тільки в один рядок форматованого тексту і не дозволяє використовувати форматовані абзаци з кількох рядків та перенесенням слів. Текст, що знаходиться між двома командами \>чи \= утворює групу, тому під час його форматуванні в межах комірки немає потреби використовувати фігурні дужки.

Для створення шаблону (розбиття на комірки) таблиці у першому рядку вводиться будь-який текст необхідної ширини, а сам рядок не відображається за допомогою команди **kill**.

Основним недоліком оточення є те, що за некоректного розбиття на колонки можливе наповзання тексту сусідніх комірок один на оден (текст не вміщується в межах заданої ширини комірок).

5.2 Оточення tabular

Для початківців найпростіше скористатися оточенням **tabular**. Це оточення є більш гнучким та універсальним засобом створення таблиць. В ньому ширина колонок вибирається автоматично за максимальною шириною їх вмісту. Основними елементами оточення (табл. 5.1) є: *преамбула таблиці* — задається як обов'язковий параметр та визначає загальне оформлення. Обов'язкові аргументи оточення **tabular** (у фігурних дужках) задають способи вирівнювання та

обробки тексту в колонках. Рядки таблиці відокремлюють командою \\, колонки — знаком &. У найпростішому випадку це набір літер (по одній на колонку), що описують структуру таблиці:

Таблиця 5.1 — Основними елементами оточення tabular

Літера	**Особливості форматування тексту**
l	Вирівнювання в колонці по лівому краю.
r	Вирівнювання в колонці по правому краю.
c	Вирівнювання в колонці по центру.
|	Використовується для відокремлення колонок таблиці вертикальними лініями.
@{символ}	Вставка між комірками будь-якого символу (символ), що задано як обов'язковий параметр.
p{N}	Текст у комірці буде розміщено у кілька рядків, якщо він має більшу ніж комірка ширину. N задає ширину колонки з використанням будь-яких допустимих одиниць довжини.

Розташуванням таблиці відносно тексту (якщо вона знаходиться в межах тексту) можна управляти за допомогою таких необов'язкових параметрів (табл. 5.2) у преамбулі таблиці:

Таблиця 5.2 — Необов'язкові параметри оточення tabular

Літера	**Особливості вирівнювання таблиці в тексті**
t	Вирівнювання по верхньому рядку — верхній рядок таблиці буде розміщено на одному рівні з текстовим рядком.
b	Вирівнювання по нижньому рядку — нижній рядок таблиці буде розміщено на одному рівні з текстовим рядком.
c	Вирівнювання по центру — середина таблиці буде розміщена на одному рівні із текстовим рядком.

Ширина комірок визначається автоматично за заданим форматом. Висота комірок визначається найбільшою висотою введеного тексту у рядку. Приклад формування преамбули таблиці:

\begin{tabular}[c]{||c|c|p{300pt}||}

Тут необов'язковий параметр **[c]** задає розташування таблиці у тексті документу. Преамбула формує таблицю, що складається з трьох колонок: текст в перших двох комірках буде вирівняно по центру, а для третьої комірки явним чином задано ширину 300pt. Таблиця зліва та справа буде обведена подвійною лінією, а комірки будуть розділені одинарними лініями.

Власне таблиця створюється за допомогою управляючих символів:

& — формування рядків таблиці (поділ на комірки);

\\ — формування колонок таблиці. Команда може мати необов'язковий параметр, що задає додатковий зсув по вертикалі. Наприклад: \\[2mm] задає додатковий зсув між рядками таблиці величиною 2mm.

Для форматування таблиці додатково можна використати такі команди:

hline — горизонтальна розділююча лінія на всю ширину таблиці;

cline — горизонтальна лінія тільки в межах заданих номерів комірок. Наприклад, \cline{2-3} проводить лінію тільки для другої та третьої комірки;

multicolumn — об'єднання кількох суміжних комірок рядка. Команда має три обов'язкових параметри, що задаються у окремих фігурних дужках: **multicolumn{число}{преамбула}{текст}**, де

- **число** — кількість комірок, що об'єднуються;
- **преамбула** — преамбула комірки є аналогічною до преамбули таблиці;
- **текст** — текст, що вводиться в цій комірці.

Для об'єднання комірок по вертикалі необхідно підключити в преамбулі документу пакет **multirow**. У пакеті визначено команду об'єднання комірок: **multirow** — об'єднання кількох суміжних комірок стовпчика (по вертикалі): **multirow{число}{ширина}{текст}**, де

- **число** — кількість комірок, що об'єднуються;
- **ширина** — ширина об'єднаної комірки. Якщо явно не задана — обчислюється автоматично;
- **текст** — текст, що вводиться в цій комірці.

Приклад використання команд об'єднання комірок для формування складної структури таблиці наведено нижче [2]:

```
\begin{tabular}[c]{|c|c|c|p{100pt}|}
\hline
% обєднання комірок для формування заголовку
% таблиці
\multicolumn{4}{|c|}{Заголовок } \\[2pt]
\hline
1&2 &\multicolumn{2}{|c|}{Підзаголовок } \\
\hline
\multirow{2}{40pt}{текст1} & & &
\multirow{4}{40pt}{текст2} \\
\cline{2-3}
& \multirow{3}{50pt}{текст3} & & \\
\cline{1-1} \cline{3-3}
\multirow{2}{40pt}{текст4}& & & \\
\cline{3-3}
```

```
& & & \\
\hline
\end{tabular}
```

Результат компіляції:

Заголовок			
1	2	Підзаголовок	
текст1	текст3		текст2
текст4			

Визначення в преамбулі таблиці параметру **p** не дозволяє центрувати текст в цих комірках. Для усунення цього в тих комірках, де текст має бути відцентрований, варто записати так: **multicolumn{1}{|c|}{*текст*}**

Щоб уникнути напівзаповнених сторінок, таблиця в тексті повинна «плавати» — змінювати своє розташування в межах сторінки, або переноситися на наступну. Для роботи з таблицею як з плаваючим об'єктом, формування заголовку та її автоматичної нумерації слугує оточення **table**.

Якщо необхідно змінити спосіб нумерації таблиць, замінивши наскрізну нумерацію, що використовується за замовчуванням, на нумерацію всередині рубрики, необхідно застосувати команду **numberwithin{table}{section}**.

Для створення таблиці, в оточенні **table** необхідно додатково використати одне з розглянутих вище оточень (**tabbing** або **tabular**) за таким синтаксисом:

begin{table}[розташування]
begin{tabular} ...
...
end{tabular}...
caption{підпис до таблиці}
label{мітка таблиці}

Необов'язковий параметр **[розташування]** визначає бажане розташування таблиці на сторінці:

h — в місці знаходження оточення **table** в тексті;
t — вгорі сторінки;
b — внизу сторінки;
p — на окремій сторінці, яка призначена для «плаваючих» об'єктів.

Одночасно можна використовувати кілька параметрів. Для стандартного класу документів **article** за замовчуванням приймається значення параметрів **tbp** — спочатку LaTeX намагається розташувати таблицю вгорі сторінки, якщо

це не можливо — то внизу, а якщо й це неможливо — таблиця переноситься на окрему сторінку для «плаваючих» об'єктів.

Пакет **babel** з опцією **ukrainian** визначає ключове слово для формування заголовка таблиці „Табл“. Його можна переозначити на початку документу за допомогою команди:

\renewcommand {\tablename}{Таблиця}

Команда визначає для таблиці новий заголовок **Таблиця**.

Додатково управляти стилем заголовка таблиці можна за допомогою команди **\captionsetup[table]{параметри}**, якщо в преамбулі документу підключити пакет **caption**. Параметри задають через кому:

font=шрифт — шрифт, яким відображається заголовок таблиці;

labelsep=символ — тип розділового знаку між ключовим словом та заголовком таблиці.

Таблиця 5.3 — Типи розділових знаків, що можуть використовуватися під час формування заголовка

Символ	Розділовий знак
none	Розділовий знак відсутній
colon	Двокрапка
period	Крапка
space	Пробіл
quad	Пробіл величиною в чотири символи
newline	Ключове слово та заголовок знаходяться в різних рядках

Розташування заголовку та номера таблиці (вгорі чи внизу таблиці) визначається розташуванням команди **\caption** відносно оточення **tabular** (до чи після оточення).

Нижче наведено приклад створення таблиці з використанням основних управляючих конструкцій, що описано вище:

```
\documentclass[12pt,a4paper]{article}
\usepackage[cp1251]{inputenc}
\usepackage[ukrainian]{babel}
\usepackage{caption}
\captionsetup[table]{font=sf, labelsep=newline}
\begin{document}
\renewcommand {\tablename}{Таблиця}
Документ ілюструє основні прийоми створення таблиці
засобами видавничої системи.
Використано основні прийоми форматування наповнення
```

```
таблиці, її розміщення, проведення ліній та
формування заголовку таблиці
\begin{table}[t]
\begin{tabular}[c]{||c||c||p{300pt}||}
\hline \hline
\multicolumn{3}{|||c|||}{ \bf{Обов'язкові
параметри оточення tabular} } \\[10pt]
\hline
\hline
№ & \it{Параметр} & \it{Призначення} \\[10pt]
\hline
1 & l & Введені в комірку дані вирівнюються в
колонці по лівому краю \\
\hline
2 & r & Вирівнювання в колонці по правому краю \\
\hline
3 & c & Вирівнювання в колонці по центру \\
\hline
4 & | & Символ використовується для розділення
стовпців таблиці вертикальними лініями. \\
\cline{2-3}
& || & За необхідності проведення кілька
вертикальних ліній необхідно поставити відповідну
кількість символів. У цьому випадку буде проведено
дві вертикальні лінії \\[10pt]
\hline \hline
\end{tabular}
\caption{Таблиця основних обов'язкових параметрів}
\label{tb1}
\end{table}
\end{document}
```

5.3 Поворот таблиці

Якщо таблиця є широкою й не вміщується на аркуші, то її бажано повернути її на 90°. Це можна зробити за допомогою команди **rotatebox** з пакету **graphicx**.

```
\rotatebox{90}{
\begin{center}
```

```
\begin{tabular}[c]{|c|c|c|p{100pt}|}
\hline
% об'єднання комірок для формування заголовку
% таблиці
\multicolumn{4}{|c|}{Заголовок } \\[2pt]
\hline
1&2 &\multicolumn{2}{|c|}{Підзаголовок } \\
\hline
\multirow{2}{40pt}{текст1} & & &
\multirow{4}{40pt}{текст2} \\
\cline{2-3}
& \multirow{3}{50pt}{текст3} & & \\
\cline{1-1} \cline{3-3}
\multirow{2}{40pt}{текст4}& & & \\
\cline{3-3}
& & & \\
\hline
\end{tabular}
\end{center}
}
```

Результат компіляції:

Заголовок			
	1	2	Підзаголовок
	текст1		текст2
	текст4	текст3	

5.4 Таблиці, що займають кілька сторінок

Як вже було згадано, оточення **tabular** не розбивається за сторінками.

Якщо ми маємо довгу таблицю, варто скористатися пакетом **longtable** та оточенням **longtable**. Як і оточення **tabular**, воно має один обов'язковий параметр — преамбулу. Всередині оточення діють такі ж правила, що і в оточенні **tabular**. Різниця полягає в тому, що таблиця може бути розташована на кількох сторінках.

Крім того у довгих таблицях бажано, щоб шапка таблиці повторювалася на кожній сторінці. Для цього записуємо шапку звичайним чином і після цього запису ставимо команду **endfirsthead**, тим самим задаємо шапку на першій сторінці.

Для того, щоб ця шапка дублювалася на кожній сторінці, необхідно ще раз записати шапку таблиці і додати команду **endhead**.

Наведемо приклад такої таблиці

```
\begin{longtable}{|c|c|c|c|}
\caption{Назва} \\  \hline

\textbf{ichi}&\textbf{ni}&\textbf{san}&\textbf{shi} \\ \hline
\endfirsthead
\hline
ichi&ni&san&shi \\ \hline
\endhead

\hline

% Заповнення таблиці
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
```

```
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
```

```
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
\end{longtable}
```

Результат компіляції:

Таблиця 5.4: Назва

ichi	**ni**	**san**	**shi**
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

ichi	ni	san	shi
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

ichi	ni	san	shi
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

Часто довгі таблиці необхідно повернути на 90°, для цього використовують оточення **landscape**. Попередньо в преамбулі необхідно підключити пакет

lscape.

Наведемо приклад такої таблиці

```
\begin{landscape}
\begin{longtable}{|c|c|c|c|}
\caption{Назва} \\  \hline

\textbf{ichi}&\textbf{ni}&\textbf{san}&\textbf{shi} \\ \hline
\endfirsthead
\hline
ichi&ni&san&shi \\ \hline
\endhead

\hline

% Заповнення таблиці
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
```

```
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
```

```
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
123&34&78216&325 \\ \hline
\end{longtable}
\end{landscape}
```

Результат компіляції:

Таблиця 5.5: Назва

ichi	**ni**	**san**	**shi**
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

ichi	ni	san	shi
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

ichi	ni	san	shi
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

ichi	ni	san	shi
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325
123	34	78216	325

5.5 Дві таблиці на одній сторінці

У цьому випадку, як і для рисунків, варто змінити визначення й отримати результат, як це показано нижче.

```
\begin{table}[h]
\caption{Модельні коефіцієнти}
 \label{tabl:coef}
\centering
\begin{minipage}{6cm}
\begin{tabular}{|*{3}{c|}} \hline
\multicolumn{2}{|c|} {$s_r$}& $R^2$\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,01905110940&0,8292498465\\ \hline
\multicolumn{2}{|c|} {$a + bm$}& \\ \cline{1-2}
a&0,05767432657 &0,4116688009\\ \cline{1-2}
b&0,008974437432&\\ \hline
\multicolumn{2}{|c|} {$C + d \lg{m}$}& \\ \cline{1-2}
C&???? &???\\ \cline{1-2}
d&??????&\\ \hline
\multicolumn{3}{|c|}{Метод найменших квадратів}\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,0180744793&0,7396780733\\ \hline
\multicolumn{2}{|c|} {$a + bm$}& \\ \cline{1-2}
a&0,009876682433 &0,6864321253\\ \cline{1-2}
b&0,01741418940&\\ \hline
\multicolumn{2}{|c|} {$C + d \lg{m}$}& \\ \cline{1-2}
C&???? &???\\ \cline{1-2}
d&??????&\\ \hline
\multicolumn{3}{|c|}{Стандарт}\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,019&0,8239385400\\ \hline
 \end{tabular}
 \end{minipage}
\hspace{2cm}
\begin{minipage}{6cm}
\begin{tabular}{|*{3}{c|}} \hline
\multicolumn{2}{|c|} {$s_R$}& $R^2$\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,04604103104&0,5404680136\\ \hline
\multicolumn{2}{|c|} {$a + bm$}& \\ \cline{1-2}
a&0,04095968858 &0,3939303433\\ \cline{1-2}
```

```
b&0,03449940002&\\ \hline
\multicolumn{2}{|c|} {$C + d \lg{m}$}& \\ \cline{1-2}
C&???? &???\\ \cline{1-2}
d&??????&\\ \hline
\multicolumn{3}{|c|}{Метод найменших квадратів}\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,0474501062&0,5752975928\\ \hline
\multicolumn{2}{|c|} {$a + bm$}& \\ \cline{1-2}
a&-0,07689987608 &0,7049981991\\ \cline{1-2}
b&0,05259112588&\\ \hline
\multicolumn{2}{|c|} {$C + d \lg{m}$}& \\ \cline{1-2}
C&???? &???\\ \cline{1-2}
d&??????&\\ \hline
\multicolumn{3}{|c|}{Стандарт}\\ \hline
\multicolumn{2}{|c|} {$bm$}& \\ \cline{1-2}
b&0,04&0,4579805809\\ \hline
\multicolumn{2}{|c|} {$a + bm$}& \\ \cline{1-2}
a&0,086 &0,3391214686\\ \cline{1-2}
b&0,03&\\ \hline
\multicolumn{2}{|c|} {$C + d \lg{m}$}& \\ \cline{1-2}
C&0,078 &0,3091805392\\ \cline{1-2}
d&0,72&\\ \hline
 \end{tabular}
 \end{minipage}
 \end{table}
```

5.6 Розфарбовування таблиці

Зміна кольору комірок стовпчиків чи рядків таблиці не є надто простою. Частково її розв'язує пакет **color**, забезпечуючи єдиний інтерфейс з різноманітними вихідними пристроями, але повний розв'язок надає пакет **colortbl**, спеціально написаний для розфарбовки таблиць.

Пакет **colortbl** автоматично завантажує стандартні пакети **color, array** й може використовуватися сумісно з пакетами **longtable, dcolumn, hhline** тощо.

5.6.1 *Колір колонки*

Для зміни кольору тла окремої колонки він вводить команду
\columncolor[model]colorl-hangh-hang

Таблиця 5.6 — Модельні коефіцієнти

s_r		R^2
bm		
b	0,01905110940	0,8292498465
$a + bm$		
a	0,05767432657	0,4116688009
b	0,008974437432	
$C + d \lg m$		
C	????	???
d	??????	
Метод найменших квадратів		
bm		
b	0,0180744793	0,7396780733
$a + bm$		
a	0,009876682433	0,6864321253
b	0,01741418940	
$C + d \lg m$		
C	????	???
d	??????	
Стандарт		
bm		
b	0,019	0,8239385400

s_R		R^2
bm		
b	0,04604103104	0,5404680136
$a + bm$		
a	0,04095968858	0,3939303433
b	0,03449940002	
$C + d \lg m$		
C	????	???
d	??????	
Метод найменших квадратів		
bm		
b	0,0474501062	0,5752975928
$a + bm$		
a	-0,07689987608	0,7049981991
b	0,05259112588	
$C + d \lg m$		
C	????	???
d	??????	
Стандарт		
bm		
b	0,04	0,4579805809
$a + bm$		
a	0,086	0,3391214686
b	0,03	
$C + d \lg m$		
C	0,078	0,3091805392
d	0,72	

Її можна використовувати як специфікатор колонок >**{decl}**, всередині преамбули таблиці **cols** чи в описі колонок команди **multicolumn**. Ця команда визначає тло певної колонки чи комірки. Аргументи **model** (модель кольору) та **color** (колір) мають той же сенс, що і в командах, які визначено в пакеті **color**. Необов'язкові аргументи **l-hang** та **r-hang** визначають, як далеко ліворуч чи праворуч від колонки сягає кольоровий фон. За замовчуванням вони дорівнюють величині **tabcolsep** (у випадку процедури **tabular**) чи **arraycolsep** (у випадку процедури **array**), тобто колір повністю заповнює колонку.

Розфарбуємо таблицю:

```
\setlength{\extrarowheight}{3pt}
\begin{tabular}[t]{|c|%
>{\columncolor{yellow}\bfseries}l|%
>{\columncolor{green}\itshape}c|}
\hline \multicolumn{3}{|>{\columncolor{blue}}c|}
{\color{white}{\bfseries Колір}} \\
\hline A & B & C \\
```

```
\hline 100 & 10 & 1 \\
\hline
\end{tabular}
```

Результат компіляції:

Колір		
A	**B**	*C*
100	**10**	*1*

5.6.2 *Колір рядка*

Щоб змінити колір тла всього рядка, передбачено команду
\rowcolor[model]{color}{l-hang}{h-hang}

Її слід розташовувати на початку рядка.

```
\begin{tabular}[t]{|c|>{\bfseries}l|>{\itshape}c|}
\rowcolor[blue]{0.7} \hline A & B & C \\
\rowcolor[yellow]{0.7}\hline 100 & 10 & 1 \\
\rowcolor[blue]{0.85} \hline 200 & 20 & 2 \\
\rowcolor[yellow]{0.85}\hline 300 & 30 & 3 \\
\rowcolor[blue]{0.85} \hline 400 & 40 & 3 \\
\hline
\end{tabular}
```

Результат компіляції:

A	**B**	*C*
100	**10**	*1*
200	**20**	*2*
300	**30**	*3*
400	**40**	*3*

На перетині кольорових колонок та рядків перевагу мають установки команди фарбування рядка **\rowcolor**, але їх дія обмежена поточним рядком.

5.6.3 *Колір комірки*

Щоб змінити колір тла комірки, передбачено команду
\cellcolor[model]{color}{вміст комірки}.
Її слід розташовувати на початку комірки.

```
\begin{center}
\begin{tabular}{|*{6}{c|}} \hline
Рівень $j$& Поодинокий & Поодинокий &Подвійний &Подвійний
 & Значення \\
&  нижній&  верхній&нижній& верхній&
\\ \hline
1& 1,229192343& 1,807128392& 0,54097& 0,301594&  \\ \cline{1-5}
2& 0,898944377& 2,088992941& 0,701836&
\cellcolor{red}{0,107273}&\\ \cline{1-5}
3& 1,668578793& 1,585944415& 0,416044& 0,455331&\\ \cline{1-5}
4& 0,936924053& 2,101748552& 0,686264& 0,121325&\\ \hline
К.в& 2,126& 2,126 &0,1101& 0,1101& 5\%\\ \hline
С.в& 2,274& 2,274& 0,0563& 0,0563& 1\%\\ \hline

\end{tabular}
\end{center}
```

Результат компіляції:

Рівень j	Поодинокий нижній	Поодинокий верхній	Подвійний нижній	Подвійний верхній	Значення
1	1,229192343	1,807128392	0,54097	0,301594	
2	0,898944377	2,088992941	0,701836	0,107273	
3	1,668578793	1,585944415	0,416044	0,455331	
4	0,936924053	2,101748552	0,686264	0,121325	
К.в	2,126	2,126	0,1101	0,1101	5%
С.в	2,274	2,274	0,0563	0,0563	1%

5.6.4 *Колір рамки*

Кольори рамки і роздільних ліній між комірками таблиці, що створено командами **hline,** **cline,** **vline** чи символом | в преамбулі таблиці **cols**, змінює декларація **arrayrulecolor[model]{clr}**.

Вона діє на всі наступні лінії й може розташовуватися поза таблицею, на початку рядка чи всередині специфікації >**{decl}** в преамбулі таблиці **cols**. Її вплив не обмежується поточною областю дії декларацій, але не скасовує специфікації, що задано в **cols**.

```
\setlength{\arrayrulewidth}{2pt}
\arrayrulecolor{cyan}
\begin{tabular}[t]{|c|>{\bfseries}l|>{\itshape}c|}
```

```
\arrayrulecolor{cyan} \hline A & B & C \\
\arrayrulecolor{cyan} \hline 100 & 10 & 1 \\
\hline 200 & 20 & 2 \\
\arrayrulecolor{cyan} \hline
\end{tabular}
```

Результат компіляції:

A	**B**	*C*
100	**10**	*1*
200	**20**	*2*

Проміжок між подвійними лініями, що утворено двома символами || в преамбулі таблиці **cols** чи двома командами **hline** в тілі таблиці, можна зафарбувати за допомогою декларації **doublerulecolor[model]{color}**.

Якщо фарбування ліній здійснюється одночасно з фарбуванням тла в комірках, часткові лінії, що утворено командою **cline**, набувають кольору тла, стаючи невидимими.

6 Переліки (списки)

LaTeX пропонує три процедури (оточення) для складання списків:
`\begin{itemize} item-list \end{itemize}`
`\begin{enumerate} item-list \end{enumerate}`
`\begin{description} item-list \end{description}`

- Оточення **itemize** призначено для найпростіших переліків.
- Оточення **enumerate** призначено для нумерованих переліків.
- Оточення **description** призначено для переліків, в яких кожен пункт має заголовок (наприклад, словарних статей чи інших описів).

Тіло процедур item-list складається з послідовності записів, що починаються з команди `\item[mark]`.

Команда `\item` позначає запис міткою **mark**. Якщо необов'язковий аргумент (разом з квадратними дужками) є відсутнім, то використовується "мітка за замовчуванням". Вигляд "мітки за замовчанням" залежить як від процедури, що використовують, так і від рівня вкладеності списку в інші процедури складання списків. Дозволяються чотири рівня вкладеності [2, 7].

Відстань між записами у переліках можна змінити командою **itemsep** = <**відстань**>. Параметр «відстань» визначається у одиницях довжини, що використовують в LaTeX. Цей параметр може набувати як додатних, так і від'ємних значень.

Зсув записів у переліках можна визначити командою **itemindent** = <**відстань**>.

Розглянемо послідовно, як працюють згадані оточення [7].

6.1 Процедура itemize

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

```
\item Другий запис першого рівня.
\end{itemize}
```

Результат дії процедури `\itemize` показує, як помічаються записи «за замовчуванням»:

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

Щоб змінити мітку, треба описати її у необов'язковому аргументі команди `\item`:

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

$ Перший запис

Другий запис

Мітки в цьому прикладі притискаються до лівого краю поля. Змінити це правило можна за допомогою команди `\hfill`.
Наприклад:`\item[\$\hfill]`.

Мітки, що використовує процедура **itemize** за замовчуванням на відповідному рівні вкладеності, здійснюються командами \labelitemi, \labelitemii, \labelitemiii, \labelitemiv.

Перевизначивши їх за допомогою `\renewcommand`, можна замінити мітки відразу в усіх записах:

```
\begin{itemize}
\renewcommand{\labelitemi}{\#}
\item Перший запис
\item Другий запис
\end{itemize}
```

Перший запис

Другий запис

Тут команду `\labelitemi` перевизначено всередині процедури **itemize**. Якщо її змінити до початку процедури, то нове визначення буде використано в подальшому.

6.2 Процедура enumerate

Процедура enumerate нумерує записи. Вона також може бути вкладена в інші процедури складання списків. Відповідно до чотирьох рівнів вкладання використовується чотири лічильники: **enumi, enumii, enumiii, enumiv**. Кожен запис збільшує значення лічильника відповідного рівня на одиницю. За наявності необов'язкового аргументу в команді `\item[mark]` значення лічильника не збільшується. У прикладі

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

можна побачити як нумеруються записи всіх можливих рівнів за замовчуванням:

1. Перший запис першого рівня.
    - Перший запис другого рівня.
        - Третій рівень.
            * Четвертий рівень.
    - Другий запис другого рівня.
2. Другий запис першого рівня.

«Мітки за замовчуванням» друкують команди **\labelenumi, \labelenumii, \labelenumiii, \labelenumiv**.

Наведемо ще один приклад:

```
\begin{enumerate}
\renewcommand{\theenumi}{\Asbuk{enumi}}
\item Перший запис
\item Другий запис
\end{enumerate}
```

А. Перший запис

Б. Другий запис

В результаті перевизначення **\theenumi** команда **\labelenumi** (яку за замовчуванням визначено як \theenumi) тепер друкує номера першого рівня великими кириличними літерами.

6.3 Процедура description

В процедурі **description** «мітка за замовчуванням» відсутня. Тому в команді **\item[mark]** має бути необов'язковий аргумент. За замовчанням мітку mark буде надруковано напівжирним шрифтом. Шрифт мітки можна змінити, поставивши необхідну декларацію безпосередньо в аргумент команди \item.

Наведемо простий приклад:

```
\begin{description}
\item[Апроксимація] --- наближене зображення одних математичних
об'єктів іншими.
\item[Інтерполяція] --- побудова для функції $f(x)$ за її значеннями
на дискретній множині $S$ деякої нової функції $H(x)$, яку
визначено на більш широкій множині, такій, щоб $H(x) = f(x)$ за
всіх   $x \in S$.
\item[\itshape Екстраполяція]
--- наближене визначення значень функції $f(x)$ у точках $x$, що
розташовано поза відрізком $[x_0, x_n]$, за її значеннями у точках
$x_0 < x_1 < \ldots < x_n$.
\end{description}
```

Результат компіляції:

Апроксимація — наближене зображення одних математичних об'єктів іншими.

Інтерполяція — побудова для функції $f(x)$ за її значеннями на дискретній множині S деякої нової функції $H(x)$, яку визначено на більш широкій множині, такій, щоб $H(x) = f(x)$ за всіх $x \in S$.

Екстраполяція — наближене визначення значень функції $f(x)$ у точках x, що розташовано поза відрізком $[x_0, x_n]$, за її значеннями у точках $x_0 < x_1 < \ldots < x_n$.

У списках, які складено за процедурами **itemize, enumerate, description**, відстань між записами за необхідністю можна змінювати, використовуючи команду **\itemsep = distance**. Причому параметр **distance** може мати як від'ємне, так і додатне значення.

6.4 Список використаних джерел

Створення списку використаних джерел та оформлення на нього посилань є трудомісткою процедурою під час підготовки наукових публікацій. Посилання на список використаних джерел можна утворити двома способа- ми:

1. за допомогою оточення thebibliography;

2. використовуючи допоміжний bib-файл.

Список використаних джерел з використанням оточення **thebibliography** може розташовуватися у будь-якому місці документу.

Створення списку використаних джерел відбувається таким чином:

```
\begin{thebibliography}{99}
%Джерело № 1
\bibitem{Morishita} \textit{Morishita Y.} CAD of  microwave tubes.
/ Y. Morishita // Теребідзьон гаккай ші. --- 1978.,  Vol. 32,
N 3.--- C. 182 - 188. Яп.
%Джерело № 2
\bibitem{nik}\textit{Nikitenko O. M.} Distribution of
electrostatic potential in crossed-field system with
complex electrodes' configuration
/ O. M. Nikitenko // Journal of Microwaves and Optoelectronics,
Vol. 2, No. 2, December 2000, P. 1 - 9
%Джерело №3
\bibitem{silin} \textit{Силин Р. А.} Замедляющие системы.
/ Р. А. Силин, В.П. Сазонов, Москва :  Сов. радио, 1966. 632 с.
%Джерело №4
\bibitem{samson} \textit{Самсонов Д. Е.} Основы расчета и
проектирования многорезонаторных магнетронов.
/ Д. Е. Самсонов, Москва :  Сов. радио, 1966. 234~с.
%Джерело №5
\bibitem{koval} \textit{В. Ф. Коваленко} Введение в
электронику сверхвысоких частот.
/ В. Ф. Коваленко --- Москва :  Сов. радио, 1955, 344 с.
%Джерело №6
\bibitem{taran} \textit{Тараненко З. И.} Замедляющие системы.
/ З. И. Тараненко, Я. К. Трохименко   - Киев: Техніка, 1965. -
307 с.
\end{thebibliography}
```

Результат компіляції:

Література

1. *Morishita Y.* CAD of microwave tubes. / Y. Morishita // Теребідзьон гаккай ші. — 1978., Vol. 32, N 3.— С. 182 – 188. Яп.

2. *Nikitenko O. M.* Distribution of electrostatic potential in crossed–field system with complex electrodes' configuration / O. M. Nikitenko // Journal of Microwaves and Optoelectronics, Vol. 2, No. 2, December 2000, P. 1 – 9

3. *Силин Р. А.* Замедляющие системы. / Р. А. Силин, В.П. Сазонов, Москва : Сов. радио, 1966. 632 с.

4. *Самсонов Д. Е.* Основы расчета и проектирования многорезонаторных магнетронов. / Д. Е. Самсонов, Москва : Сов. радио, 1966. 234 с.

5. *В. Ф. Коваленко* Введение в электронику сверхвысоких частот. / В. Ф. Коваленко — Москва : Сов. радио, 1955, 344 с.

6. *Тараненко З. И.* Замедляющие системы. / З. И. Тараненко, Я. К. Трохименко – Киев: Техніка, 1965. – 307 с.

Заголовок до списку використаних джерел генерується відповідно до стилю документа. За необхідності його можна змінити в тілі документа. Для стилю **article** за допомогою команди **\renewcommand {\refname} {Перелік джерел посилань}**, для стилів **book** та **report** за допомогою команди **\renewcommand {\bibname} {Перелік посилань}**.

Основними параметрами під час використання цього оточення є:

{99} — величина відступу списку літератури від лівої границі документу. В прикладі задано відступ 2 символи;

\bibitem{silin} — починає кожен елемент списку літератури. Як обов'язковий параметр виступає мітка (певний текст), що використовується для організації автоматичної генерації посилань на це джерело;

\textit — обов'язковий параметр, що задає прізвище автора публікації.

Команди **\bibitem** формують бібліографічні посилання, на які можна посилатися з допомогою команди **\cite**:

```
Удосконалення технічних прийомів проектування приладів НВЧ
з допомогою ЕОМ дозволяє різко скоротити витрати на їх
виробництво \cite{Morishita}.

Раніше  \cite{nik} було розглянуто розподіл електростатичного
потенціалу в резонансних системах приладів зі схрещеними
полями.
```

```
Метод еквівалентних схем, згідно з яким передбачено розв'язання
рівнянь Кірхгофа для замкненого ланцюжка резонаторів
\cite{silin, samson, koval, taran}.
```

Результат компіляції:

Удосконалення технічних прийомів проектування приладів НВЧ з допомогою ЕОМ дозволяє різко скоротити витрати на їх виробництво [1].

Раніше [2] було розглянуто розподіл електростатичного потенціалу в резонансних системах приладів зі схрещеними полями.

Метод еквівалентних схем, згідно з яким передбачено розв'язання рівнянь Кірхгофа для замкненого ланцюжка резонаторів [3–6].

Як і у випадку з перехресними посиланнями, для правильного відображення бібліографічних посилань, вихідний документ необхідно скомпілювати двічі.

Створення списку літератури з використанням бібліографічної бази даних BibTEX дозволяє:

- скоротити час формування великого списку літератури або списку, основну частину якого інтенсивно використовують,
- створити власну бібліографічну базу даних автора, що періодично оновлюють. Джерела літератури у цьому випадку додають не в сам LATEX-документ, а у допоміжний bib-файл, що дозволяє використовувати створену бібліографічну базу для багатьох документів.

Бібліографічні бази мають визначену структуру. База зберігається у звичайному текстовому файлі з розширенням **bib** (у ньому перелічуються всі бібліографічні посилання автора). В процесі роботи з файлом **fn.tex** (**fn** — ім'я файлу) програма BibTEX видобуває з бази даних джерела літератури, які вказані в тексті файлу.

З 1985 року BibTEX є основним інструментом для створення бібліографії у видавничій системі LATEX, внаслідок тісної інтеграції з цією системою і простоти у використанні.

Розглянемо основні пакети бібліографічних стилів інструмента BibTEX які є актуальними для створення математичних текстів. Стиль у термінах BibTEX визначається **bst**-файлом [3, 4].

Очевидна перевага роботи з базами даних полягає в тому, що бібліографічна інформація набирається тільки один раз, під час занесення її в базу даних. Кожен пункт бібліографії оформлюється як окремий запис з унікальною міткою. Вставка цього пункту до списку літератури будь-якого документу з посилан-

ням на нього в тексті або без посилання здійснюється простими за форматом командами, що вказують цю мітку.

1. Посилання в основному документі з'являються у міру згадування й завжди нумеруються правильно.
2. Існує можливість використання однієї бази у багатьох документах.
3. Якщо в описі посилання на літературу виявилася помилка, її можна виправити в базі, тим самим вона буде виправлена відразу в усіх документах, які посилаються на базу.
4. Існує простий пошук в базі для з'ясування ключа цитування (аргументу **\cite{mitkacite}**).
5. Можливий імпорт і експорт в інші бібліографічні формати.

Для формування і роботи з особистою базою даних необхідна наявність файлів з розширеннями **bib** і **bst**. Перед початком необхідно створити й заповнити файл з розширенням **bib**.

Кожен запис має відповідати певному типу: стаття в журналі, стаття для конференції, книжка, атестаційна робота тощо.

Всі записи файлу **fn.bbl** повинні мати визначену структуру. Записи різних типів в базі даних мають різний набір полів, причому поля поділяються на три класи: обов'язкове поле, необов'язкове поле і поле, яке ігнорують.

Обов'язкове відсутність поля викличе повідомлення про помилку.

Необов'язкове інформація в цьому полі буде використана, якщо вона міститься в записі, але її відсутність не викличе повідомлення про помилку чи проблем під час друку.

Ігнороване якщо BibTEX не розпізнає поле як обов'язкове або необов'язкове, то це поле ігнорується.

Дані можна набирати вручну у будь-якому текстовому редакторі, а можна поставити програму-оболонку для роботи з BibTEXом.

Крім того з багатьох систем управління бібілографічною інформацією та наукометричних баз даних можна безпосередньо отримати вже сформований файл BibTEX.

Система управління бібліографічною інформацією **EndNote** експортує в BibTEX.

Бібліографічний менеджер **Mendeley** експортує/імпортує в BibTEX.

Наукометрична база даних **WebOfScience** експортує в **EndNote**.

Наукометрична база даних **Scopus** експортує в BibTEX.

Наукометрична база даних **Google Scholar** експортує в BibTEX.

ORCID експортує/імпортує в BibTEX.

В преамбулі наукової праці слід розташувати команду **\bibliographystyle{бібліографічний стиль}**, аргумент якої — це ім'я файлу бібліографічного стилю з розширенням **bst**.

Існує чотири стандартних стилі: **plain, unsrt, alpha, abbrv**. Крім стандартних стилів розроблено ще багато стилів, які часто використовують.

Складання списку літератури здійснюється за допомогою двох описових команд, одна з яких визначає стиль бібліографічного списку, а друга — місце розташування списку і бази даних, які використовують.

Розглянемо формат файлу бібліографічних даних.

База даних — це файл, що розділений на записи. Кожен запис зберігає бібліографічну інформацію однієї роботи.

Файл бази даних для LATEX'у створюють у будь-якому текстовому редакторі.

Імена файлів бібліографічних баз даних повинні мати розширення **bib**.

Запис починається символом @. Потім розташовують позначку типу роботи, яка визначає запис — article (стаття), book (книжка) тощо.

За типом запису у фігурних дужках або подвійних лапках подається власна інформація про роботу. Ця інформація складається з ідентифікатора і полів.

Ідентифікатор є унікальною міткою запису. Саме його мають використовувати як обов'язковий аргумент команди **\cite** або **\nocite** під час посилання на певну роботу.

Кожне поле складається з імені, знаку рівності й текстового рядка, обмежувачем якого є фігурні дужки або подвійні лапки. Якщо текст повністю складається з цифр, дужки або подвійні лапки можуть бути опущені.

У табл. 6.4 наведено типи записів, які найчастіше використовують

Таблиця 6.4 – Типи записів

Типи записів	**Обов'язкові поля**		**Необов'язкові поля**	
article	author	(автор)	volume	(том)
журнальна	title	(заголовок)	number	(номер)
стаття	journal	(журнал)	pages	(сторінки)
	year	(рік)	month	(місяць)
			note	(примітка)
book	author	(автор) або	volume	(том)
книжка	editor	(редактор)	series	(серія)
	title	(заголовок)	address	(адреса)
	publisher	(видавництво)	edition	(видання)
	year	(рік)	month	(місяць)

			note	(примітка)
booklet буклет	title	(заголовок)	author howpublished address month year note	(автор) (спосіб видання) (адреса) (місяць) (рік) (примітка)
conference конференція **inproceeding** стаття в працях конференції	author title booktitle year	(автор) (заголовок) (загальний заголовок) (рік)	edition organization pages publisher address month note	(видання) (організація-спонсор) (сторінки) (видавництво) (адреса) (місяць) (примітка)
inbook витримка з книжки	author editor title publisher chapter year pages	(автор) або (редактор) (заголовок) (видавництво) (глава) (рік) і/або (сторінки)	volume series address edition month note	(том) (серія) (адреса) (видання) (місяць) (примітка)
incollection витримка з книжки із заголовком	author title booktitle publisher year	(автор) (заголовок) (заголовок книжки) (видавництво) (рік)	editor chapter pages address month note	(редактор) (глава) (сторінки) (адреса) (місяць) (примітка)
manual технічна документація	title	(заголовок)	author organization edition year month	(автор) (організація) (видання) (рік) (місяць)

			note	(примітка)
masterthesis магістерська дисертація **phdthesis** докторська дисертація	author title school year	(автор) (заголовок) (навчальний заклад) (рік)	address month note	(адреса) (місяць) (примітка)
misc різне howpublished			author title year month note	(автор) (заголовок) (спосіб видання) (рік) (місяць) (примітка)
proceedings праці конференції	title year	(заголовок) (рік)	editor publisher organization address month note	(редактор) (видавництво) (оргацізація- спонсор) (алреса) (місяць) (примітка)
techreport звіт	author title institution year	(автор) (заголовок) (установа) (рік)	type number month note	(тип звіту) (номер) (місяць) (примітка)
unpublished не неопублікований офіційно документ	author title note	(автор) (заголовок) (примітка)	month year	(місяць) (рік)

Приклад фрагменту файлу **bib**:

```
@article{Prots,
author = {И. И. Проценко},
title = {Приближенный метод расчета уравнений гармоник в приборах
```

```
М-типа},
journal = {Электронная техника. Сер.Электроника СВЧ},
year = {1970},
number = {3},
pages = {19-25},
language = {russian},
}
%LaTeX
@conference{Gryshchenko,
author = {Т. Б. Грищенко and Ж. В. Дейнеко and О. М. Нікітенко},
title = {Використання системи LaTeX під час підготовки наукових
публікацій},
booktitle = {Поліграфічні, мультимедійні та web-технології: тези доп.
IV Міжнар. наук.-техн. конф. (14-17 травня 2019, м. Харків)},
year = {2019},
publisher = {«Друкарня Мадрид»},
volume ={1},
address = {Харків},
pages = {96 - 99},
language = {ukrainian}
}
@conference{Korbut,
author = {О. Г. Корбут},
title = {Електронний підручник як елемент освітнього середовища},
booktitle = {Науково-практична конференція <<Новітні освітні
технології>>},
language = {ukrainian}
}
@book{Znamen,
author = {О. В. Знаменская and С. В. Знаменский and Д. Е. Лейнартас
and В. М. Трутнев},
title = {Математическая типография Курс лекций},
year = {2008},
address = {Красноярск},
numpages = {421},
language = {russian}
}
@article{Lysenko,
author = {С. М. Лисенко and А. Ф. Крищук and Ю. П. Дзюбак},
title = {Дослідження переваг застосування LATEX при оформленні
```

```
наукових праць},
journal = {Вісник Хмельницького національного університету},
year = {2012},
number = {5},
pages = {225-234},
language = {ukrainian}
}
@book{Podoshv,
author = {Ю.Г. Подошвелев},
title = {Система LaTeX Інтерактивний електронний посібник},
year = {2016},
address = {Полтава},
numpages = {189},
language = {ukrainian}
}
@book{Knuth,
author = {Donald E. Knuth},
title = {The TeXbook },
year = {2016},
publisher = {Addison-Wesley Professional},
address = {Полтава},
numpages = {496},
language = {english}
}
@book{Zhelez,
author = {Б. Е. Железовский and Э. В. Комьянов},
title = {Многочастотные режимы в приборах СВЧ},
year = {1978},
publisher = {Радио и связь},
address = {Москва},
numpages = {256},
language = {russian}
}
```

Фрагмент тексту

```
До недоліків електронного підручника можна віднести:
\begin{itemize}
\item сприйняття з екрана комп'ютера текстової інформації набагато
менш зручно і ефективно, ніж читання книги;
\item більш висока вартість у порівнянні із звичайним паперовим
підручником \cite{Korbut}.
```

```
\end{itemize}
\LaTeX\ є набором програм і продовженням оригінальної програми TeX,
створеної видатним американським математиком і програмістом
Дональдом Кнутом \cite{Knuth}.
\LaTeX\ --- це комп'ютерна видавнича система, веб-орієнтована
система подання і рецензування рукописів, що містить набір
фундаментальних сервісів і функцій.
Основне її призначення - підготовка наукових документів.
Ця система була розроблена на базі системи \TeX{}
\cite{Knuth, Lysenko}.
За останні роки в багатьох серйозних наукових журналах вимагають від
авторів підготовку статей саме у форматі \LaTeX{}.
Видавничу систему \LaTeX\ прийнято як стандарт більшістю відомих
науково-технічних видавництв світу, зокрема: Elsevier,
Springer-Verlag, John Wiley \& Sons, Kluwer, Addison Wesley Longman,
AMS, SIAM, Мир, ТВП, Факториал тощо.
Тексти, що підготовлені за допомогою видавничої системи \LaTeX\,
мають високу якість оформлення і можуть використовуватися більшістю
сучасних операційних систем \cite{Gryshchenko}.
Для створення електронних презентацій в \LaTeX\ існує низка
спеціальних пакетів, таких як Beamer, Prosper, PowerDot, PdfScreen,
PPower4, PdfSlide тощо.
При цьому майже всі прийоми оформлення тексту в \LaTeX\
працюватимуть і в презентаціях, а отже зі статті чи звіту pdf-файл
презентації створити дуже просто \cite{Znamen}.
Таким чином, всі команди, що контролюють розміри сторінки
перевизначаються з метою формування динамічних розмірів заповненості
екрану \cite{Podoshv}.
Використовуючи пакет PdfScreen, можна легко досягти вище зазначених
результатів. Крім того цей пакет дозволяє разом з пакетом exerquiz
створювати професійні інтерактивні тести \cite{Podoshv}.

\bibliographystyle{gost2008} %Стиль бібліографічних посилань BibTeX

\newpage
\renewcommand{\refname}{ Перелік джерел посилань (gost2008)}
\bibliography{rudenko}
```

Результат компіляції для стилю **gost2008**:

Перелік джерел посилань (gost2008)

[1] Корбут, О. Г. Електронний підручник як елемент освітнього середовища

[Текст] / О. Г. Корбут // Науково-практична конференція «Новітні освітні технології».-[S. l. : s. n.].
[2] Knuth, D. E. The TeXbook [Text] / Donald E. Knuth. - Washington : Addison-Wesley Professional, 2016.
[3] Лисенко, С. М. Дослідження переваг застосування latex при оформленні наукових праць [Текст] / С. М. Лисенко, А. Ф. Крищук, Ю. П. Дзюбак // Вісник Хмельницького національного університету.-2012.-№ 5.-С. 225-234.
[4] Грищенко, Т. Б. Використання системи latex під час підготовки наукових публікацій [Текст] / Т. Б. Грищенко, Ж. В. Дейнеко, О. М. Нікітенко // Поліграфічні, мультимедійні та web-технології: тези доп. IV Міжнар. наук.-техн. конф. (14-17 травня 2019, м. Харків). - Т. 1. - Харків : «Друкарня Мадрид», 2019.-С. 96 - 99.
[5] Математическая типография Курс лекций [Текст] / О. В. Знаменская, С. В. Знаменский, Д. Е. Лейнартас, В. М. Трутнев. - Красноярск : [б. и.], 2008.
[6] Ю. Г., П. Система LaTeX Інтерактивний електронний посібник [Текст] / Подошвелев Ю. Г.-Полтава : [s. n.], 2016.

Таке застосування **BibTeX** не завжди адекватно створює список використаних джерел, особливо, якщо ці джерела містять кириличні символи.

Для уникнення цього варто застосувати такий набір команд:

pdflatex fn.tex
bibtex8 -H -c cp1251 fn.aux
pdflatex fn.tex
pdflatex fn.tex

6.5 Створення змісту роботи

LaTeX автоматично збирає інформацію для створення змісту документу й зберігає її у спеціальному файлі з ім'ям компільованого файлу та розширенням **.toc**. Для виводу змісту необхідно дати команду: **tableofcontents**

Зміст, що генерується, завжди відстає від реального стану справ, тому документ необхідно додатково скомпілювати принаймні ще один раз.

Приклад формування змісту документу із дещо зміненими стильовими параметрами та додатковим розбиттям документу на окремі сторінки:

```
%додається лічильник сторінок, щоб зі змісту потрапляти в розділи
\stepcounter{page}
% Визначає рівень змісту (будуть тільки \section)
% За замовчуванням зміст формується, включаючи \subsubsection
\setcounter{tocdepth}{1}
```

```
%переозначення заголовку

\renewcommand {\contentsname}{\centering Зміст лекції}
%Тут команда \centering вказує, що "Зміст лекції" буде розташовано
% посередині рядка
%формування змісту
\tableofcontents


%Для тих розділів, які позначені * додається
\addcontentsline{toc}{section}{\numberline{}Вступ}
```

Перша команда ілюструє можливість зміни стандартного заголовку для змісту.

7 Математика

Видавничий пакет LaTeX було спочатку оптимізовано для набирання математичних публікацій.

Для формул, що записують всередині тексту (математичний режим), використовують обмежувачі \(...\) або $ ... $. Всередині тексту краще не використовувати багаторядкові конструкції, такі, як дроби і матриці.

Для формул, що записують як окремий абзац (виділений математичний режим), використовують обмежувачі \[...\] або $$ ... $$.

Для більшої виразності формули обводять "рамкою". Це можна зробити таким чином:

```
Розв'язання квадратного рівняння \(ax^2+bx+c=0\):
$$\boxed{x_{1,2}=\frac{-b\pm\sqrt{b^2-4ac}}{2a}}$$
```

Результат компіляції:

Розв'язання квадратного рівняння $ax^2 + bx + c = 0$:

$$\boxed{x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}}$$

Складніші і різноманітніші математичні записи можна отримати у виділеному математичному режимі, використовуючи певні оточення. У цих оточеннях формули автоматично нумеруються і можна використовувати посилальний механізм.

Якщо необхідно змінити спосіб нумерації формул, замінивши наскрізну нумерацію, що використовується за замовчуванням, на нумерацію всередині рубрики, необхідно застосувати команду **\numberwithin{equation}{section}**.

Для написання багаторядкових математичних формул в LaTeX 2_ε використовують операторні дужки **align** і **gather**. Рядки відокремлюють один від одного командою \\ і кожен рядок нумерується автомтично. Відміна нумерації в рядку здійснюється командою **\nonumber** в кінці рядка. Для повної відміни нумерації необхідно скористатися версією оточення з зірочкою: **align***. Можна вирівнювати формулу за певним символом, встановлюючи перед ними знак &. За замовчуванням формули вирівнюються за лівою межею. Оточення **gather** відрізняється від **align** центруванням формул і неможливістю вирівнювання за символами &. Оточення **multline** дозволяє розбивати довгу формулу на рядки, заповнюючи рядок документу на всю довжину так, як заповнює рядок оточення **flalign**, в іншому схоже на оточення **align**.

Команди вводу математичних операторів, символів зі значками й літер грецької абетки наведено в табл. 7.1—7.3. Символи грецької абетки у формулах набираються курсивом. Прямі грецькі символи можна набирати, використовуючи пакет розширення **upgreek**. Група так званих «великих» математичних

Таблиця 7.1 — Набір математичних символів

$+$	+	$-$	-	$=$	=	$\neq$	\ne, \neq
$\pm$	\pm	$\mp$	\mp	$\approx$	\approx	$\equiv$	\equiv
$<$	<	$>$	>	$\sim$	\sim	$\simeq$	\simeq
$\leq$	\le	$\geq$	\ge	$\ll$	\ll	$\gg$	\gg
$\times$	\times	$\div$	\div	$\cdot$	\cdot	$\doteq$	\doteq
$\cdots$	\cdots	$\ldots$	\ldots	$\vdots$	\vdots	$\ddots$	\ddots
(	(	)	)	[	[	]	]
$\{$	\lbrace	$\}$	\rbrace	$\langle$	\langle	$\rangle$	\rangle
$\lceil$	\lceil	$\rceil$	\rceil	$\lfloor$	\lfloor	$\rfloor$	\rfloor
:	\colon, :	$\circ$	\circ	$\parallel$	\parallel, |	\	\backslash, \
|	\vert, \mid, |	$\bullet$	\bullet	$\star$	\star	$*$	\ast, *
$\clubsuit$	\clubsuit	$\hbar$	\hbar	$\heartsuit$	\heartsuit	$\spadesuit$	\spadesuit
$\diamondsuit$	\diamondsuit	ℓ	\ell	$\imath$	\imath	$\jmath$	\jmath
$\aleph$	\aleph	$\Re$	\Re	$\Im$	\Im	$\wp$	\wp
∞	\infty	∂	\partial	∇	\nabla	$\ni$	\ni, \owns
$\forall$	\forall	$\exists$	\exists	$\neg$	\neg, \lnot	$\emptyset$	\emptyset
$\surd$	\surd	$\angle$	\angle	$\triangle$	\triangle	$\frown$	\frown
$\prime$	\prime	$\dagger$	\dagger	$\ddagger$	\ddagger	$\odot$	\odot
$\oplus$	\oplus	$\ominus$	\ominus	$\otimes$	\otimes	$\oslash$	\oslash
$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp	$\wr$	\wr
$\top$	\top	$\cup$	\cup	$\vdash$	\vdash	$\dashv$	\dashv
$\bot$	\bot, \perp	$\cap$	\cap	$\sqcup$	\sqcup	$\triangleright$	\triangleright
$\vee$	\vee, \lor	$\prec$	\prec	$\sqcap$	\sqcap	$\triangleleft$	\triangleleft
$\wedge$	\wedge, \land	$\succ$	\succ	$\preceq$	\preceq	$\succeq$	\succeq
$\uplus$	\uplus	$\supset$	\supset	$\subseteq$	\subseteq	$\supseteq$	\supseteq
$\models$	\models	$\subset$	\subset	$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq
$\propto$	\propto	$\asymp$	\asymp	$\cong$	\cong	$\amalg$	\amalg
$\in$	\in	$\notin$	\notin	$\smile$	\smile	$\bowtie$	\bowtie

операторів наведена в табл. 7.4. Стрілки в математичних виразах вводяться командами з табл. 7.5.

Таблиця 7.2 — Запис символів з акцентами в математичному режимі

x'	x'	x''	x''	$\bar{x}$	\bar x	$\hat{x}$	\hat x
$\vec{x}$	\vec x	$\dot{x}$	\dot x	$\ddot{x}$	\ddot x	$\check{x}$	\check x
$\grave{x}$	\grave x	$\acute{x}$	\acute x	$\tilde{x}$	\tilde x	$\breve{x}$	\breve x
$\dddot{x}$	\dddot x	$\mathring{x}$	\mathring				

Запис верхніх та нижніх індексів величин здійснюється тільки в математичному режимі за допомогою операторів піднесення «^» та пониження «_».

Таблиця 7.3 — Грецька абетка

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

Дріб записують двома способами: **\frac{чисельник}{знаменник}** або **{чисельник} \over {знаменник}**. В математичному режимі визначено найрозповсюдженіші функції, такі як **\sin, \cos, \tan, \exp, \ln**. Крім того, визначена спеціальна функція **\sqrt** для виводу на друкування кореня n-ого ступеня від будь-якого виразу і може бути записана таким чином:

```
\[\sqrt[\mbox\(n\)]{\alpha \cdot \frac{x}{y}}\]
```

або

```
$$\sqrt[\mbox{$n$}]{\alpha \cdot \frac{x}{y}}$$
```

Результат компіляції:

$$\sqrt[n]{\alpha \cdot \frac{x}{y}}$$

Для того, щоб набрати границю, слід використати таку конструкцію:

\lim_{x\to 0} \frac{\sin x}{x} = 1

Результат компіляції:

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

Команда **\stackrel** розташовує текст над символом, дозволяючи розташувати написи над символами

$$ A \stackrel{a'}{\to} B \stackrel{b'}{\to} C $$

Результат компіляції:

$$A \stackrel{a'}{\to} B \stackrel{b'}{\to} C$$

Подібна команда також визначена як **\overset**. Ще одна команда, **\underset**, дозволяє робити підписи під символами:

$$ A \overset{a'}{\to} B \underset{b'}{\to} C $$

Результат компіляції:

$$A \overset{a'}{\to} B \underset{b'}{\to} C$$

Математичні акценти можна розставити таким чином:

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

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

Команди **\overline** та **\underline** використовують відповідно для надкреслення та підкреслення формули:

```
$\overline{x^2+\overline{y}} = \underline{5z}$.
```

$$\overline{x^2 + \overline{y}} = \underline{5z}$$

Щоб відобразити знак вектора над кількома символами, слід скористатися командою **\overrightarrow{XY}**: $\overrightarrow{XY}$.

Під час роботи з математичними виразами часто потрібні групуючі оператори (обмежувачі) різних розмірів. Це такі оператори:

( ) { } / / [ ] ⟨ ⟩ | | ‖ ‖ ⌊ ⌋ ⌈ ⌉

Групуючі оператори доступні у п'яти розмірах:

- нормальний розмір;
- \big — трохи більше за нормальний розмір;
- \Big — у 1,5 рази більше за розмір оператора \big;
- \bigg — у 2,0 рази більше за розмір оператора \big;
- \Bigg — у 2,5 рази більше за розмір оператора \big.

$$\Bigg[\Big[\big[[A] \big] \Big] \Bigg]$$

Запис тексту всередині математичних формул для pdfLaTeX здійснюється командами **\text, \textrm, \textit, \textbf** та **\mbox**.

Таблиця 7.4 — Великі оператори

$\sum$	\sum	$\int$	\int	$\oint$	\oint
$\prod$	\prod	$\coprod$	\coprod	$\|$	\Vert
$\bigcup$	\bigcup	$\bigcap$	\bigcap	$\biguplus$	\biguplus
$\bigvee$	\bigvee	$\bigwedge$	\bigwedge	$\bigcirc$	\bigcirc
$\bigtriangleup$	\bigtriangleup	$\bigtriangledown$	\bigtriangledown	$\bigodot$	\bigodot
$\bigsqcup$	\bigsqcup	$\bigotimes$	\bigotimes	$\bigoplus$	\bigoplus

Таблиця 7.5 — Стрілки

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

7.1 Оточення equation

Оточення **equation** надає можливість оформити математичні формули з використанням нумерації та автоматичної генерації посилань. Виділена формула, сформована за допомогою цього оточення, автоматично отримає номер. Для посилання на формулу з будь-якого місця документу їй присвоюється мітка командою **label** (обов'язковий параметр команди — мітка формули):

label{**мітка**}

Посилання на формулу в тексті можна задати за допомогою однієї з команд (як обов'язковий параметр виступає мітка формули, на яку посилаються):

- **ref** генерування посилання тільки на номер формули;
- **eqref** генерування посилання тільки на номер формули в дужках;
- **pageref** генерування посилання і на номер сторінки, на яку потрапляє формула.

За необхідності самостійно нумерувати формули можна скористатися командами **eqno** (номер праворуч від формули) та **leqno** (номер ліворуч від формули): автоматичне генерування посилань при цьому не передбачається. Приклад використання оточення **equation** для створення виділених формул, організації їх нумерації та створення посилань на формулу у тексті документу:

```
\begin{equation}
%мітка першої формули
\label{Formula1}
% перша формули
y=\bigl( \frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}
\bigr)
\end{equation}
\begin{equation}
%мітка другої формули
\label{Formula2}
% друга формули
y=\Bigl( \frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}
\Bigr)
\end{equation}
\begin{equation}
%мітка третьої формули
\label{Formula3}
% третя формула
y=\biggl(\frac{\sqrt{a_1}+1}
{\sqrt[3]{b_1^2}}\biggr)
\end{equation}
\begin{equation}
%мітка четвертої формули
\label{Formula4}
% четверта формула
y=\Biggl(\frac{\sqrt{a_1}+1}
{\sqrt[3]{b_1^2}}\Biggr)
\end{equation}
% нумерація формул вручну
$$ 2^2=4\eqno (@)$$
$$ 2^2=4 \leqno (*) $$
%посилання в тексті на першу формулу
Перша формула на сторінці \pageref{Formula1} ілюструє
вставку дужок за допомогою команд bigl--bigr.
%посилання в тексті на другу формулу
```

```
Формула номер \eqref{Formula2} ілюструє вставку дужок
за допомогою команд Bigl--Bigr.
%посилання в тексті на четверту формулу
Формула (\ref{Formula4} на сторінці
\pageref{Formula4}) ілюструє вставку дужок за
допомогою команд Biggl--Biggr.
```

Результат компіляції:

$$y = (\frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}) \tag{2}$$

$$y = \left(\frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}\right) \tag{3}$$

$$y = \left(\frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}\right) \tag{4}$$

$$y = \left(\frac{\sqrt{a_1}+1}{\sqrt[3]{b_1^2}}\right) \tag{5}$$

$$2^2 = 4 \tag{@}$$

$$2^2 = 4 \tag{$*$}$$

Перша формула на сторінці 154 ілюструє вставку дужок за допомогою команд bigl–bigr. Формула номер (3) ілюструє вставку дужок за допомогою команд Bigl–Bigr. Формула (5 на сторінці 154) ілюструє вставку дужок за допомогою команд Biggl–Biggr.

7.2 Системи рівнянь

Системи рівнянь в LaTeX, як правило, створюють за допомогою оточення **eqnarray** для нумерованих та **eqnarray*** для ненумерованих формул. В оточенні рівняння, що розміщуються на окремих рядках, відділяються одне від одного командами \\. Вираз в межах рядка складається з трьох частин (можна порожніх), що розділено &. Кожна частина розташовується в своєму стовпчику. В лівому стовпчику формули вирівнюються по правому краю, в середньому — по центру, а в правому — по лівому краю. Символи & задають точки вирівнювання формул різних рядків по вертикалі. Кожен рядок в оточенні **eqnarray** нумерується. Відмінити нумерацію будь-якого рядка можна командою **nonumber**. Приклад фрагменту документу для набору кількох формул наведено нижче:

```
\begin{eqnarray}
%перша формула
C=2 \pi r& \approx &6.283r \\
%друга формула
S=\pi r^2 & \approx &3.142r^2 \\
%усунення нумерації третьої формули
\nonumber
%третя формула
d=2 \sqrt{s/\pi} &\approx &1.28 \sqrt{S}\\
%четверта формула
\pi= \frac{C}{d} &\approx &
3.141592653589792 \ldots
\end{eqnarray}
```

Результат компіляції:

$$
\begin{aligned}
C = 2\pi r &\approx 6.283r && (6)\\
S = \pi r^2 &\approx 3.142r^2 && (7)\\
d = 2\sqrt{s/\pi} &\approx 1.28\sqrt{S} &&\\
\pi = \frac{C}{d} &\approx 3.141592653589792\ldots && (8)
\end{aligned}
$$

Оточення **equation** і **eqnarray** використовують один лічильник **equation**, значення якого друкується як номер формули, тому всі нумеровані формули матимуть єдину нумерацію.

7.3 Системи рівнянь без вирівнювання

Для системи рівнянь без вирівнювання по вертикалі в пакеті **amsmath** визначено оточення **gather**. Воно дозволяє створювати нумеровані системи рівнянь (ненумеровані створює оточення **gather***). Місця перенесення рядків задаються командою \\, а всі формули в рядках розміщуються по центру. Приклад використання оточення:

```
\begin{gather}
%перша формула
C=2 \pi r \approx 6.283r \\
%друга формула
\nonumber
S=\pi r^2 \approx 3.142r^2 \\
\nonumber
%третя формула
```

```
d=2 \sqrt{s/\pi} \approx 1.28 \sqrt{S}\\
%четверта формула
\nonumber
\pi= \frac{C}{d} \approx
3.141592653589792 \ldots
\end{gather}
```

Результат компіляції:

$$
\begin{gathered}
C = 2\pi r \approx 6.283r \\
S = \pi r^2 \approx 3.142r^2 \\
d = 2\sqrt{s/\pi} \approx 1.28\sqrt{S} \\
\pi = \frac{C}{d} \approx 3.141592653589792\ldots
\end{gathered}
\tag{9}
$$

7.4 Системи рівнянь з вирівнюванням

Для систем рівнянь з додатковим вирівнюванням по вертикалі в пакеті **amsmath** визначено оточення **align** для нумерованих, та **align*** для ненумерованих формул. Точки вирівнювання рівнянь задаються &. На відміну від **eqnarray** немає необхідності штучно розбивати рівняння на три частини:

```
\begin{align}
%перша формула
C=& 2 \pi r \approx 6.283r \\
%друга формула
\nonumber
S=& \pi r^2 \approx 3.142r^2 \\
\nonumber
%третя формула
d=& 2 \sqrt{s/\pi} \approx 1.28 \sqrt{S}\\
%четверта формула
\nonumber
\pi=& \frac{C}{d} \approx
3.141592653589792 \ldots
\end{align}
```

Результат компіляції:

$$
\begin{aligned}
C =& 2\pi r \approx 6.283r \\
S =& \pi r^2 \approx 3.142r^2 \\
d =& 2\sqrt{s/\pi} \approx 1.28\sqrt{S} \\
\pi =& \frac{C}{d} \approx 3.141592653589792\ldots
\end{aligned}
\tag{10}
$$

7.5 Розбиття довгих формул

Виділені формули LaTeX не переносить. Якщо така формула не вміщується в рядку, то під час трансляції видається повідомлення про переповнення (Overfull ...). Для вводу довгих формул їх доводиться розбивати на рядки вручну. Для створення таких формул можна скористатися оточеннями **eqnarray** або **eqnarray***. Наприклад:

```
\begin{eqnarray*}
%перший рядок формули
&\frac{b-a}{2}\int_{-1}^1
f(\frac{a+b}{2} + \frac{b-a}{2}s)dt &\\
%додаткові пробіли для зсуву другого рядка
%відносно першого
& \qquad \qquad \qquad \qquad \qquad \qquad
%другий рядок формули
\approx \frac{b-a}{2}\sum_{j=1}^N
c_j^{(n)} f(\frac{a+b}{2} +
\frac{b-a}{2}s_j) &
\end{eqnarray*}
```

Результат компіляції:

$$
\begin{aligned}
&\tfrac{b-a}{2}\int_{-1}^{1} f(\tfrac{a+b}{2} + \tfrac{b-a}{2}s)dt \\
&\qquad\qquad \approx \tfrac{b-a}{2}\sum_{j=1}^{N} c_j^{(n)} f(\tfrac{a+b}{2} + \tfrac{b-a}{2}s_j)
\end{aligned}
$$

Щоб частини формули в різних рядках не починалися точно одна під одною вставлено додаткові пропуски командою **qquad**. Для нумерації розбитих таким чином формул використовується оточення **eqnarray**, а перед командою \\, що завершує розірваний рядок, необхідно вставити команду **nonumber**. Зсув праворуч частини формули, що розташовано в другому рядку, задається точкою вирівнювання (&). Наприклад (зверніть увагу на розташування точок вирівнювання):

```
\begin{eqnarray}
\nonumber
%перший фрагмент формули
\int_{t_j}^{t_j+1} K(x,s)
\varphi(s)ds =& &\\
%другий фрагмент формули
\frac{h}{2}\int_{-1}^1 &
K(x,\frac{t_{j+1}+t{j}}{2} +
\frac{h}{2}s) \varphi(\frac{t_{j+1}+t{j}}{2} +
\frac{h}{2}s)ds &
\end{eqnarray}
```

Результат компіляції:

$$\int_{t_j}^{t_j+1} K(x,s)\varphi(s)ds = \frac{h}{2}\int_{-1}^{1} K(x,\tfrac{t_{j+1}+tj}{2}+\tfrac{h}{2}s)\varphi(\tfrac{t_{j+1}+tj}{2}+\tfrac{h}{2}s)ds \tag{11}$$

7.6 Матриці

Пакет **amsmath** містить кілька спеціалізованих оточень для набору матриць, що застосовуються в середині математичних формул. Всередині стовпчиків елементи завжди центруються. Максимально матриця може містити до 10 стовпчиків. Приклади створення матриць (використовується одна й та ж матриця, тільки створена з використанням різних оточень):

7.6.1 *Оточення matrix:*

```
$$
\begin{matrix}
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{matrix}
$$
```

Результат компіляції:

$$
\begin{matrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{matrix}
$$

7.6.2 *Оточення pmatrix:*

```
$$
\begin{pmatrix}
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{pmatrix}
$$
```

Результат компіляції:

$$
\begin{pmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{pmatrix}
$$

7.6.3 *Оточення bmatrix:*

```
$$
\begin{bmatrix}
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{bmatrix}
$$
```

Результат компіляції:

$$
\begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
$$

7.6.4 *Оточення Bmatrix:*

```
$$
\begin{Bmatrix}
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{Bmatrix}
$$
```

Результат компіляції:

$$\begin{Bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{Bmatrix}$$

7.6.5 *Оточення vmatrix:*

```
$$
\begin{vmatrix}
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{vmatrix}
$$
```

Результат компіляції:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

7.6.6 Оточення Vmatrix:

```
$$
\begin{Vmatrix}
```

```
%перший рядок
a_{11} & a_{12}&a_{13} \\
%другий рядок
a_{21} & a_{22}&a_{23} \\
%третій рядок
a_{31} & a_{32}&a_{33}
\end{Vmatrix}
$$
```

Результат компіляції:

$$\left\|\begin{matrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{matrix}\right\|$$

Ряд крапок у матрицях можна видрукувати командою **hdotsfor**. Обов'язковий аргумент команди задає кількість стовпчиків, що зайнято крапками, а необов'язковий, що стоїть першим, — відстань між крапками у відносних одиницях до стандартного. Наприклад:

```
$$
K_n^a(\omega) =
\begin{Vmatrix}
a_{11} & 0 & \hdotsfor{2} & 0 \\
\omega_{i} & a_{22} & 0 & \dots & 0 \\
\hdotsfor[2]{5} \\
\omega_{n1} & \omega_{n2} &
\dots & \omega_{n\,n-1} & a_{nn}
\end{Vmatrix}
$$
```

Результат компіляції:

$$K_n^a(\omega) = \left\|\begin{matrix} a_{11} & 0 & \hdots & 0 \\ \omega_{i} & a_{22} & 0 & \dots & 0 \\ \hdotsfor{5} \\ \omega_{n1} & \omega_{n2} & \dots & \omega_{n\,n-1} & a_{nn} \end{matrix}\right\|$$

Для того, щоб крапки розташовувалися по всій довжині рядка варто скористатися командою **dotfill**. Цю команду можна використовувати й у текстовій моді.

Для вставки матриці в текст документу можна скористатися оточенням **smallmatrix** (розташовується в межах текстової формули). Наприклад (матриця вставляється між квадратними дужками, що створено командами **left[** та **right]**):

```
Приклад ілюструє можливість вставки
матриці $\left[
\begin{smallmatrix}
1 & 0 \\
0 & 1 \end{smallmatrix}
\right]$ в текстовий рядок.
```

Результат компіляції:
Приклад ілюструє можливість вставки матриці $\left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ в текстовий рядок.

7.7 Система умов з дужками

Для набору систем умов в пакеті **amsmath** передбачено оточення **cases**.
Наприклад:

```
$$ f(x) =
\begin{cases}
%перша умова
A \sin(\omega t+\varphi) & \text{при $x< 0$,}\\
%друга умова
1 & \text{при $x= 0$,}\\
%третя умова
A \cos(\omega t+\varphi) & \text{при $x>0$}
\end{cases} $$
```

Результат компіляції:

$$f(x) = \begin{cases} A\sin(\omega t+\varphi) & \text{при } x<0, \\ 1 & \text{при } x=0, \\ A\cos(\omega t+\varphi) & \text{при } x>0 \end{cases}$$

7.8 Кратні інтеграли

Команди **iint**, **iiint** і **iiiint** дозволяють створювати кратні інтеграли, а команда **idotsint** виводить знак інтегралу з трьома крапками між ними; ці команди визначені в пакеті **amsmath**. Наприклад:

```
$$ \iint\limits_S f(x,y) dxdy $$
```

Результат компіляції:

$$\iint\limits_S f(x,y) dxdy$$

```
$$ \iiint\limits_V f(x,y,z) dxdydz $$
```

Результат компіляції:

$$\iiint\limits_V f(x,y,z) dxdydz$$

```
$$ \idotsint\limits_V f(x,y,z) dxdyz $$
```

Результат компіляції:

$$\idotsint\limits_V f(x,y,z) dxdyz$$

7.9 Багаторядкові індекси

Команда **substack** з пакету **amsmath** дозволяє створювати багаторядкові індекси для символів з межами. Вони розбиваються на рядки командою \\.

Наприклад:

```
$$
\sum_{\substack{0<i<N\\0<j<N\\0<k<N}}
S_0(i,j,k)
$$
```

Результат компіляції:

$$\sum_{\substack{0<i<N\\0<j<N\\0<k<N}} S_0(i,j,k)$$

7.9.1 *Індекси по кутах символів*

Команда **sideset** (пакету **amsmath**) дозволяє виводити індекси по кутах символів з межами. Наприклад:

```
$$\sideset{_1^2}{_i^j}\prod_k D_k $$
$$\sideset{}{''}\sum_{0\le i\le k}S_i\beta z$$
```

Результат компіляції:

$$\sideset{_1^2}{_i^j}\prod_k D_k$$

$$\sideset{}{''}\sum_{0\le i\le k} S_i\beta z$$

7.9.2 *Стрілки з індексами*

Команди **xleftarrow** та **xrightarrow** з пакету **amsmath** дозволяють створювати стрілки з індексами. Необов'язковий параметр задає індекси під стрілкою, а обов'язковий — над нею. Наприклад:

```
$$ -\infty \xleftarrow[k]{\chi=1}
\Omega \oplus \Psi
\xrightarrow{\Delta_k \alpha(\omega)}
H_{i,j}^\gamma $$
```

Результат компіляції:

$$-\infty \xleftarrow[k]{\chi=1} \Omega \oplus \Psi \xrightarrow{\Delta_k \alpha(\omega)} H_{i,j}^{\gamma}$$

7.10 Оточення array

Оточення за функціональними можливостями фактично повністю повторює оточення **tabular**, але використовується для роботи в математичному режимі: для створення векторів, матриць, системи рівнянь чи розбиття формул на кілька рядків:

\begin{array}{преамбула}
...
\end{array}

преамбула описує структуру таблиці аналогічно до оточення **tabular**.

В доповнення до стандартних типів колонок, що використовуються в попередньому оточенні, можна використовувати дві нові:

m{N} — ширина комірки (аналогічно до p{N}) з одночасним вертикальним вирівнюванням вмісту по центру;

b{N} — ширина комірки з одночасним вирівнюванням вмісту за нижньою базовою лінією останнього рядка.

Приклади використання оточення:

1) створення матриці:

```
$$
\begin{array}{cccc}
a_{11} & a_{12} & \dots & a_{1n}\\
a_{21} & a_{22} & \dots & a_{23}\\
\dots & \dots & \dots & \dots \\
a_{n1} & a_{n2} & \dots & a_{nn}
\end{array}
$$
```

Результат компіляції:

$$
\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{23} \\
\ldots & \ldots & \ldots & \ldots \\
a_{n1} & a_{n2} & \ldots & a_{nn}
\end{array}
$$

2) визначник матриці:

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

Результат компіляції:

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

3) система рівнянь:

```
$$
\left\{
\begin{array}{ccc}
a_1 x+b_1 y+c_1 z=k_1\\
a_2 x+b_2 y+c_2 z=k_2\\
a_3 x+b_3 y+c_3 z=k_3\\
\end{array}
\right.
$$
```

Зауважте: щоб права фігурна дужка не відображалася, після команди **right** стоїть крапка.

Результат компіляції:

$$
\left\{
\begin{array}{ccc}
a_1 x+b_1 y+c_1 z=k_1\\
a_2 x+b_2 y+c_2 z=k_2\\
a_3 x+b_3 y+c_3 z=k_3\\
\end{array}
\right.
$$

4) система умов:

```
$$
y(x)=
\left\{
\begin{array}{rcc}
-1& \text{при} &x<0\\
0&\text{при} &x=0\\
1&\text{при} &x>0\\
\end{array}
\right.
$$
```

Результат компіляції:

$$y(x)=\left\{\begin{array}{rcc} -1& \text{при} &x<0\\ 0&\text{при} &x=0\\ 1&\text{при} &x>0\\ \end{array}\right.$$

5) багаторядкова формула (**qquad** задає додатковий зсув по горизонталі для другого фрагменту формули):

```
$$
\begin{array}{l}
e^x=1+x+\frac{x^2}{2!}\\
\qquad {}+\frac{x^3}{3!}+\cdots
\end{array}
$$
```

Результат компіляції:

$$\begin{array}{l} e^x=1+x+\frac{x^2}{2!}\\ \qquad {}+\frac{x^3}{3!}+\cdots \end{array}$$

7.11 Визначення додаткових оточень

Книжки та статті з математики і не тільки зазвичай містять теореми та інші теоремоподібні структури, такі як леми, аксіоми тощо. Нематематичний текст також може містити аналогічні структури: правила, закони, принципи тощо.

Оточення, які використовують в LaTeX'і для оформлення фрагментів тексту типу «теорема», заздалегідь не визначені, через те, що кількість різноманітних типів об'єктів на кшталт теореми, що присутні в одному тексті, може бути досить великою. Це можуть бути теореми, визначення, леми, припущення, зауваження, приклади тощо.

Будь-яке оточення самостійно організовує нумерацію введених об'єктів, а для нумерації по підрозділах можна використати відповідні підлеглі лічильники. За замовчуванням при виводі заголовку використовується напівжирний шрифт, а для самого формулювання — курсив. Шрифти можна задати явним чином за допомогою команд форматування тексту. При створенні оточення автоматично створюється і відповідний лічильник, ім'я якого співпадає з його іменем.

Таким чином створення процедур на кожен випадок є марним витрачанням ресурсів, і тому LaTeX має декларацію **\newtheorem**. Нові оточення визначають в преамбулі документа.

Команда **\newtheorem** зазвичай використовують в одному з таких двох варіантів: **\newtheorem{оточення}[лічильник]{назва}**
\newtheorem{оточення}{назва}[розділ]

Ця команда має два обов'язкових аргументи.

Аргумент **оточення** — це назва нового типу оточення, — коротке ключеве слово, яку використовують для ідентифікації теореми. Воно не має збігатися з іменем вже існуючого оточення чи лічильника.

Аргумент **назва** визначає ту назву оточення типу теореми (теорема, лема, визначення, закон тощо), під яким вона друкуватиметься в документі. За замовченням він друкується напівжирним шрифтом, а формулювання — курсивом. Абзац, що розташований після оточення, починається з абзацним відступом, якщо після команди, що закриває оточення розташовано порожній рядок, і без відступу у протилежному випадку.

Аргументи в квадратних дужках є необов'язковими. Обидва вони використовуються для визначення стилю нумерації теореми.

Аргументом **лічильник** є назва попередньо визначеної теореми. За наявності цього аргументу для нумерації використовуватиметься раніше визначений лічильник, тоді **оточення** і **theorem** матимуть єдину нумерацію. Тоді нова теорема нумеруватиметься у тій же послідовності. Якщо у списку аргументів **лічильник** є відсутнім, то для кожного **оточення** створюють лічильник з тим же іменем.

Аргумент **розділ** дозволяє вказати розділ, всередині якого нумеруватиметься створене оточення. У варіанті команди **\newtheorem{оточення}{назва}[розділ]** аргумент **розділ** — це ім'я вже існуючого лічильника, зазвичай лічильника команд секціонування. Фактично вибирається розділ, всередині якого будут нумеруватимуться «теоремы». Створений лічильник **оточення** є внутрішнім для **розділ**: він автоматично обнулятиметься за кожної зміни значення лічильника **розділ**.

Оточення, що створюють, можуть мати один необов'язковий аргумент, який зазвичай використовують для позначення автора; він друкуватиметься після

номера «теоремы».

Посилання на теореми здійснюються так, як і посилання на оточення типу **equation**.

Найчастіше створюють такі оточення

\newtheorem{theorem}{Теорема}
\newtheorem{property}{Властивість}
\newtheorem{proposition}{Твердження}
\newtheorem{remark}{Зауваження}
\newtheorem{lemma}{Лема}
\newtheorem{problem}{Проблема}
\newtheorem{corollary}{Наслідок}
\newtheorem{definition}{Означення}
\newtheorem{example}{Приклад}
\newtheorem{task}{Задача}
\newtheorem{vpr}{Вправа}
\newtheorem{zavd}{}

Лічильники для цих оточень визначають як

\renewcommand{\thezavd}{\arabic{chapter}.\arabic{zavd}.}
\renewcommand{\theexample}{\arabic{chapter}.\arabic{example}.}

Розглянемо кілька прикладів.

7.11.1 Оточення Завдання

В преамбулі визначаємо оточення

\newtheorem{zavd}{}

і лічильник

\renewcommand{\thezavd}{\arabic{chapter}.\arabic{zavd}.

В тексті позначаємо це завдання

```
\begin{zavd}
 Кинуто дві гральні кості. Знайти ймовірність того, що сума очок на
 гранях, які випали, --- парна, причому на грані щонайменше однієї
 з костей з'явиться шістка.
\end{zavd}
```

Результат компіляції

1.8. *Кинуто дві гральні кості. Знайти ймовірність того, що сума очок на гранях, які випали, — парна, причому на грані щонайменше однієї з костей з'явиться шістка.*

7.11.2 Оточення Приклад

В преамбулі визначаємо оточення

\newtheorem{example}{Приклад}

і лічильник

\renewcommand{\theexample}{\arabic{chapter}.\arabic{example}.

В тексті позначаємо цей приклад

```
\begin{example}
  З урни, де міститься дві білих і три чорних кульки, витягнуто
  одну кульку, колір якої невідомий. Наступна витягнута кулька ---
  біла. Яка ймовірність того, що перша кулька також біла?
\end{example}
```

Результат компіляції

Приклад 1.6. *З урни, де міститься дві білих і три чорних кульки, витягнуто одну кульку, колір якої невідомий. Наступна витягнута кулька — біла. Яка ймовірність того, що перша кулька також біла?*

8 Титульний аркуш

Для того, щоб оформити заголовок до всього документу, необхідно: задати інформацію для заголовка (автор, назва тощо) й дати LaTeX'у команду цей заголовок згенерувати. Останнє здійснюється за допомогою команди **maketitle**. Вона створить титульний аркуш, якщо це передбачено класом та опціями. (Якщо титульний аркуш не передбачено, то команда **maketitle** розташує задану інформацію про автора, заголовок та інше на першій сторінці, вибравши потрібні шрифти й зробивши відповідні відступи між титульною інформацією та текстом.) За замовчуванням для класів **report** і **book** титульний аркуш створюється завжди (й не створюється, якщо в описі класу задано опцію **notitlepage**), для класу **article** титульний аркуш не створюється (але буде створено, якщо вказати опцію **titlepage**). В класі **proc** титульна інформація завжди друкується на першій сторінці тексту.

Пояснимо, як задавати LaTeX'у інформацію для титульного аркуша. Автор задається за допомогою команди **author**. Вона має єдиний обов'язковий аргумент — ім'я автора. Якщо авторів кілька, їх імена мають бути розділені командою **and**.

Заголовок задається за допомогою команди **title**. Якщо заголовок є довгим, можна задати його розбиття на рядки за допомогою команди \\.

Наступний елемент інформації для титульного аркуша — команда **date**. Вона має один обов'язковий аргумент де можна задати будь-який текст, який буде розташовано на титульному аркуші (або перед початком основного тексту). Зокрема, можна залишити аргумент цієї команди «порожнім», тоді тексту не буде. Якщо взагалі не буде цієї команди, то LaTeX надрукує на титульному аркуші дату свого запуску.

Команди **author**, **title** і **date** можна подавати у будь-якому порядку, але обов'язково до команди **maketitle**, яка має бути першою з команд, що генерують текст.

Титульний аркуш, що створено командою **maketitle** найчастіше не відповідає формам титульних аркушів, які мають бути у вихідних документах, таких, як пояснювальна записка до кваліфікаційної, курсової роботи, звіту з науково-дослідної роботи, робочої програми, комплексу навчально-методичного забезпечення тощо.

У таких випадках не використовують стиль оформлення титульного аркушу, який визначається тим, що диктує LaTeX. Для цього варто скористатися оточенням **titlepage**. Текст між **begin{titlepage}** та **end{titlepage}** утворить титульний аркуш, оформлення якого визначається користувачем.

LaTeX всередині цього оточення робить таке:

- встановлює друкування в одну колонку (навіть якщо сам документ буде друкуватися у дві колонки);

- починає нову сторінку й встановлює лічильник кількості сторінок в нуль;
- встановлює на сторінці стиль оформлення **empty** (без колонтитула й номера).

Для форматування параметрів титульного аркушу варто скористатися послідовністю команд **hfill** **break**, яка додає порожній рядок в тексті.

Наведемо приклад титульного аркушу для комплексу навчально-методичного забезпечення.

```
\begin{center}

Міністерство освіти і науки України

 \vspace{1cm}

Харківський національний університет радіоелектроніки

 \vspace{1cm}
%Факультет \underline{Інфокомунікацій} \\
%\hspace{20mm} \footnotesize{(назва факультету)}

\large{Кафедра \underline{метрології та технічної експертизи}}
 \\
\hspace{20mm} \footnotesize{(назва кафедри)}

\vspace{2.5cm}

\large{\textbf{КОМПЛЕКС НАВЧАЛЬНО-МЕТОДИЧНОГО ЗАБЕЗПЕЧЕННЯ}}

\vspace{0.5cm}

\large{навчальної дисципліни}

 \vspace{0.5cm}

\underline{ \hspace{3cm} Моделювання на ЕОМ \hspace{3cm}}  \\
\footnotesize{(назва дисципліни)}

\vspace{1cm}

\large{підготовки \underline{\hspace{3.5cm} бакалаврів
```

```
\hspace{5cm}}} \\
\hspace{15mm} \footnotesize{(назва освітнього рівня)}
\end{center}

\large{спеціальність \underline{152 --- метрологія та
інформаційно-вимірювальна техніка}} \\
\hspace*{70mm}  \footnotesize{(шифр і назва спеціальності)}

\vspace{1cm}

\large{спеціалізація \underline{метрологія, стандартизація
та сертифікація}} \\
\hspace*{70mm}  \footnotesize{(назва спеціалізаці)}

\vspace{1cm}

\large{освітня програма підготовки \underline{
 \hspace{10cm}}} \\
\hspace*{20mm} \underline{ \hspace{15cm}} \\
\hspace*{70mm}  \footnotesize{(назва освітньої програми)}

\vspace{2cm}

\large{Розробник(и):   \underline{ \hspace{2cm}
Нікітенко О.\, М., доцент, к.т.н, с.н.с \hspace{2cm}} } \\
\hspace*{40mm}  \footnotesize{(ініціали та прізвище авторів,
їхні посади, наукові ступені та вчені звання)}

\vspace{1cm}

\large{Схвалено на засіданні кафедри \underline{метрології та
технічної експертизи}}

\vspace{1cm}

  Протокол  від ''\underline{2}'' \underline{жовтня} 2017 р.
  № \underline{2}

\vspace{0.5cm}
```

```
\begin{center}
Харків 2017
\end{center}
```

За подібним шаблоном будуються титульні аркуші для інших документів.

Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра метрології та технічної експертизи
(назва кафедри)

КОМПЛЕКС НАВЧАЛЬНО-МЕТОДИЧНОГО ЗАБЕЗПЕЧЕННЯ

навчальної дисципліни

Моделювання на ЕОМ
(назва дисципліни)

підготовки бакалаврів
(назва освітнього рівня)

спеціальність 152 — метрологія та інформаційно-вимірювальна техніка
(шифр і назва спеціальності)

спеціалізація метрологія, стандартизація та сертифікація
(назва спеціалізаці)

освітня програма підготовки _____
(назва освітньої програми)

Розробник(и): Нікітенко О. М., доцент, к.т.н, с.н.с
(ініціали та прізвище авторів, їхні посади, наукові ступені та вчені звання)

Схвалено на засіданні кафедри метрології та технічної експертизи

Протокол від "2" жовтня 2017 р. № 2

Харків 2017

Наступним прикладом буде титульний аркуш для методичних вказівок.

```
\begin{titlepage}
\begin{center}

\large

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

\vspace{0.5cm}

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

\vspace{5cm}

МЕТОДИЧНІ ВКАЗІВКИ \\[0.5cm]
до лабораторного практикуму з дисциплін \\[0.5cm]

«МОДЕЛЮВАННЯ НА ЕОМ»  та\\
«ОСНОВИ МОДЕЛЮВАННЯ ПРОЦЕСІВ НА ПЕОМ»


\vspace{0.5cm}

для студентів спеціальності \\ 152 --- «Метрологія та
інформаційно-вимірювальна техніка»

Кваліфікація "--- бакалавр

\vspace{2cm}
{\Huge{Електронне видання}}
\end{center}

\vspace{2cm}

\newlength{\ML}
\settowidth{\ML}{«\underline{\hspace{0.7cm}}»
\underline{\hspace{2cm}}}

\hfill\begin{minipage}{0.4\textwidth}
\large
```

```
  ЗАТВЕРДЖЕНО\\
  кафедрою метрології \\
  та технічної експертизи. \\
  Протокол № 2 від 2.10.2017 \\

\end{minipage}

\vspace{3cm}

\begin{center}
\large
Харків 2018
\end{center}
\end{titlepage}
```

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторного практикуму з дисциплін

«МОДЕЛЮВАННЯ НА ЕОМ» та «ОСНОВИ МОДЕЛЮВАННЯ ПРОЦЕСІВ НА ПЕОМ»

для студентів спеціальності
152 — «Метрологія та інформаційно-вимірювальна техніка»
Кваліфікація — бакалавр

Електронне видання

ЗАТВЕРДЖЕНО
кафедрою метрології
та технічної експертизи.
Протокол № 2 від 2.10.2017

Харків 2018

9 Презентація

У звичайному уявленні презентація є послідовністю зображень, яку оптимізовано для показу через проектор [5].

Створення презентації не є простою роботою. Створюючи презентацію, варто не забувати про емпіричні правила:

- один слайд потребує не менше однієї хвилини;
- один слайд з контентом потребує не менше п'яти хвилин;
- часу на показ завжди не вистачає;
- не варто «пхати» в презентацію більше слайдів, ніж вийде розповісти за часом. Перебір за часом тільки дратує слухачів;
- кожен слайд мусить мати свій заголовок (**frametitle**).
- в один слайд можна вмістити приблизно 20–40 слів й напевно не більше 80;
- бажано використовувати **block, theorem, proof** і **example**. Ці оточення структурують текст і допомогають виділяти основні думки;
- для різних аудиторій ці правила можуть суттєво різнитися [7].

Для створення презентацій в LaTeX 2_ε можна використовувати клас документів **slides**, який має незначні можливості, або їх можна створювати за допомогою спеціальних пакетів розширення, таких як **seminar, foils, prosper, beamer, talk ppower4, powerdot, pdfscreen, pdfslide** тощо.

Одним з основних недоліків послідовності зображень є лінійна структура викладення матеріалу.

Іншим важливим напрямком є розділення структури, змісту та оформлення презентації — вони можуть змінюватися незалежно одне від одного [5].

Розглянемо пакет розширення **beamer**, який інтенсивно розвивається, має значні можливості й є одним з найбільш розвинутих та найпопулярніших для створення електроних презентацій [2, 5, 7]. Він дозволяє:

- використовувати стандартні команди системи LaTeX;
- створювати елементи, що перекриваються;
- створювати динамічні об'єкти;
- використовувати стандартні шаблони оформлення презентацій.

В класі **beamer** презентація складається з набору окремих слайдів (фреймів), кожен з яких може мати кілька сторінок (шарів, які використовуються для організації динамічних ефектів під час показу). Презентація в LaTeX 2_ε є звичайним TeX документом з преамбулою та тілом документу. Для того, щоб створити презентацію, необхідно створити відповідний tex–документ, скомпілювати його та отримати pdf-файл. Для запуску презентації на перегляд необхідно

відкрити її за допомогою **Acrobat Reader**'а та встановити повноекранний режим.

Розмір віртуальної сторінки в **beamer** дорівнює 128x96mm (співвідношення 4:3), а вікно фрейму складається з ряду елементів, присутність яких на сторінці визначається темою оформлення:

- верхня та нижня панелі;
- ліва та права бокові панелі;
- панелі навігації;
- навігаційні значки;
- малюнок-логотип;
- заголовок слайду;
- тло слайду;
- основне наповнення слайду.

Зовнішній вигляд презентації задається за допомогою п'яти груп тем:

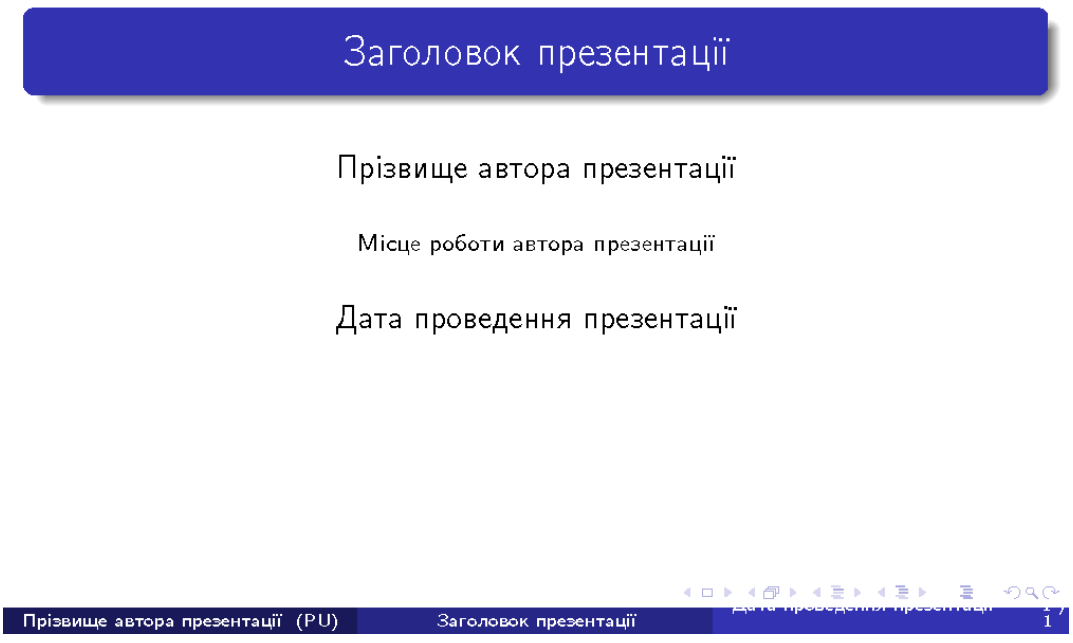
- **основна тема (theme)**: задає зовнішній вигляд всіх базових елементів презентації. Задається в преамбулі командою **usetheme [опції]{тема}**. Пакет містить такі теми: AnnArbor, Antibes, Bergen, Berkeley, Berlin, Boadilla, boxes, CambridgeUS, Copenhagen, Darmstadt, default, Dresden, Frankfurt, Goettingen, Hannover, Ilmenau, JuanLesPins, Luebeck, Madrid, Malmoe, Marburg, Montpellier, PaloAlto, Pittsburgh, Rochester, Singapore, Szeged, Warsaw;
- **внутрішня тема (inner)**: визначає вигляд елементів в межах робочої області презентації. Задається в преамбулі командою **useinnertheme [опції]{тема}**. Пакет містить такі теми: circles, default, inmargin, rectangles, rounded;
- **зовнішня тема (outher)**: визначає загальну розмітку слайду. Задається в преамбулі документу командою **useoutertheme[опції]{тема}**. Пакет містить такі теми: default, infolines, miniframes, shadow, sidebar, smoothbars, smoothtree, split, tree;
- **колірна схема (color)**: визначає колір основних елементів презентації. Задається в преамбулі командою **usecolortheme[опції] {тема}**. Пакет містить такі теми: albatross, beaver, beetle, crane, default, dolphin, dove, fly, lily, orchid, rose, seagull, seahorse, sidebartab, structure, whale, wolverine;
- **шрифтова схема (font)**: визначає шрифт основних елементів презентації. Задається в преамбулі командою **usecfonttheme [опції]{тема}**. Пакет містить такі теми: default, professionalfonts, serif, structurebold, structureitalicserif, structuresmallcapsserif.

Розмір шрифту в межах всього документу задається як параметр класу **beamer** аналогічно до звичайного документу LaTeX. Підтримуються розміри шрифтів: 8pt, 9pt, 10pt, 11pt, 12pt, 14pt, 17pt та 20pt.

Загальна структура документу для створення презентації є такою:

```
\documentclass{beamer}
\usepackage[cp1251]{inputenc}
\usepackage[ukrainian]{babel}
%задання основної теми
\usetheme {Madrid}
\title {Заголовок презентації}
\date{Дата проведення презентації}
\author{Прізвище автора презентації}
\institute[PU]{Місце роботи автора презентації}
\begin{document}
%перший фрейм презентації
\begin{frame}[t]
%формування заголовка презентації
\titlepage
\end{frame}
\end{document}
```

Результат компіляції:



Кожен фрейм презентації оформляється за допомогою оточення **frame**. Все, що знаходиться між **\begin{frame}[параметр]** та **\end{frame}**, розташовується на окремому слайді. Параметр (може бути опущеним) керує розташуванням тексту на слайді:

t — текст «притискається» до верхнього краю слайду;

b — текст «притискається» до нижнього краю слайду;

c — текст вирівнюється по центру слайду;

allowframebreaks — автоматично розбиває довгий текст на кілька слайдів.

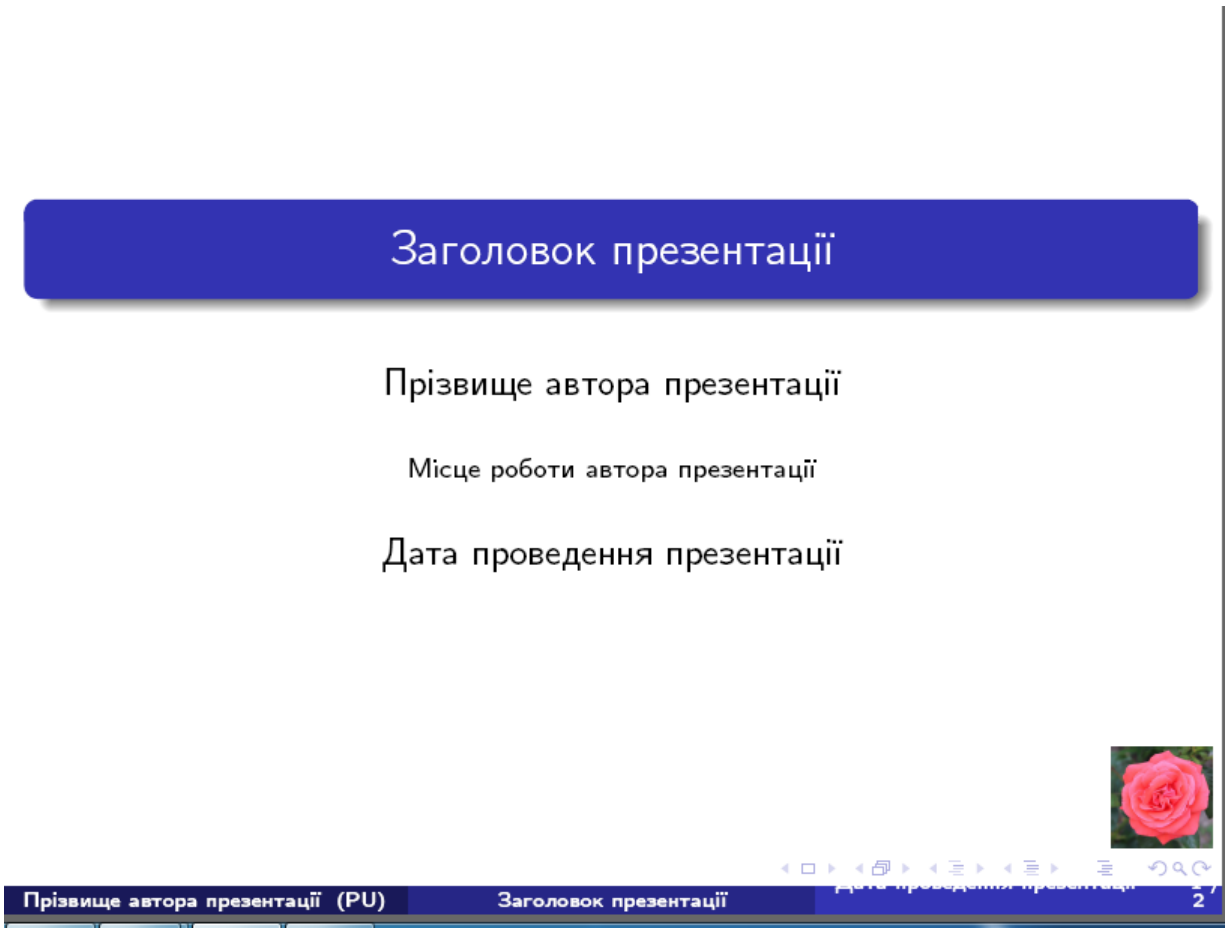
У прикладі, що наведено вище, на першому слайді формується титульна сторінка презентації за допомогою команди **\titlepage**. Необов'язковий параметр до команди **\author, \title, \date, \institute** дозволяє задати короткий текст, що буде виводитись на інформаційній панелі слайду. Наприклад: **\institute[Університет]{Місце роботи автора презентації}**

Додати логотип (зображення, що буде відображатися на всіх слайдах) до презентації можна командою **logo**. Наприклад:
\logo{\includegraphics[height=0.5cm]{Graf.jpg}}

тут

height=0.5cm — розмір логотипу;

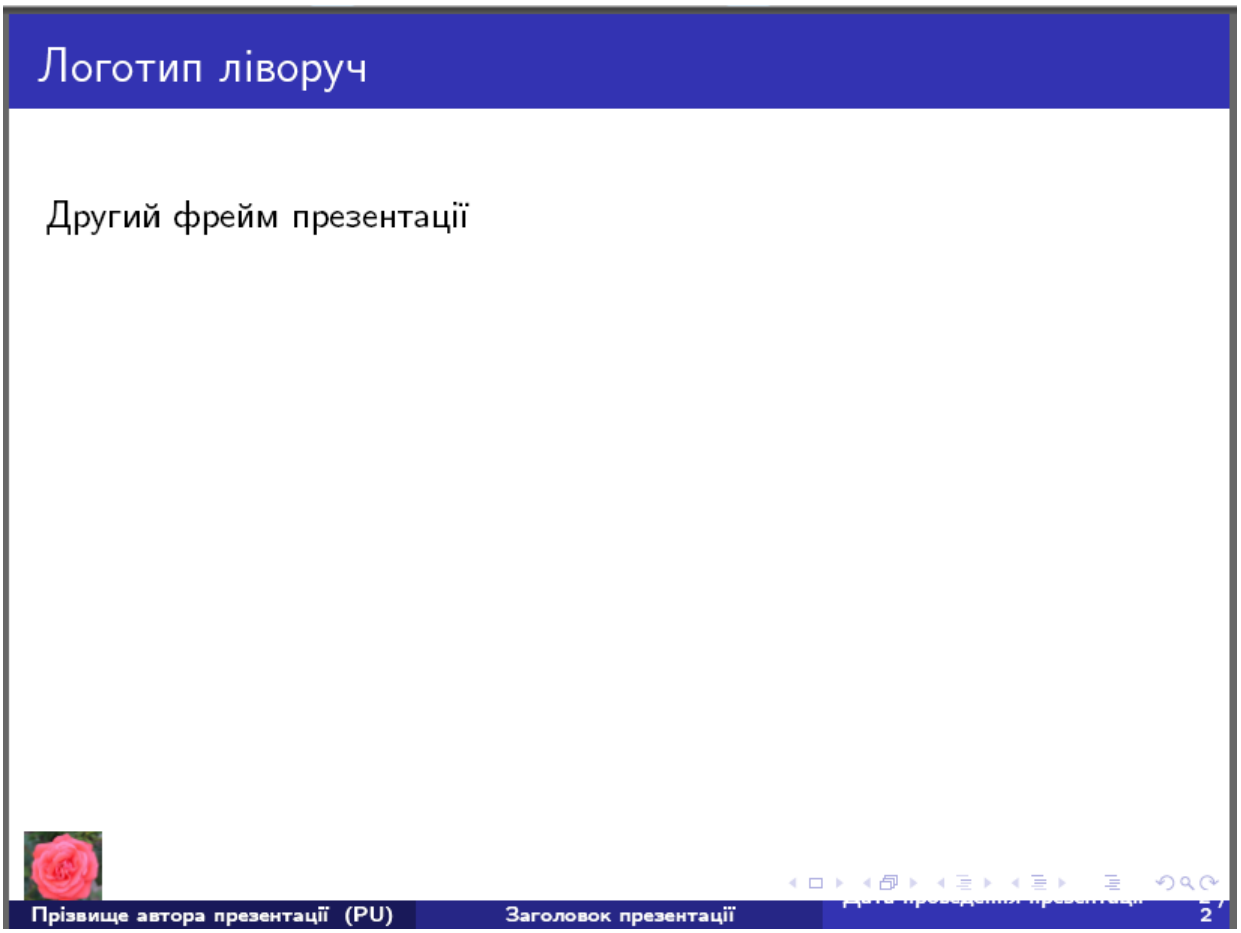
Graf.jpg — файл з зображенням, яке буде використовуватися як логотип.



Для того, щоб логотип було розташовано в іншому кутку, застосовують таку конструкцію:

```
\addtobeamertemplate{frametitle}{}{%
\begin{tikzpicture}[remember picture,overlay]
\node[anchor=south west,yshift=2pt] at (current page.south west)
{\includegraphics[height=0.8cm]{car.jpg}};
\end{tikzpicture}}
```

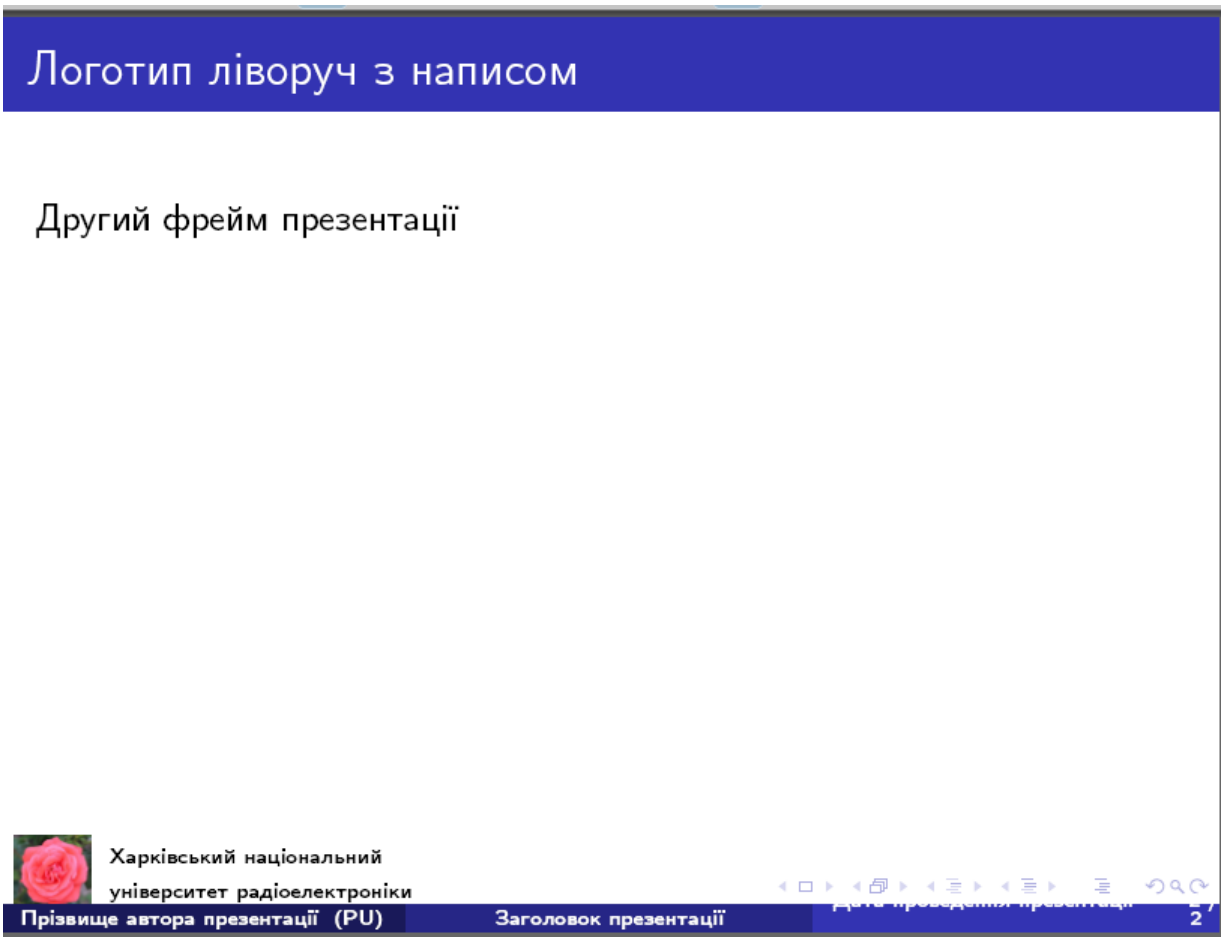
У цьому випадку логотип відображатиметься, починаючи з другого фрейму.



Щоб біля логотипу був напис, застосовують таку конструкцію:

```
\addtobeamertemplate{frametitle}{}{%
\begin{tikzpicture}[remember picture,overlay]
\node[anchor=south west,yshift=2pt] at (current page.south west)
{\includegraphics[height=0.8cm]{car.jpg}};
\node[anchor=south west,xshift=1cm,yshift=15pt] at
(current page.south west) {\color{black}\tiny{Харківський
національний}};
\node[anchor=south west,xshift=1cm,yshift=5pt] at
(current page.south west) {\color{black}\tiny{університет
```

```
радіоелектроніки}};
\end{tikzpicture}}
```



Інформаційна панель розташовується внизу кожного слайду та може бути додана за допомогою команди (якщо це не передбачено темою, яку використовують): **useoutertheme{infolines}**

9.1 Команди, що використовуються під час формування слайдів

Заголовки до фреймів формують командою **frametitle{Заголовок слайду}**, або як параметр оточення **frame**. До фрейму можна також додати підзаголовок за допомогою команди **framesubtitle{Підзаголовок слайду}**.

Для структурування презентації за межами оточення **frame** можуть використовуватися розділи **section{Назва розділу}** та підрозділи **subsection{Назва підрозділу}**. Вони полегшують швидкий доступ до слайдів відповідних розділів та дозволяють командою **tableofcontents** сформувати зміст презентації.

Окремі фрейми можуть повністю відображатися за кілька кадрів (оверлеїв). **Оверлей** — фрейм, що складається з кількох сторінок, які накладаються одна на одну. У найпростішому випадку для його створення використовується команда **pause**. Ця команда призводить до створення двох сторінок фрейму: на

першій виводиться текст слайду до команди, а на другій додатково буде виведено й текст після команди (до наступної команди **pause**, якщо вона присутня). Приклад створення фрейму, що складається з трьох сторінок:

```
\begin{frame}[t]
\begin{itemize}
\item{Перша сторінка}
\pause
\item{Друга сторінка}
\pause
\item{Третя сторінка}
\end{itemize}
\end{frame}
```

Команди в межах фрейму можуть використовувати специфікації перекриття сторінок: в кутових дужках після команди задається перелік номерів чи діапазонів сторінок фрейму, на яких діє команда чи відображається текст. Наприклад, формування маркованого списку з використанням специфікації перекриття сторінок:

```
\begin{itemize}
\begin{frame}[t]
\item<1-3>{\textbf<1>{Текст присутній на всіх
сторінках
(на першій він виділений жирним)}}
\pause
\item<2>{Текст присутній тільки на другій
сторінці}
\pause
\item<3>{Текст присутній тільки на третій
сторінці}
\end{itemize}
\end{frame}
```

Для організації динамічної зміни елементів фрейму, що розташовано в певному місці слайду, також можна скористатися оточенням **overprint**, як це проілюстровано на прикладі нижче (фрейм складається з двох сторінок, кожна з яких створюється командою **onslide**, і для якої теж задається специфікація перекриття):

```
\begin{frame}[t]
Формування титульної сторінки презентації
\begin{overprint}
```

```
\onslide<1>
Перша сторінка слайду
\onslide<2>
Друга сторінка слайду
\end{overprint}
\end{frame}
```

9.2 Робота з кольором

Для роботи з кольором в преамбулі документу необхідно підключити пакет **\usepackage{xcolor}**. Колір тексту задається командою:

\textcolor{колір}{Текст буде виділено заданим кольором}

тут:

колір — колір тексту англійською мовою (red, orange, yellow, green, cyan, blue, violet).

Будь-який власний колір можна визначити такою командою в преамбулі документу чи в самому документі за межами оточення **frame** (під час вводу команди в межах оточення **frame** він визначений тільки в його межах):

\definecolor{ім'я кольору}{колірна модель}{RGB}

тут:

ім'я кольору — ім'я, що буде використовуватися для доступу до цього кольору;

колірна модель — використовуватимемо RGB-колірну модель, в якій будь-який колір формується як суміш трьох базових кольорів: червоного, зеленого та синього;

RGB — інтенсивність базових кольорів у межах від 0 (мінімальна інтенсивність — кольору немає) до 1 (максимальна інтенсивність). Наприклад:

```
%визначення власного кольору iro
\definecolor{iro}{rgb}{0.1,0.4,0.2}
%застосування власного кольору iro до тексту
\textcolor{col1} {Для цього використано колір iro}
```

Виділити текст кольоровою рамкою з фоном певного кольору можна за допомогою команди: **\fcolorbox{колір рамки}{колір фону}{текст}**

Наприклад:

```
\fcolorbox{red}{green}{Текст буде взято в червону
рамку з зеленим тлом}
```

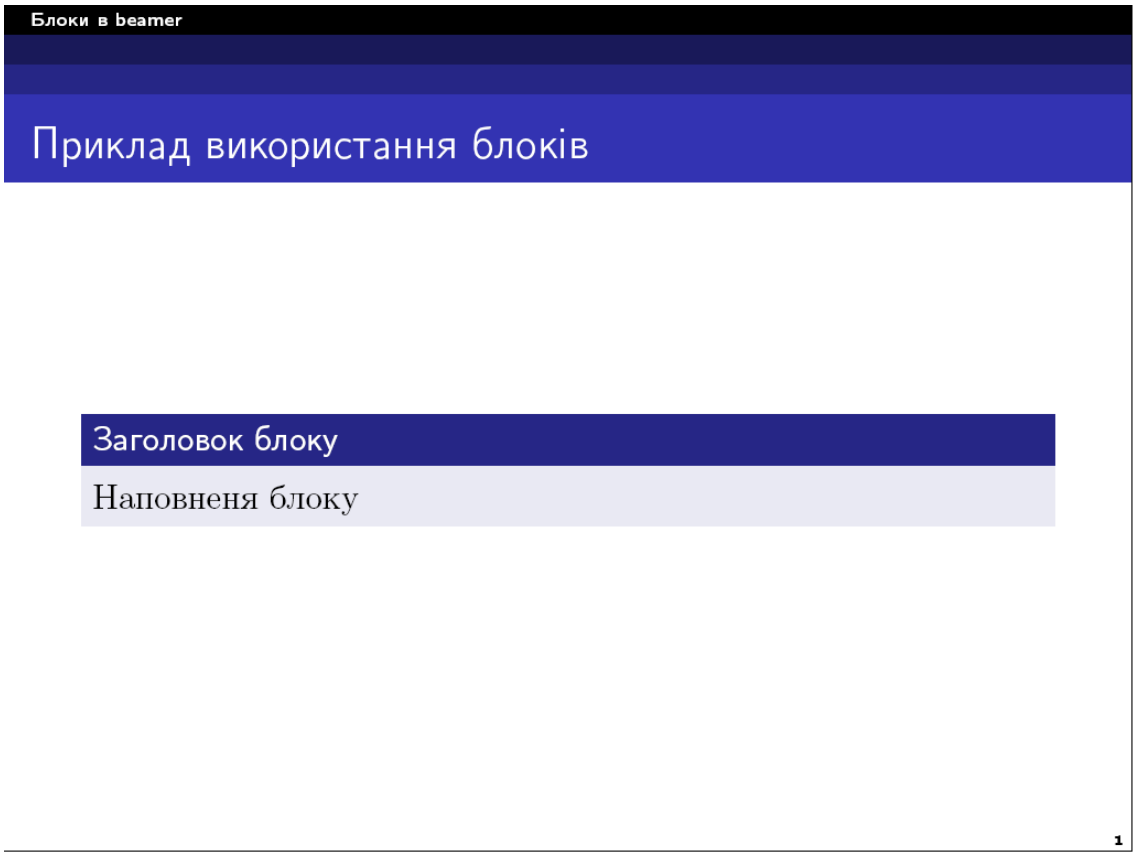
Текст буде взято в червону рамку з зеленим тлом

9.3 Робота з блоками

Для виділення фрагментів слайду зручно використовувати блочні конструкції, які можна сформувати з допомогою оточення **block**:

```
\begin{block}{Заголовок блоку}
Наповненя блоку
\end{block}
```

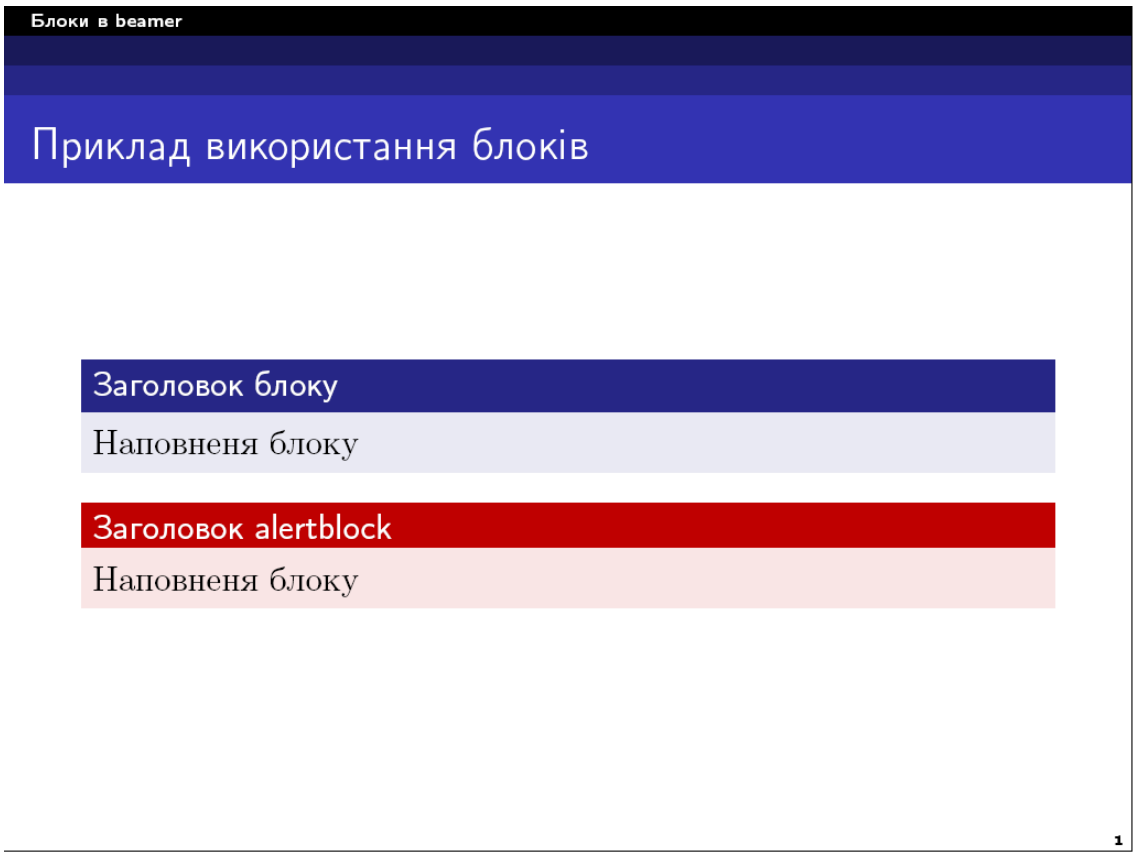
Результат компіляції:



Для виділення більш важливої інформації слугує оточення **alertblock** (від попереднього відрізняється тільки кольоровою гамою):

```
\begin{alertblock}{Заголовок блоку}
Наповненя блоку
\end{alertblock}
```

Результат компіляції:



Для більшої виразності слайду застосовують конструкцію, яка складається з двох блоків, які розташовані поруч.

```
\begin{columns}
  \begin{column}[]{.5\textwidth}
    \begin{block}{Заголовок блоку 1}
   Наповненя блоку 1
    \end{block}
  \end{column}
  \begin{column}[]{.5\textwidth}
    \begin{block}{Заголовок блоку 2}
   Наповненя блоку 2
\end{block}
  \end{column}
\end{columns}
```

Результат компіляції:

Кольорову гаму блоку можна задати в преамбулі документу явним чином за допомогою таких команд:

колір заголовку блоку

\setbeamercolor{block title}{fg=колір1,bg=колір2}

тут:

fg=колір1 — колір шрифту заголовку;

bg=колір2 — задає колір фону для заголовку;

колір тла — **\setbeamercolor{block body}{bg=колір}**.

9.4 Робота з тлом фрейму

Колір тла презентації можна змінювати за допомогою команди:

\beamertemplateshadingbackground{колір1!яскравість1}
{колір2!яскравість2}

тут:

колір1 змінює свою інтенсивність (у відсотках) від значення **яскравість1** у нижній частині і до 0 у верхній частині слайду;

колір2 змінює свою інтенсивність від значення **яскравість2** у верхній частині слайду і до 0 у нижній.

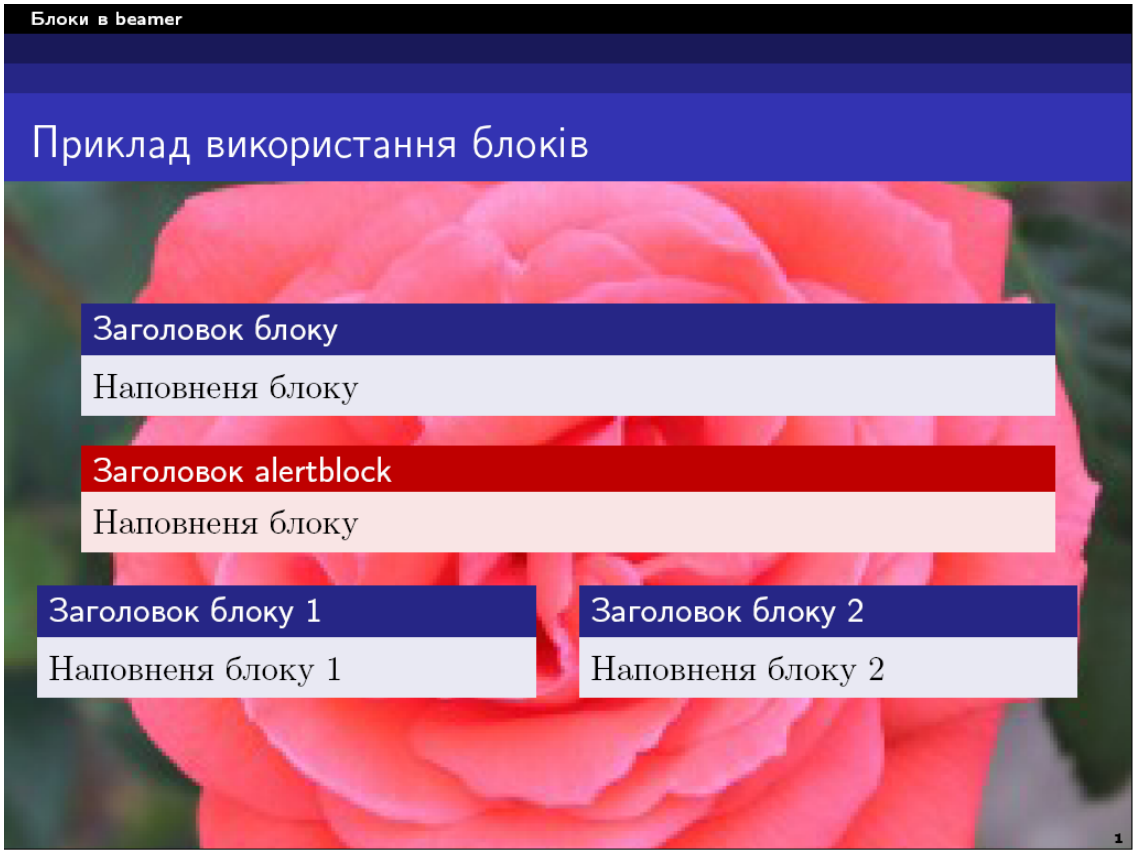
Наприклад:

```
\beamertemplateshadingbackground{green!65}{red!42}
```

Як тло можна використати будь-який малюнок. Для того, щоб він займав всю область слайду, задано ширину, яка дорівнює ширині слайду (**paperwidth**).

Наприклад (команда повинна стояти перед тим оточенням frame, тло для якого вона задає):

```
\usebackgroundtemplate{
{\includegraphics[width=\paperwidth]
{Graf.jpg}}
}
```



9.5 Мультімедійні вставки

Для підвищення зацікавленості аудиторії доцільно додавати до презентації мультімедійні файли.

Такі файли можна додати за допомогою команди **includemovie** з пакету **movie15**. Крім цього пакету, в преамбулі варто підключити пакети **animate,media9**.

Нижче наведено фрагмент презентації з додаванням мультімедійного файлу.

```
\begin{frame}
\begin{figure}
   \includemovie[
                 poster,
```

```
                    autoplay,
                    externalviewer,
                    inline=false,
                text={\small(sample)}
                        ]
{6cm}{6cm}{qwert.avi}
   \end{figure}
\end{frame}
```

9.6 Інтерактивні елементи управління презентацією. Гіперпосилання

За роботу із гіперпосиланнями відповідає пакет **hyperref**, який повинен бути підключений в преамбулі документу. Посилання можуть бути як на елементи в середині самої презентації, так і на зовнішні документи чи сторінки мережі Internet. Для організації переходів в межах презентації необхідно:

- командою **label** присвоїти унікальну мітку сторінці фрейму (мітку можна також задати як параметр до оточення **frame**). Міткою можуть бути тільки будь-які комбінації латинських літер, цифр та символів-розділювачів;
- командою **hyperlink**<**слайди**>**{мітка}{текст}** організувати перехід до мітки.

Параметр <**слайди**> задає сторінки фрейму, на яких відображається текст, на який буде гіперпосилання.

Альтернативною до **label** є команда **hypertarget** <**слайд**> **{мітка} {текст}**. Вона дозволяє задати мітки для певних сторінок в межах фрейму. Текст є гіперпосиланням на слайд, вказаному в параметрі слайду. Він може бути опущеним.

Управляючі кнопки

Команда **beamerbutton{текст}** дозволяє малювати управляючі кнопки. Текст задає напис, що буде розміщено на кнопці. Для налаштування роботи кнопки вона використовується в середині команди створення гіперпосилання.

Наприклад:

```
\hyperlink {a1}{\beamerbutton{На початок}}
```

10 Екзотичні можливості LaTeX

У цьому розділі розглянемо деякі можливості LaTeX, які на перший погляд видаються дивними, але в певних ситуаціях є доцільними.

10.1 Текст

10.1.1 Незвичайне розташування літер

Команда **raisebox** надає можливість зсувати окремі частини тексту, зокрема літери у вертикальному напрямку.

Нижче наведений фрагмент тексту

```
Рух \raisebox{0.39\height}{х}\raisebox{0.77\height}{в}
\raisebox{1.11\height}{и}\raisebox{1.41\height}{л}
\raisebox{1.2\height}{і}\raisebox{1.96\height}{в}
\raisebox{1.6\height}{і}\raisebox{1.96\height}{д}
\raisebox{1.\height}{б}\raisebox{1.66\height}{у}
\raisebox{1.41\height}{в}\raisebox{1.11\height}{а}
\raisebox{0.77\height}{є}\raisebox{0.39\height}{т}{ь}
\raisebox{-0.39\height}{с}\raisebox{-0.77\height}{я}
\raisebox{-1.41\height}{у}\raisebox{-1.85\height}{д}
\raisebox{-1.96\height}{в}\raisebox{-2.\height}{о}
\raisebox{-1.96\height}{х}\raisebox{-1.66\height}{н}
\raisebox{-1.41\height}{а}\raisebox{-1.11\height}{п}
\raisebox{-0.77\height}{р}\raisebox{-0.39\height}{я}мках:
вертикальному (коливання) і повздовжному (розповсюдження).
```

виглядатиме таким чином

Рух х^в^и^л^і ^в^і^д^б^у^в^а^є^т^ь^с^я у двох на^п^р^ямках: вертикальному (коливання) і повздовжному (розповсюдження).

10.1.2 Поворот слова

Команда **rotatebox** надає можливість для поворотів. Слово обертається (проти годинникової стрілки) на заданий кут, який задають як параметр цієї команди.

Повернемо слово **rotatebox**{45}{компіляція} на кут 45 градусів.

Повернемо слово 

на кут 45 градусів.

10.1.3 Підняття та опускання фрагментів тексту

Для того, щоб надрукувати слово чи групу слів над або під рядком, в якому розташовано речення, використовують команду **raisebox{length}{text}**. Ці команда форматує **text** й піднімає його на висоту **lenght**. Від'ємне значення висоти призводить до опускання тексту.

З конструкції

```
\raisebox{1.2}{Кацапи-безбожники}
 та \raisebox{-1.2}{жиди-анцихристи} зняли церковні дзвони, які всіх
 нас єднали не тільки голосом Божим, а й погуком до боротьби з
 московською навалою.
```

в результаті компіляції отримаємо:

Кацапи-безбожники та жиди-анцихристи зняли церковні дзвони, які всіх нас єднали не тільки голосом Божим, а й погуком до боротьби з московською навалою.

Щоб вставити в речення невеличкий абзац, варто скористатистя командою **parbox[position]{width}{text}**

Її зазвичай використовують для того, щоб вставити один чи кілька абзаців у текст.

Опція **position** визначає розташування тексту **text** всередині боксу. Ця опція може набувати значень:

- **c** — текст розташовано в центрі боксу (за замолчаннням);
- **l** — текст притиснуто до лівого краю боксу;
- **r** — текст притиснуто до правого краю боксу.

Опція **width** — визначає ширину боксу. Якщо її не задано, то ширина боксу дорівнює ширині тексту **text**.

Опція **text** — визначає текст, що розташовано всередині боксу. В цій опції можуть міститися команди LaTeX за винятком команд завершення рядка (типу **par**, **begin** тощо).

З конструкції

```
По праву руку від Сєні Кацмана \parbox[b]{4cm}{начальника
Звенигородської повітової ЧК} сидів діжкуватий начальник упродкому
Сиром'ятніков, \parbox{4cm}{із пащеки якого тхнуло, як із жомової
ями}, а лівобіч крутив на всі боки качиною головою начальник
ревкому Долбоносов:  \parbox[t]{4cm}{у нього й справді був
ніс-долото, плескатий і довгий, як у качура}.
```

в результаті компіляції отримаємо (тут навмисно абзаци виділено різними кольорами):

По праву руку від Сєні Кацмана начальника Звенигородської повітової ЧК сидів діжкуватий начальник упродкому Сиром'ятніков, із пащеки якого тхнуло, як із жомової ями, а лівобіч крутив на всі боки качиною головою начальник ревкому Долбоносов: у нього й справді був ніс-долото, плескатий і довгий, як у качура.

10.1.4 Речення незвичайної форми

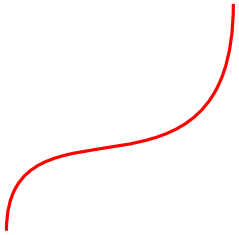
Для створення речень незвичайної форми спершу необхідно створити путь (**path**) вздовж якої вони будуть розташовані.

Така форма речень створюється за допомогою бібліотеки **decoration** пакету **tikz**.

Путь, вздовж якої розташовуватимуться речення, можна утворити, використовуючи криві Без'є, будь-яку геометричну криву або контур геометричної фігури.

Нижче наведено кілька прикладів речень незвичайної форми.

Речення розташоване вздовж кривої Без'є



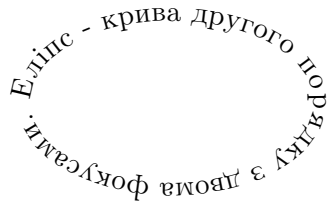
```
\begin{tikzpicture}
\path [decorate,decoration={text along path,
text={З гарним  |\color{yellow}|сонячним|| днем.}}]
(0,0) .. controls (0,2) and (3,0) .. (3,3);
\end{tikzpicture}
```

Речення розташоване по колу радіусу 1 см.

```
  \begin{tikzpicture} [scale=2.5,
decoration={text along path,text={|\color{green}|Весна|| у гості вже
прийшла, І вабить подихом |\color{red}|серця||. Вона дарує кожній
з вас}}]
\draw [decorate] (1,1) circle (1);
\end{tikzpicture}
```

Речення розташоване по еліпсу

```
\begin{tikzpicture}
[decoration={
raise=2pt,text along path, text=
Еліпс - крива другого порядку з двома фокусами.}]
\draw[decorate](0,1) arc (180:-180:1.8cm and 1cm);
\end{tikzpicture}
```



Речення розташоване навколо прямокутника

```
\begin{tikzpicture}
[decoration={
raise=2pt,text along path, text=
{У відповідь швидко з'явилися листівки із закликами не вірити
московським катам, бити на кожному кроці жидо-кацапську комуну.}}]
\draw[decorate](0,0) rectangle (7,5);
\end{tikzpicture}
```

Речення розташоване по спіралі

```
\begin{center} % Центрування вмісту документу
\begin{tikzpicture} % Малюнок tikz
[decoration={text  along path, % Текст вздовж путі
text={ Вітаю з Днем Народження! Бажаю
|\color{red}|Мудрості,|| |\color{green}|радості,||
|\color{red}|Друзів||, благополуччя, енергії, оптимізму,
посмішок, здоров'я, добра, везіння, успіхів! }, % Власне текст
},
scale=4] % Збільшення рисунку
\path [postaction=decorate]
(180:2) \foreach \a in {0,...,12}
{ arc (180-\a*90:90-\a*90:1.5-\a/10) };
\end{tikzpicture} % Закінчення малюнку tikz
\end{center} % Закінчення центрування
```

10.1.5 Абзаци незвичайної форми

Пакет **shapepar** задає команду **shapepar**, яка дозволяє набирати абзаци незвичаного вигляду. Загальний розмір регулюється автоматично таким чином, що текст заповнює всю задану фігуру. Такі абзаци форматуються за кілька ітерацій, поки його розмір і форма не стануть правильними. Пакет бажано застосовувати для листівок чи запрошень.

Команду **shapepar** варто розташовувати на початку абзаца, й тоді вона діє на весь абзац:

shapepar{shapespec} текст абзацу

Параметр **shapespec** описує форму абзацу.

Існує кілька заданих форм

diamondpar — абзац у вигляді діаманту

squarepar — абзац у вигляді квадрату

heartpar — абзац у вигляді серця

nutshape — абзац у вигляді гайки

circleshape — абзац у вигляді кола

CDshape — абзац у вигляді кола з діркою (схоже на CD-диск)

starshape — абзац у вигляді п'ятикутної зірки

hexagonshape — абзац у вигляді шестикутника

rectangleshape{height}{width} — абзац у вигляді прямокутника

Більше форм можна знайти у файлах, які позначають як ***shape.def**. Такі файли можна створити для будь-якої форми.

Відомі такі форми:

dropshape — краплі дощу;

triangleshapes — трикутники різної орієнтації;

candleshape — горяща свічка;

TeXshape — логотип TEX;

Canflagshape — канадійський прапор

Нижче наведено кілька форм таких, які містять однаковий текст.

♢

Ме-
тою ство-
рення ЕОР
є модернізація
освіти, змістове на-
повнення освітнього про-
стору, забезпечення рівно-
го доступу учасників навчально-
виховного процесу незалежно від мі-
сця їх проживання та форми на-
вчання до якісних навчальних
та методичних матеріалів,
створених на осно-
ві інформаційно-
комунікаційних
техноло-
гій.

♢

Метою створення ЕОР є модернізація освіти, змістове наповнення освітнього простору, забезпечення рівного доступу учасників навчально-виховного процесу незалежно від місця їх проживання та форми навчання до якісних навчальних та методичних матеріалів, створених на основі інформаційно-комунікаційних технологій.

Метою створення ЕОР є модернізація освіти, змістове наповнення освітнього простору, забезпечення рівного доступу учасників навчально-виховного процесу незалежно від місця їх проживання та форми навчання до якісних навчальних та методичних матеріалів, створених на основі інформаційно-комунікаційних технологій.

♡

Метою створення ЕОР є модернізація освіти, змістове наповнення освітнього простору, забезпечення рівного доступу учасників навчально-виховного процесу незалежно від місця їх проживання та форми навчання до якісних навчальних та методичних матеріалів, створених на основі інформаційно-комунікаційних технологій.

Метою створення ЕОР є модернізація освіти, змістове наповнення освітнього простору, забезпечення рівного доступу учасників навчально-виховного процесу незалежно від місця їх проживання та форми навчання до якісних навчальних та методичних матеріалів, створених на основі інформаційно-комунікаційних технологій.

10.2 Календарі

Пакет **tikz** разом бібліотекою **calendar** надає можливість створювати календарі різноманітних форм. Опис роботи з цією бібліотекою взято з [13].

Бібліотека **calendar** визначає команду **calendar**, яку можна використати для створення календарів. Команду **calendar** реалізовано на команді **pgfcalendar** з пакету **pgfcalendar**, який завантажується автоматично.

Команда **calendar** має повну конфігурацію, дозволяючи створювати різноманітні календарі будь-якого вигляду.

Таким чином, основною командою для створення календарів у **TikZ** є команда **calendar**. Ця команда може виконуватися тільки в оточенні **{tikzpicture}** подібно до команди **draw**.

Синтаксис цієї команди є подібним до синтаксису таких команд як **node** чи **matrix**.

calendar⟨**специфікація**⟩

Специфікація має бути набором елементів, кожен з яких є однією з таких структур:

- **[⟨опції⟩]** Ви додаєте ⟨**опції**⟩ в квадратних дужках як **[red,draw=none]**. Ці ⟨**опції**⟩ можуть бути будь-якими опціями **TikZ** і вони застосовуються для всього календаря. Ви можете додавати опції кілька разів, ефект накопичується.
- **(⟨ім'я⟩)** Це має такий самий ефект як **[name=⟨ім'я⟩]**. Зауважте, що *календар не вузол*, а **(⟨ім'я⟩)** *не є іменем вузла.*
- **at (⟨координати⟩)** Це має такий самий ефект як **[at=⟨координати⟩]**.
- **if (⟨умова для дати⟩) ⟨опції чи команди⟩ else ⟨else опції чи команди⟩**

На початку будь-якого календаря використовують стиль
\tikz\every calendar

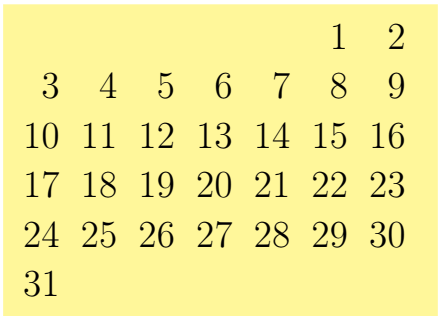
Діапазон дат. Основний ефект команди **\calendar** полягає в тому, щоб виконати код для кожного дня з діапазона дат. Цей діапазон дат визначають, використовуючи таку опцію:

\tikz\dates=⟨початкова дата⟩ to ⟨кінцева дата⟩

Ця опція визначає діапазон дат. Ви можете визначити дати за форматом ISO як 2021-09-12. Ви можете замінити день місяця на **last**, щоб визначити останній день місяця (2021-10-last теж саме, що й 2021-10-31). Ви можете додати знак +, за яким розташовують число, для визначення зсуву (так 2022-01-01+-1 теж саме, що й 2021-12-31), причому, якщо число від'ємне, то дату відраховують назад, якщо додпьне — вперед.

Добре використовувати дві частини для таких описів: Команда **\calendar** здійснює ітерацію дат в діапазоні. *Current date* має відношення до поточної дати, яку обробляє команда, і так відбувається для всіх дат. Код виконується для кожної поточної дати, який називають *current date code*. Код поточної дати містить дві різні частини.

Центральною частиною кода поточної дати є виконання коду **\tikzdaycode**. За замовчанням цей код просто утворює вузол дня місяця. Це означає, що неможлива подальша дія, тому що всі дні календаря накладатимуться один на одний! Для запобігання цьому ви маєте модифікувати поточну дату через зсув перепризначених днів. Перевизначивши порядок як **day list downward** або **week list**, але ви можете визначити властий порядок. З визначення порядку за допомогою трюків з бітами. Для початку розглянемо перевизначення дат для нашого першого календаря.

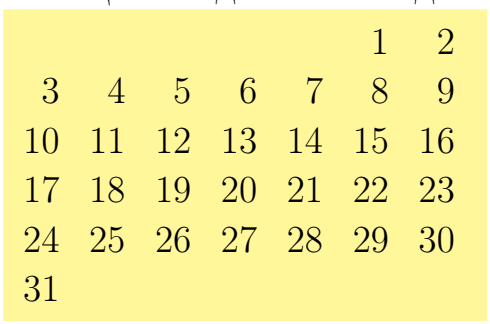



```
\tikz\calendar[dates=2022-01-01 to 2022-01-31,week list];
```

Зміна відстані. У вище наведеному календарі відстань між днями визначають числовими опціями. Більшість розташувань не використовує всі з цих опцій, а тільки ті, що природно застосовують.

\tikz\day xshift=⟨відстань⟩

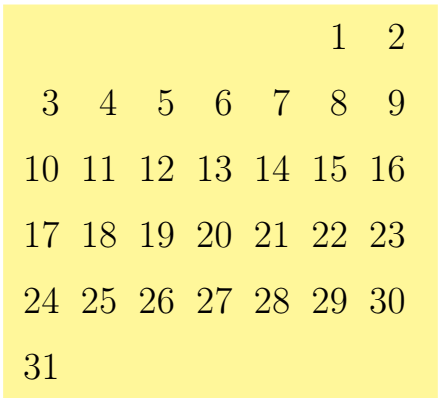
Початкова відстань дорівнює 3.5ex. Це визначає горизонтальну відстань між днями. Це не відстань між днями, а відстань між якорями їхніх вузлів.




```
\tikz\calendar[dates=2022-01-01 to 2022-01-31,week list, day xshift=4ex];
```

\tikz\day yshift=⟨відстань⟩

Початкова відстань дорівнює 3ex. Це визначає вертикальну відстань між днями. Знову ж таки це не відстань між днями, а відстань між якорями їхніх вузлів.



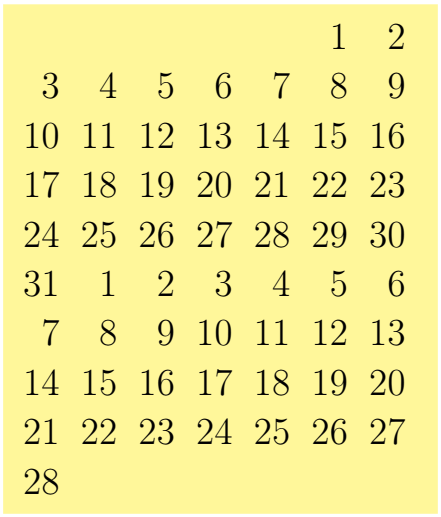

```
\tikz\calendar[dates=2022-01-01 to 2022-01-31,week list, day yshift=4ex];
```

\tikz\month xshift=⟨відстань⟩

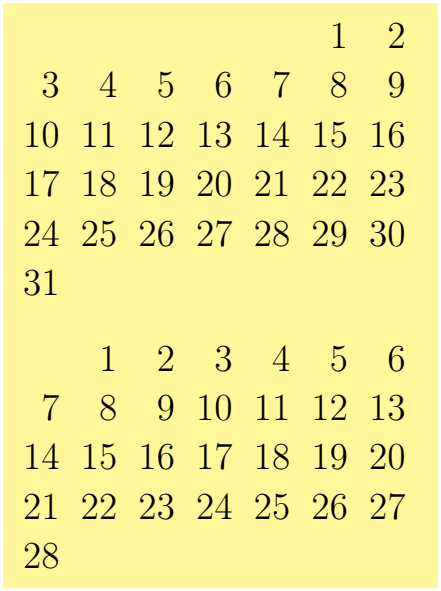
Це визначає додаткову горизонтальну відстань між місяцями.

\tikz\month yshift=⟨відстань⟩

Це визначає додаткову вертикальну відстань між місяцями.



```
\tikz\calendar[dates=2022-01-01 to 2022-02-last,week list, month yshift=0pt];
```



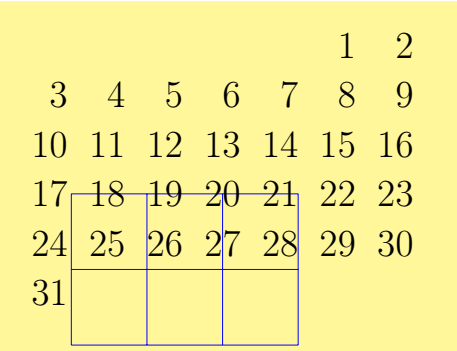
```
\tikz\calendar[dates=2022-01-01 to 2022-02-last,week list, month yshift=1cm];
```

Зміна позиції календаря.

Календар розміщено таким чином, що зазвичай якір мітки першого дня знаходиться біля початку. Це можна змінити за допомогою опції **at**. Коли ви задаєте **at = {(1,1)}**, цей якір першого дня буде знаходитися за координатами (1; 1).

Загалом, порядки не завжди розташовують якір першого дня біля місця початку. Іноді додаткові правила встановлення інтервалу заважають. Існують різні способи вирішення цієї проблеми: по-перше, ви можете просто ігнорувати це. Оскільки календарі часто розташовують у їх власній **tikzpicture**, а оскільки їх розмір, якщо його обчислює комп'ютер автоматично, то точне розташування початку координат часто взагалі не має значення. По-друге, ви можете розташувати календар всередині вузла, як у **. . . node {\tikz\calendar . . . }**. Це дозволяє розташувати вузол звичайним способом, використовуючи якір вузла. По-третє, ви можете бути дуже розумним і використовувати одноклітинну матрицю. Перевага полягає в тому, що матриця дозволяє надавати будь-який якір

будь-якого вузла всередині матриці як якір для всієї матриці. Наприклад, наступний календар розташовано таким чином, що центр 2022-01-20 знаходиться за координатами (2; 2):




```
\begin{tikzpicture}
\draw[help lines] (0,0) grid (3,2);
\matrix [anchor=cal-2000-01-20.center] at (2,2)
{\calendar(cal)[dates=2000-01-01 to 2000-01-31,week list]; };
\end {tikzpicture}
```

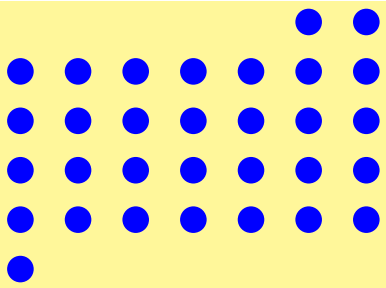
Нажаль, позиції, які базуються на матрицях і є найчистішою можливістю, не такі мобільні як інші підходи (наприклад, вони прямо не працюють з SVG).

Зміна зовнішнього вигляду днів.

Як згадувалося перед цим, кожен день у попередньому календарі створювали через виконання **\tikzdaycode**. Кожного разу цей код виконувався, система координат визнаяалася знову, щоб розташувати день місяця правильно. Ви можете змінити як код, так і його зовнішній вигляд, використовуючи накі опції.

\tikz\day code=⟨код⟩

Ця опція дозволяє вам змінити код таким чином, що він виконуватиметься для кожного дня.



```
\tikz\calendar[dates=2022-01-01 to 2022-01-last,week list, day code=\fill[blue] (0,0) circle (5pt);];
```

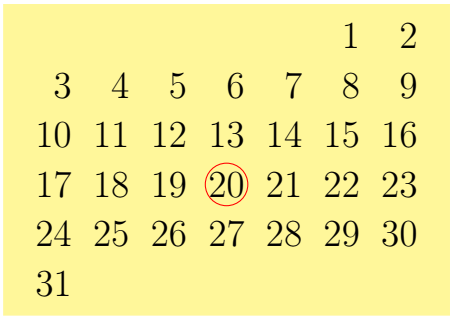
За замовчанням код є таким:

```
\node[name=\pgfcalendarsuggestedname,every day]{\tikzdaytext};
```

Перша частина показує вузли днів, які утворюються через такі імена: Якщо ⟨**name**⟩ є іменем, яке дали календареві через опцію **name=** або через елемент специфікації (⟨**name**⟩), тоді **pgfcalendarsuggestedname** розширюватиметься до ⟨**name**⟩ – ⟨**date**⟩, де ⟨**date**⟩ є датою того дня, який обробляють у форматі

ISO.

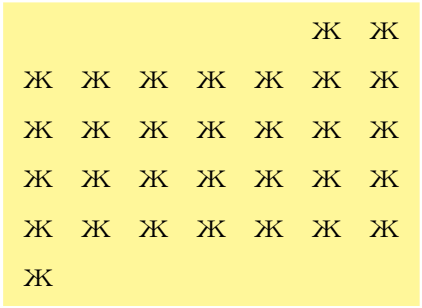
Напрклад, якщо обробляють дату 1 січня 2022 і календар назвали **mycal**, тоді узол, що містить 1 для цієї дати, називатиметься **mycal–2022–01–01**. Ви можете пізніше послатися на цей вузол.




```
\begin{tikzpicture}
\calendar (mycal) [dates=2022-01-01 to 2022-01-31,week list];
\draw[red] (mycal-2022-01-20) circle (8pt);
\end {tikzpicture}
```

\tikz\day text=⟨текст⟩

Опція змінює визначення **\tikzdaytext**. За замовчанням макрос просто повертає поточний день місяця, але ви можете змінити це задовільно. Ось дурний приклад:




```
\tikz\calendar[dates=2022-01-01 to 2022-01-last,week list, day text=ж];
```

Більш корисні приклади базуються на використанні команди \%. Ця команда перевизначає всередині **\pgfcalendar**, щоб позначити теж саме, що й **\pgfcalendarshorthand**. (Початкове значення \% губиться в календарі, і ви мусите зберігти перед календарем, якщо воно вам дійнсо потрібне.)

\% вставляє поточний день/місяць/рік/день тижня у певному форматі в текст. Перша літера, яка розташована за \% визначає тип (дозволеними є такі значення **d, m, y, w**), друга літера визначає яким чином ця величина відображатиметься (**-** означає, що чисельно, = означає, що чисельно з провідним пробілом, **0** означає, що чисельно з провідним нулем, **t** означає, що у вигляді тексту, **.** означає, що у вигляді тексту, як абревіатура). Наприклад, **\%d0** дає день з провідним нулем.

Давайте перевизначемо день таким чином, щоб отримати день з провідним нулем:

					01	02
03	04	05	06	07	08	09
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

```
\tikz\calendar[dates=2022-01-01 to 2022-01-last,week list, day text=%d0];
```

\tikz\every day (початково anchor=base east)

Цей ситль робить код вузла за замовчанням для кожного дня. Стиль кожного дня є корисним для зміни способу подання дня. Наприклад, давайте зробимо всі дні червоними:

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

```
\tikz[every day/.style=red] \calendar[dates=2022-01-01 to 2022-01-last,week list];
```

Зміна зовнішнього вигляду міток місяцю та року.

На додачу до днів календаря, мітки місяців і навіть років (для дійсно довгих календарів) можна додавати. Ці мітки додаються тільки один раз на місяць чи рік і це не відбувається за замовчуванням. Для цього використовують спеціальні стилі, починаючи з розташування мітки місяця і роблять її видимою:

January

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

February

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

```
\tikz \calendar[dates=2022-01-01 to 2022-02-last,week list, month label above centered];
```

Наступні опції змінюють вигляд мітки місяця чи року:

\tikz\month code=⟨код⟩

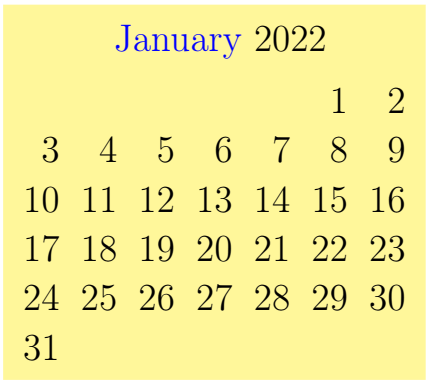
Ця опція дозволяє вам визначити, що макрос **\tikzmonthcode** має розширювати. З замовчанням **\tikzmonthcode** встановлено як

```
\node[every month]{\tikzmonthtext};
```

Зауважте, що цей вузол не має імені.

\tikz\month text=⟨текст⟩

Ця опція одзволяє вам змінити макрос **\tikzmouthtext**. За замовчанням текст місяця довгою текстовою презентацією поточного місяця.




```
\tikz \calendar[dates=2022-01-01 to 2022-01-last,week list, month label above centered,
month text = \textcolor{blue}{\%mt} \%y-];
```

\tikz\every month

Цей стиль можна використовувати для зміни вигляду мітки місяця.

\tikz\year code= ⟨код⟩

Працює як код місяця, тільки для років.

\tikz\year text=⟨текст⟩

Працює як код місяця, тільки для років.

\tikz\every year

Працює як **every month**, тільки для років.

Команди if для роботи з датами

Потужність команди **\calrndar** задежить від використання умов. Існує дві однакових путі специфікувати такі умови. Перша, ви можете додати текст **if (⟨умови⟩) ⟨код чи опції⟩** для вашого ⟨**calendar specification**⟩, який можливо містить **else ⟨код чи опції⟩**. Ви можете мати багато таких умов (але ви не можете їх просто вкладати). Друга путь використовувати таку опцію:

\tikz\if = (⟨умови⟩) ⟨код чи опції⟩ else ⟨код чи опції else ⟩

Ця опція має такий самий ефект, як і **if** у ⟨**calendar specification**⟩. Опція є найбільш використовуваною у **every calendar style**, де ви не можете забезпечити умови **if** по-іншому.

Тепер, незалежно від того, як ви визначаєте умови, це дає такий ефект (індивідуально і незалежно для кожної дата у календарі):

1. Перевіряють, чи є поточна дата однією з умов, що прописано в ⟨**conditions**⟩. Прикладом такої умови є **неділя**. Отже, коли ми записуємо **if(Saturday, Sunday) foo**, то **foo** виконуватиметься для кожного дня в календарі, який є суботою чи неділею.

   Команду **ifdate** і таким чином **pgfcalendarifdate** використовують для перевірки ⟨**conditions**⟩. Найкориснішими тестами є: Тести як понеділок тощо, **workday** для днів з понеділка по п'ятницю, **weekend** для суботи та неділі, **equals** для тестів, коли порвінюють поточну двту із заданою, **at least** для порівняння поточної дати із заданою.

2. Якщо дата пройшла перевірку, ⟨**код чи опції**⟩ виконується, щоб описати цей момент; якщо дата не пройшла перевірку, то виконується ⟨**код чи опції else** ⟩ якщо таке є. ⟨**код чи опції**⟩ можуть бути таким же самим кодом. Це визначає оточуючий код з фігурними дужками. Це може бути також список опцій **TikZ**. Це визначає оточення опцій в квадратних дужках. Наприклад, під час тестування дати **if (Sunday) {\draw. . . } else {\fill. . . }** виконуються частина коду. Для порівняння, **if (Sunday) [red] else [green]** виконуються дві опції.

Якшо ⟨**код чи опції**⟩ є кодом, то він просто виконується (для поточної дати). Якшо це список опцій, то ці опції передаються до оточення поточної дати.

Давайте тепер розглянемо кілька прикладів. Спершу, ми вкористаємо умову щоб зробити всі неділі червоними.

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

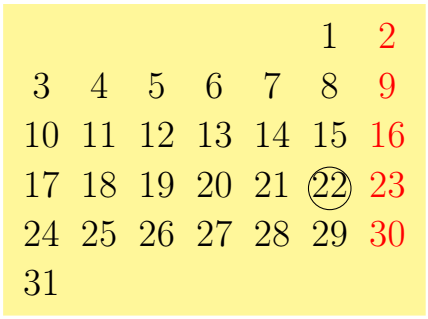
```
\tikz \calendar[dates=2022-01-01 to 2022-01-last] if (Sunday) [red];
```

Потім давайте зробимо щось на певній даті.

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

```
\tikz \calendar[dates=2022-01-01 to 2022-01-last]
if (Sunday) [red] if (equals=2022-01-22) {\draw (0,0) circle (8pt);};
```

Ви можете здивуватися чому коло зсунуто відносно дати. Дійсно воно центрується на даті, справа в тому, що мітка дати використовує базовий якір **east**, який зсуває зсуває мітку вгору праворуч. Щоб подолати цю проблему, ми можемо змінити якір:



```
\tikz[every day/.style={anchor=mid}] \calendar[dates=2022-01-01 to 2022-01-last]
if (Sunday) [red] if (equals=2022-01-22) {\draw (0,0) circle (8pt);};
```

Однак, дати простих днів тепер не подовжилися, щоб бути виставленими правильно. Для цього ми можемо змінити текст дня **%d**=, який додає простір на початку тексту простого дня.

В подальшому більше технічної інформації буде подано. Більшість читачів можливо бажає пропустити це.

Код поточної дати Як згадано раніше для кожної дати в календарі виконується код поточної дати. Це робота такого коду зсувати навкруг дати вузли, для встановлення вузлів дат, щоб намалювати мітки місяця і зробити все інше, що є необхідним для створення калелндаря.

Код поточної дати складається з таких частин, в такому порядку:

1. Перед кодом **scope**
2. Відкриття **scope**
3. Початку коду **scope**
4. Всі **if-дати** з ⟨**calendar specification**⟩ виконуються
5. Закінчення коду **scope**
6. Закриття **scope**
7. Після коду **scope**

Всі коди, про які згадувалося вище, можуть бути змінені, використовуючи призначені опції, дивіться нижче. У випадку, коли ви дивуєтесь чому так багато є небохідного, зрозуміло, що код поточної дати всілому не оточений **scope** чи TEX групою. Це означає, що код, який аиконується в оточення **scope** має вплив на всі наступні дні. Наприклад, якщо код після коду **scope** змінюється

перетворення матриці, зсунувши все вниз, усі наступні дні будуть зсунуті вниз. Якщо кожен день зробить це, ви отримаєте список днів один за одним.

Однак не завжди потрібно, щоб код впливав на все, що далі. Наприклад, якщо день має if-дату **if (Sunday) [red]**, ми хочемо, щоб ця неділя стала червоною, але не всі наступні дні. Подібним чином іноді легше обчислити позицію дня щодо фіксованого початку, і ми не хочемо, щоб будь-які зміни матриці перетворення мали ефект за межами області.

Завдяки розумній корекції різних кодів можливі всілякі різноманітні перевизначення днів.

/tikz/execute before day scope=⟨code⟩

⟨**code**⟩ виконується перед усім іншим для кожної дати. Багаторазовий виклик цієї опції має накопичувальний ефект. Отже, якщо використовувати цю опцію двічи, код після першого використання є першим для використання для кожного дня слідом за кодом, який подано другим.

/tikz/execute at begin day scope=⟨code⟩

Цей код виконується перед усім іншим у межах поточної дати.

/tikz/execute at end day scope=⟨code⟩

Цей код виконується безпосередньо перед закриттям оточення поточної дати. Ефект є також накопичувальним, однак в зворотному порядку. Це корисно для пари **scope** і **endcope** в коді на початку та в кінці коду.

/tikz/execute after day scope=⟨code⟩

Це виконується в самому кінці поточної дати, за межами області застосування. Накопичення також відбувається в зворотному порядку.

В решті наступних підрозділів ми розглянемо, як різні коди застосування можуть бути використані для створення різноманітних компонувань календаря.

10.2.1 Створення простого списку днів

Ми починаємо зі списку днів календаря, де один день нижче іншого. Для цього ми просто зсуваємо координати системи вниз у кінець коду для кожного дня. Такий зсув має бути поза оточенням дня через те, що ми бажаємо, щоб зсуви накопичувалися. Отже, ми використовуємо такий код:

```
1
2
3
4
5
6
7
8
```

```
\tikz\calendar[dates=2022-01-01 to 2022-01-08,
execute after day scope={\pgftransformyshift{-1em}}];
```

Зрозуміло, що ми можемо використати такий підхід для створення списку днів, який зростає вгору, вниз, праворуч, ліворуч або навіть по діагоналі.

10.2.2 Додавання мітки місяця

Зараз ми бажаємо додати мітку місяця ліворуч на початку кожного місяця. Ідея полягає в тому, щоб зробити дві речі:

1. Ми додаємо код, який виконується тільки на початку кожного місяця.
2. Код виконується перед тим як виводиться актуальна дата. З цього випливає, що опції, які мають відношення до днів, не впливатимуть на відображення місяця.

Ми маємо дві опції, де ми зможемо додати код місяця: або ми додаємо це на початку оточення днів або перед. Будь-який варіант працюватиме правильно, але можливо безпечніше розташувати такий код в оточенні, щоб гарантувати, що налаштування випадково «не спрацюють поза оточенням».

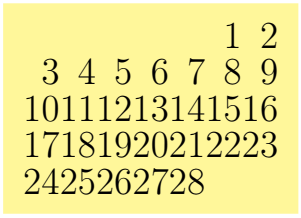
```
\tikz\calendar[dates=2022-01-01 to 2022-01-08,
execute after day scope={\pgftransformyshift{-1em}},
execute at begin day scope={\ifdate{day of month=1}{\tikzmonthcode}{}},
every month/.append style={anchor=base east,xshift=-2em}];
```

У коді, що наведено вище, ми використали команду **\ifdate{$\langle$умова$\rangle$}{$\langle$умова then$\rangle$}{$\langle$умова else$\rangle$}**, яка має багато подібних ефектів як **\ifdate{$\lang{$\langle$умова then$\rangle$}{$\langle$умова else$\rangle$}**, але працює в нормальному коді.

10.2.3 Створення розташування тижневого списку

Давайте тепер адресуємо складніше розташування: Тижневий список. У цьому розташуванні є рядок для кожного тижня. Горизонтальне розташування днів зроблено таким чином, що понеділки знаходяться один під одним, так само і для вівторків і так далі.

Для того, щоб утворити таке розташування ми можемо використати такий підхід: На початку координатної системи залишаємо якір для понеділків кожного тижня. Це означає, що в кінці кожного тижня початок пересувається нижче на один рядок. На всі інші дні початок в кінці коду дня залишається таким як і на початку. Для того, щоб коректно розташувати кожень день, ми використовуємо код на опчатку і в кінці оточення дня з горизонтальним зсувом здя згідно до його номеру в днях тижня.




```
\tikz\calendar[dates=2022-01-01 to 2022-01-28,
% кожен день зсунуто праворуч згідно дню тижня
execute at begin day scope={\pgftransformxshift{\pgfcalendarcurrentweekday em}},
% після кожного тижня, початок зсунути вниз
```

10.2.4 Створення розташування місячного списку

Для іншого прикладу давайте створимо список, що складається з одного рядка для кожного місяця. Це досить просто зробити для тижнів, поки ми не додамо такі вимоги: Знову ж таки ми хочемо щоб всі дні в колонках мали однаковий день тижня. Однак місяці починаються з різних днів тижня, це означає, що кожен рядок має індивідуальний зсув.

Одна з можливостей це використати такий підхід: Після кожного місяця (або на початку кожного місяця) ми здійснювати зсув по вертикалі на один рядок. Для горизонтального розташування в оточенні дня ми локально зсуватимемо день до його стану в місяці. Крім того, ми мусимо додатково зсувати день, щоб гарантувати, що перший день місяця розташується у правильній колонці днів тижня. Для цього ми запам'ятовуємо цей день тижня перший раз як його побачимо.

```
          1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
  1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282
  1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
        1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930
            1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
    1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930
        1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
      1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930
          1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
  1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930
      1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930 31
```

```
\newcount\mycount \tikz\calendar[dates=2022-01-01 to 2022-12-last,
execute before day scope={\ifdate{day of month=1} {
% Запам'ятовуємо день тижня першого дня місяця
\mycount=\pgfcalendarcurrentweekday
% Зсуваємо вниз
\pgftransformyshift{-1em}}{}},
execute at begin day scope={
% Кожен день зсуваємо праворуч відповідно до дня місяця
\pgftransformxshift{\pgfcalendarcurrentday em}
% і додаємо до відповідного першого дня тижня.
\pgftransformxshift{\the\mycount em}}];
```

10.2.5 Розташування

Розташування визначає яким чином дні календаря розташовано на сторінці. Бібліотека **calendar** визначає ряд розташувань.

Ми почнемо з розташувань, в яких дні перелічено вздовж рядка.

```
/tikz/day list downward
```

Цей стиль розташовує дні місяця один нижче іншого. Зсув між днями визначає **yshift**. Між місяцями додається додатковий місячний зсув **yshift**.

28
29
30
31

1
2
3
4
5

```
\tikz\calendar[dates=2022-01-28 to 2022-02-05, day list downward, month yshift=1em];
```

/tikz/day list upward

Працює як і попереднє, тільки порядок зростання зворотній.

5
4
3
2
1

31
30
29
28

```
\tikz\calendar[dates=2022-01-28 to 2022-02-05, day list upward, month yshift=1em];
```

/tikz/day list right

Цей стиль також працює як попередні, але список днів зростає праворуч.

28 29 30 31 1 2 3 4 5

```
\tikz\calendar[dates=2022-01-28 to 2022-02-05, day list right, month yshift=1em];
```

/tikz/day list left

Теж що і попереднє, тільки список днів зростає ліворуч.

5 4 3 2 1 31 30 29 28

```
\tikz\calendar[dates=2022-01-28 to 2022-02-05, day list left, month yshift=1em];
```

Наступне розташування списку днів у тижні.

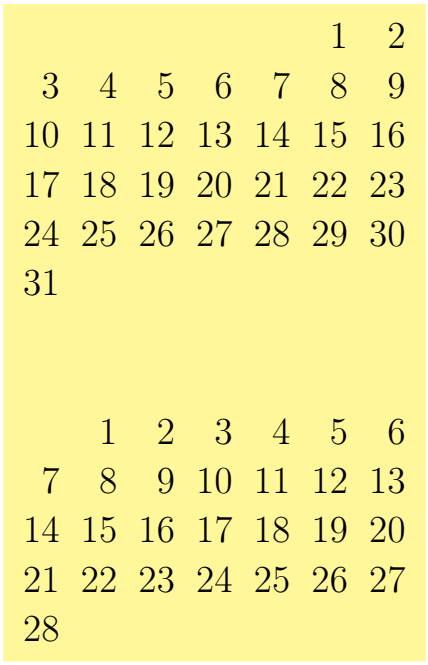
/tikz/week list

Цей стиль створює один рядок для кожного тижня в діапазоні. Величину зсуву днів **xshift** використовують для відстані між днями в тижня у кожному

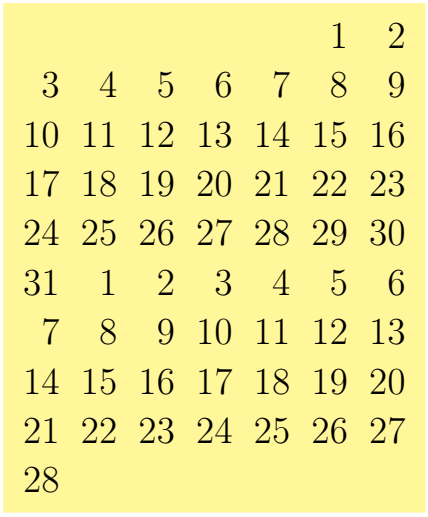
рядку, величину зсуву днів **yshift** використовують для відстані між рядками. В обох випадках «відстань» визначає відстань між якорями вузлів днів (або більш загально відстань між початком маленької картинки, яку створюють для кожного дня).

Дні у кожному тижні зсунуті так, що понеділок завжди знаходиться у першій позиції (щоб змінити це ми мусимо копіювати і оптім модифікувати код). Якщо діапазон не поченається з понеділка, перший рядок не починається у першій колонці, але у стовпчику, який відповідає першій даті діапазону.

На початку кожного місяця (за виключенням першого місяця в діапазоні) додається додатковий вертикальний прстір місяця **yshift**. Якщо це має значення 0pt, то ми отримаємо неперервний список днів.



```
\tikz\calendar[dates=2022-01-01 to 2022-02-last, week list];
```

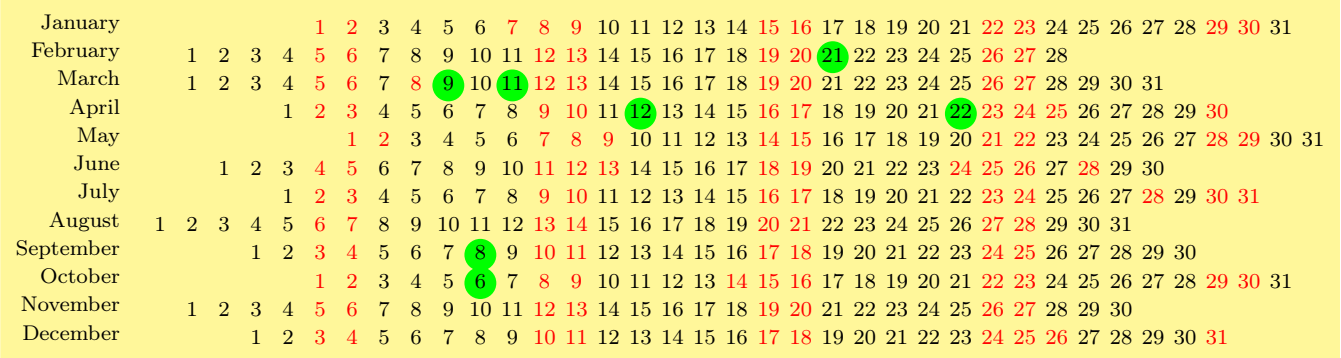


```
\tikz\calendar[dates=2022-01-01 to 2022-02-last, week list, month yshift=0pt];
```

Наступне розташування дає дуже компактний вигляд цілого року.

/tikz/month list

В уьому розташуванні є рядок для кожного місяця. Як у списку тижня **xshift** для дня використовують по горизонталі. Для вертикального зсуву використовують **yshift** для місяців.




```
\scriptsize \tikz[every day/.style={anchor=mid}]\calendar[dates=2022-01-01 to 2022-31-last,
month list, month label left, month yshift=1.25em]
  if(Saturday,Sunday) [red]
  % Свята
  if(equals=2022-01-07, equals=2022-03-08, equals=2022-05-09,equals=2022-06-28,
equals=2022-07-28,equals=2022-06-24,equals=2022-10-14) [red]
  % Понеділки після свят
  if(equals=2022-04-25, equals=2022-05-02,equals=2022-06-13,equals=2022-12-26)
[red]
  % Дні народження
  if(equals=2022-02-21, equals=2022-03-09, equals=2022-03-11, equals=2022-04-12,
equals=2022-04-12, equals=2022-04-22, equals=2022-09-08, equals=2022-10-06)
  {\fill[green] (0,0) circle (6pt);};
```

10.2.6 Мітки місяця

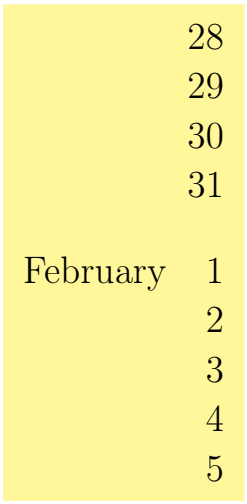
Для багатьох календаоів ви можете забажати додавати мітки кожного місяця. Ми вже розглянули як створювати вузли місяців і відтворювати в описі команди **calendar**: використовувати **month text, every month** і також якщо неодбідно **код місяця** для зміни зовнішнього вигляду міток місяця.

Ми ще не розглядали де знаходяться мітки. За замовчанням вони не виводяться взагалі і це не дуже добре для «опзиції замовчування». Взамін ви можете використати одну з таких опцій для визначення позиції для мітки:

```
/tikz/ month label left
```

Розташовує мітку місяця лворуч від першого дня місяця. (Для списку тижня і списку місяця, де місяць не починається у понеділок позиція вибирається таким чином ніби-то місяць розпочався у понеділок — що є таким як ми зазвичай

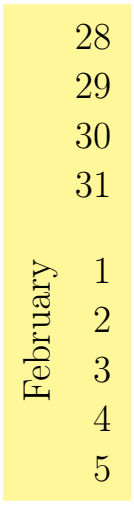
хочемо.)




```
\tikz \calendar[dates=2022-01-28 to 2022-02-05, day list downward,
month yshift=1em, month label left];
```

```
/tikz/ month label left vertical
```

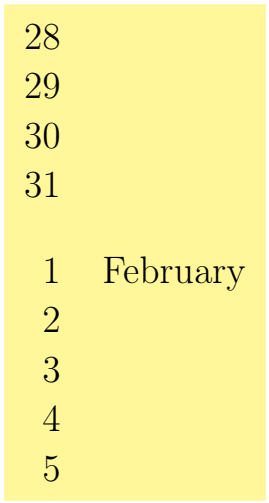
Стиль працює як вище згаданий, тільки мітка розгортається проти годинникової стрілки на 90 градусів.




```
\tikz \calendar[dates=2022-01-28 to 2022-02-05, day list downward,
month yshift=1em, month label left vertical];
```

```
/tikz/ month label right
```

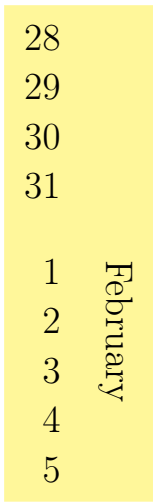
Цей стиль розташовує мітку місяця праворуч у рядку, де знаходиться перший день місяця. Це означає, що для списку днів мітка знаходиться праворуч від першого дня, для списку тижнів вона також праворуч від першого тижня, а для списку місяців — правопуч всіх місяців.




```
\tikz \calendar[dates=2022-01-28 to 2022-02-05, day list downward,
month yshift=1em, month label right];
```

`/tikz/ month label right vertical`

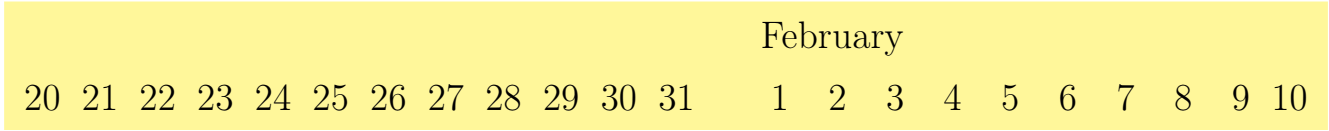
Працює як і вище згадана опція, тільки мітка повернута за годинниковою стрілкою на 90 градусів




```
\tikz \calendar[dates=2022-01-28 to 2022-02-05, day list downward,
month yshift=1em, month label right vertical];
```

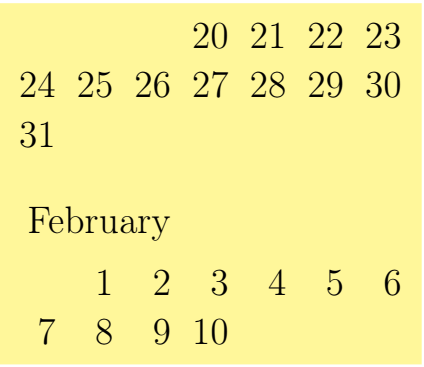
`/tikz/ month label above left`

Цей стиль розташовує мітку місяця вище рядка з першим днем виоівнюючи ліворуч по нійлівішому стовпчику. При чому мітка підіймається на фіксовану відстань **1.25em;** використовують опцію **yshift** з місячним вузлом для модифікації цього.




```
\tikz \calendar[dates=2022-01-20 to 2022-02-10, day list right,
```

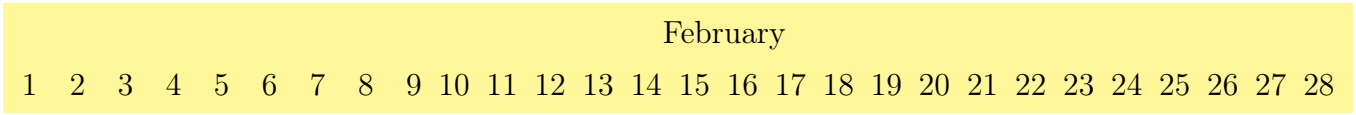
month xshift=1em, month label above left];



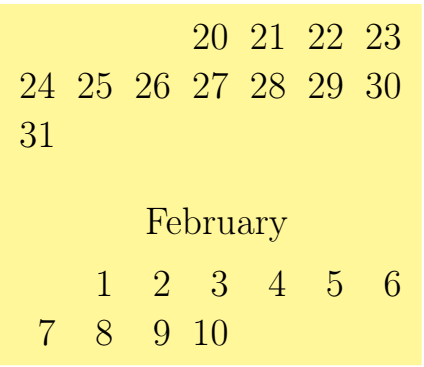
\tikz \calendar[dates=2022-01-20 to 2022-02-10, week list,
month label above left];

/tikz/ month label above centered

Працює як і раніше описане, тільки мітка центрується вище рядка, який містить перший день.



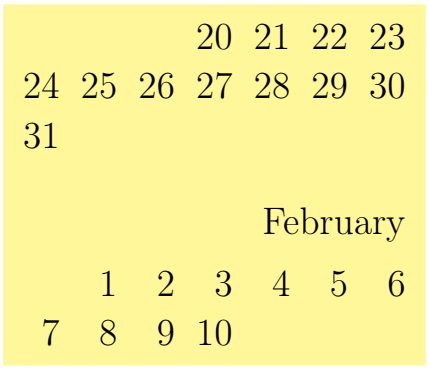
\normalsize \tikz \calendar[dates=2022-02-01 to 2022-02-last, day list right,
month label above centered];



\tikz \calendar[dates=2022-01-20 to 2022-02-10, week list,
month label above centered];

/tikz/ month label above right

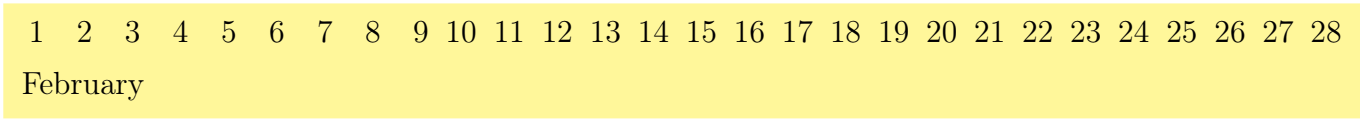
Працює як і раніше описане, але вирівнюється по правому берегу.




```
\tikz \calendar[dates=2022-01-20 to 2022-02-10, week list,
month label above right];
```

```
/tikz/ month label below left
```

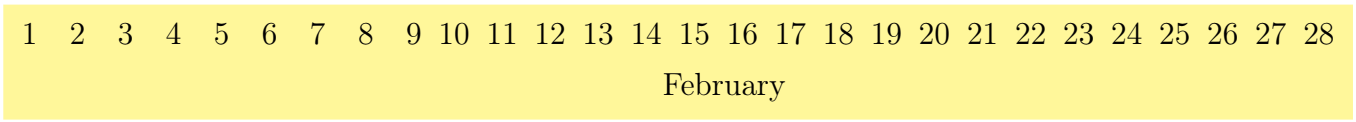
Працює як мітка місяця **label above left**, тільки мітка розташовується нижче рядка. Таке розташування дійсно не використовують зі списком тижнів, але використовують зі списком днів праворуч або розташуванням списку місяців.




```
\normalsize \tikz \calendar[dates=2022-01-20 to 2022-02-10, day list right,
month label below left];
```

```
/tikz/ month label below centered
```

Працює як мітка місяця **label above centered**, тільки нижче рядка.




```
\normalsize \tikz \calendar[dates=2022-01-20 to 2022-02-10, day list right,
month label below centered];
```

10.2.7 Приклади

Бібліотека **calendar** дозволяє складати різноманітні форми календарів (на рік, місяць, тиждень), а також розклади, графіки роботы тощо.

За замовчанням для складання календаря застосовується англійська мова. Для відображення української мови в преамбулі слід визначити необхідні назви. Назви місяців визначають за допомогою команди

```
\def\pgfcalendarmonthname#1{\translate{\ifcase#1\or Січень\or
Лютий\or Березень\or Квітень\or Травень\or Червень\or Липень\or
Серпень\or Вересень\or Жовтень\or Листопад\or Грудень\fi}}
```

Доцільно також визначити особливі дні: свята, вихідні, дні народження тощо.

```
\def\holiday{if (Sunday,
 Saturday,
         equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-02,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=10-14,
         equals=08-24,
         equals=12-25,
         equals=12-26) [red]
% Державні свята
     if (equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=08-24,
         equals=12-25,
% Релігійні свята
         equals=04-24,
         equals=04-24+49,
% Понеділки після свят
 equals=04-25,
 equals=04-25+49,
 equals=05-02,
 equals=12-26) [red]}
% Дні народження
\def\birthday{if (equals=02-21,
 equals=03-09,
 equals=03-11,
 equals=03-18,
 equals=04-12,
 equals=04-22,
 equals=09-08,
```

```
 equals=10-06) {\fill[green] (0,0) circle (10pt);}
    }
```

Розглянемо кілька прикладів календарів, які на думку автора є корисними або гарно виглядають.

10.2.8 Приклад №1

Створюємо календар найпростішої форми у вигляді таблиці з використанням бібліотеки **usetikzlibrary{calendar}**

```
\def\year{2020} %Визначаємо рік
%Записуємо назви місяців
\def\pgfcalendarmonthname#1{%
     \translate{\ifcase#1\or Січень\or Лютий\or Березень\or
 Квітень\or Травень\or Червень\or Липень\or Серпень\or
 Вересень\or Жовтень\or Листопад\or Грудень\fi}%
}

%% Визначаємо святкові, вихідні дні та дні народження
\def\holiday{if (Sunday, Saturday,
         equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-02,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=10-14,
         equals=08-24,
         equals=12-25,
         equals=12-26) [red]
% Державні свята
     if (equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=08-24,
```

```
         equals=12-25,
% Релігійні свята
         equals=04-24,
         equals=04-24+49,
% Понеділки після свят
 equals=04-25,
 equals=04-25+49,
 equals=05-02,
 equals=12-26) [red]}
% Дні народження
\def\birthday{if (equals=02-21,
 equals=03-09,
 equals=03-11,
 equals=03-18,
 equals=04-12,
 equals=04-22,
 equals=09-08,
 equals=10-06) {\fill[green] (0,0) circle (10pt);}
    }
\colorlet{iro}{green!30}
\begin{document}
\ThisCenterWallPaper{1}{tiger.png}
\begin {center}
\includegraphics[scale=0.25]{tiger2.jpg} \hspace{1cm}
{\bf \textit{\color{red}{\Huge \year}} } \hspace{1cm}
\includegraphics[scale=0.2]{tiger.jpg} \\[1cm]

\rmfamily \large
\begin {tabular}{|c|c|c|c|} \hline
% Січень
\tikz[every day/.style={anchor=mid}]
\calendar
[dates=\year-01-01 to \year-01-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
&
% Лютий
\tikz [every day/.style={anchor=mid}]
```

```
\calendar
[dates=\year-02-01 to \year-02-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
 &
% Березень
\tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-03-01 to \year-03-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
\\ \hline

% Квітень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-04-01 to \year-04-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
 &
% Травень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-05-01 to \year-05-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
&
% Червень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-06-01 to \year-06-last,
week list,
```

```
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
\\ \hline
 % Липень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-07-01 to \year-07-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
&
  % Серпень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-08-01 to \year-08-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
&
 % Вересень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-09-01 to \year-09-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday \birthday;
\\ \hline
  % Жовтень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-10-01 to \year-10-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday\birthday;
&
```

```
  % Листопад
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-11-01 to \year-11-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt}]
\holiday\birthday;
 &
  % Грудень
 \tikz [every day/.style={anchor=mid}]
\calendar
[dates=\year-12-01 to \year-12-last,
week list,
month label above centered,
month text=\bfseries\textcolor{blue}{\%mt} ]
\holiday\birthday;
 \\ \hline

\end {tabular}
\end {center}


\rmfamily \large

\colorbox{iro}{Дні Народження}:  21.2 --- Олександр,
9.3 --- Ксенія, 11.3 --- Людмила, \\
\hspace*{4.7cm}
12.4 --- Вікторія,  22.4 --- Сергій,   \\ \hspace*{4.7cm}
8.9 --- Тетяна, 6.10 --- Сергій
%19.8 --- Євген,

\newpage
\ThisCenterWallPaper{1}{tiger.png}
\centerline {\huge  \color{red}{Свята}}
\LARGE
\begin{tabular}{c p{15cm}}
1.1&Новий Рік \\
7.1&Різдво юдо-християнське\\
%8.1&Понеділок після  юдо-християнського Різдва \\
```

```
8.3&Міжнародний жіночий день \\
%9.3&Понеділок після Міжнародного жіночого дня \\
24.4&Великдень \\
25.4&Понеділок після Великодня \\
1.5& День солідарності трудящих \\
9.5& День Перемоги \\
12.6&Трійця \\
13.6& Понеділок після Трійці \\
28.6&День Конституції України \\
%29.6&Понеділок після Дня Конституції України \\
28.7&День Української Державності\\
24.8&День Незалежності \\
14.10&День захисників і захисниць вітчизни \\
%15.10&Понеділок після  Дня Захисника Вітчизни \\
25.12&Різдво \\
\end{tabular}
\\[2cm]

\centerline {\huge  \color{magenta}Нові Роки}
\LARGE
\begin{tabular}{c p{10cm}}
1.2& Китайський Новий Рік \\
2.3& Православний (український) Новий Рік 7530 \\
30.7& Мусульманський Новий Рік 1444\\
25 --- 27.9&Єврейський Новий Рік 5783\\

\end{tabular}


\end{document}
```

Результат компіляції

2022

January

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

February

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

March

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

April

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

May

						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

June

		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

July

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

August

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

September

			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

October

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

November

	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

December

			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Дні Народження: 21.2 — Олександр, 9.3 — Ксенія, 11.3 — Людмила,
12.4 — Вікторія, 22.4 — Сергій,
8.9 — Тетяна, 6.10 — Сергій

Свята

1.1	Новий Рік
7.1	Різдво юдо-християнське
8.3	Міжнародний жіночий день
24.4	Великдень
25.4	Понеділок після Великодня
1.5	День солідарності трудящих
9.5	День Перемоги
12.6	Трійця
13.6	Понеділок після Трійці
28.6	День Конституції України
28.7	День Української Державності
24.8	День Незалежності
14.10	День захисників і захисниць вітчизни
25.12	Різдво

Нові Роки

1.2	Китайський Новий Рік
2.3	Православний (український) Новий Рік 7530
30.7	Мусульманський Новий Рік 1444
25 — 27.9	Єврейський Новий Рік 5783

10.2.9 Приклад №2

Створюємо календар, що наведено на сайті www.texample.net/tikz/examples/all. Цей календар побудовано з використанням команди **matrix**.

```
\documentclass[tikz,border=5]{standalone}

\usepackage[cp1251]{inputenc}
\usepackage[ukrainian]{babel}

\renewcommand\familydefault\sfdefault
\usepackage{tikz}
\usetikzlibrary{positioning,calendar}

\colorlet{darkgreen}{black}
\colorlet{holiday}{red}
\newcommand{\calrow}[1]{\node[anchor=base east](Mon){П};
\node[base right=of Mon](Tue){В}; \node[base right=of Tue](Wed){С};
\node[base right=of Wed](Thu){Ч}; \node[base right=of Thu](Fri){П};
\node[color=red,base right=of Fri](Sat){С}; \node[color=red,base
right=of Sat](Sun){Н};
\node[above=of Thu]{\textbf{#1}};}

\newcommand{\calperiod}[2][\currentyear]{%
  \calendar[dates=\currentyear-#2-01 to \currentyear-#2-last]
    if (Saturday,Sunday) [holiday]
\holidays %щоб позначати свята
\birthdays %щоб позначати Дні Народження
;}
\edef\currentyear{\the\year}
\newcommand{\holidays}{% свята в Україні
if (equals=01-01, % Новий Рік
equals=01-07, % юдо-християнське Різдво
equals=03-08, % Міжнародний Жіночий день
equals=05-01, % День солідарності трудящих
equals=05-09, % День Перемоги
equals=06-28, % День Конституції
equals=07-28, % День Української Державності
equals=08-24, % День Незалежності
equals=10-14, % День захисників і захисниць вітчизни]
equals=12-25, % Різдво
% Релігійні свята
```

```
    equals=04-24, % Великдень
equals=04-24+49, % Трійця
% Понеділки після свят
equals=04-24+1, % Понеділок після Великодня
equals=04-24+50, % Понеділок після Трійці
equals=05-02, % Понеділок після Дня солідарності трудящих
equals=12-26) [holiday] % Понеділок після Різдва
}

\newcommand{\birthdays}{% Дні Народження
if  (equals=02-21) {\fill[green] (0,0) circle (6pt);} % Олександр
if  (equals=03-09) {\fill[green] (0,0) circle (6pt);} % Оксана
if  (equals=03-11) {\fill[green] (0,0) circle (6pt);} % Люда
if  (equals=04-12) {\fill[green] (0,0) circle (6pt);} % Віта
if  (equals=04-22) {\fill[green] (0,0) circle (6pt);} % Сергій
if  (equals=09-08) {\fill[green] (0,0) circle (6pt);} % Тетяна
if  (equals=10-06) {\fill[green] (0,0) circle (6pt);} % Сергій
}

\begin{document}
\begin{tikzpicture}[every calendar/.style={week list},
year label/.style={
  fill=white,text=darkgreen,font=\bfseries\Large
}, current year/.store in=\currentyear,
current year=2022,
every day/.style={anchor=mid}]
\matrix[%
row 1/.style={darkgreen,node distance=.3ex},%
row 3/.style={darkgreen,node distance=.3ex},
row 5/.style={darkgreen,node distance=.3ex},
row 7/.style={darkgreen,node distance=.3ex},
column sep=1ex,%
draw=darkgreen,thick,rounded corners=5pt,%
append after command={
  \pgfextra{\edef\matrixname{\tikzlastnode}}
  node [year label/.try, right=1ex of \matrixname.south west]
  {\currentyear}
  node [year label/.try, right=1ex of \matrixname.north west]
  {\currentyear}
  node [year label/.try, left=1ex of \matrixname.south east]
```

```
  {\currentyear}
  node [year label/.try, left=1ex of \matrixname.north east]
  {\currentyear}
}
]{%

% first row: week day and month
 \calrow{Січень} & \calrow{Лютий} &  \calrow{Березень} \\
\calperiod{01} & \calperiod{02} & \calperiod{03} \\[1ex]

% second row: calendar
 \calrow{Квітень} &  \calrow{Травень} & \hspace{3pt} \calrow{Червень}
 \\
\calperiod{04} & \calperiod{05} & \calperiod{06} \\[1ex]

% third row: week day and month
 \calrow{Липень} & \calrow{Серпень} & \calrow{Вересень} \\
\calperiod{07} & \calperiod{08} & \calperiod{09} \\[1ex]

% forth row: calendar
 \calrow{Жовтень} & \calrow{Листопад} & \calrow{Грудень} \\
\calperiod{10} & \calperiod{11} & \calperiod{12} \\[1ex]\\
};

\end{tikzpicture}
\end{document}
```

Результат компіляції

2022 2022

Січень

П	В	С	Ч	П	С	Н
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Лютий

П	В	С	Ч	П	С	Н
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

Березень

П	В	С	Ч	П	С	Н
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Квітень

П	В	С	Ч	П	С	Н
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Травень

П	В	С	Ч	П	С	Н
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Червень

П	В	С	Ч	П	С	Н
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Липень

П	В	С	Ч	П	С	Н
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Серпень

П	В	С	Ч	П	С	Н
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Вересень

П	В	С	Ч	П	С	Н
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Жовтень

П	В	С	Ч	П	С	Н
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Листопад

П	В	С	Ч	П	С	Н
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Грудень

П	В	С	Ч	П	С	Н
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

2022 2022

10.2.10 Приклад №3

Наведемо календар у вигляді кола з [13].

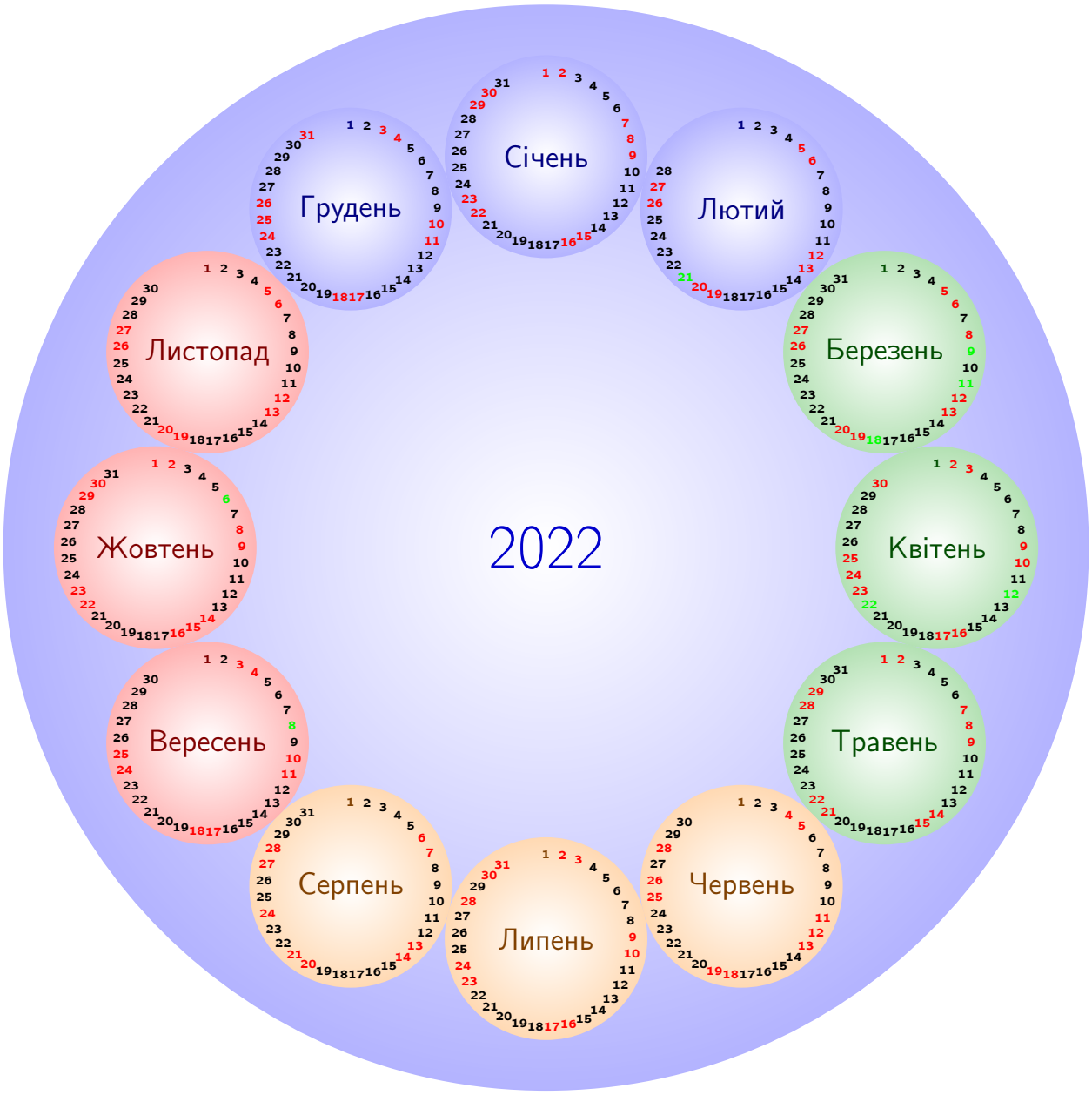
```
\documentclass{article}
\usepackage[cp1251]{inputenc}
\usepackage[ukrainian]{babel}
\usepackage{tikz}
%%%<
\usepackage{verbatim}
\usepackage[active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\setlength\PreviewBorder{5pt}%
%%%>
\begin{comment}
:Title: A calendar of circles
:Tags: Foreach
:Author: Till Tantau
:Slug: calendar-circles

A calendar example from the PGF manual. Modifications are
  - changes in font family and size commands
  - shaded background circle
  - shaded month circles
  - font sizes and distances.
\end{comment}
\usetikzlibrary{calendar,shadings}
\renewcommand*{\familydefault}{\sfdefault}
\colorlet{winter}{blue}
\colorlet{spring}{green!60!black}
\colorlet{summer}{orange}
\colorlet{fall}{red}
% A counter, since TikZ is not clever enough (yet) to handle
% arbitrary angle systems.
\newcount\mycount
% ============================================================
% =============== Change start ===============================
% ============================================================
\year=2022
\def\pgfcalendarmonthname#1{%
     \translate{\ifcase#1\or Січень\or Лютий\or Березень\or
 Квітень\or Травень\or Червень\or Липень\or Серпень\or
```

```
 Вересень\or Жовтень\or Листопад\or Грудень\fi}%
}

%% Визначаємо святкові, вихідні дні та дні народження
\def\holiday{if (Sunday,
 Saturday,
         equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-02,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=10-14,
         equals=08-24,
         equals=12-25,
         equals=12-26) [red]
% Державні свята
     if (equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=08-24,
         equals=12-25,
% Релігійні свята
         equals=04-24,
         equals=04-24+49,
% Понеділки після свят
 equals=04-25,
 equals=04-25+49,
 equals=05-02,
 equals=12-26) [red]}
% Дні народження
\def\birthday{if (equals=02-21,
 equals=03-09,
 equals=03-11,
```

```
 equals=03-18,
 equals=04-12,
 equals=04-22,
 equals=09-08,
 equals=10-06) [green]}
% ================================================================
% ================ Change end ====================================
% ================================================================

\begin{document}
\begin{tikzpicture}[transform shape,
  every day/.style={anchor=mid,font=\tiny}]
  \node[circle,shading=radial,outer color=blue!30,inner color=white,
    minimum width=15cm] {\textcolor{blue!80!black}{\Huge\the\year}};
  \foreach \month/\monthcolor in
    {1/winter,2/winter,3/spring,4/spring,5/spring,6/summer,
    7/summer,8/summer,9/fall,10/fall,11/fall,12/winter} {
      % Computer angle:
      \mycount=\month
      \advance\mycount by -1
      % ================================================================
      % ================ Change start ==================================
      % ================================================================
      \multiply\mycount by -30
      \advance\mycount by 90
      % ================================================================
      % ================ Change end ====================================
      % ================================================================
      \shadedraw[shading=radial,outer color=\monthcolor!30,middle
  color=white, inner color=white,draw=none]
  (\the\mycount:5.4cm) circle(1.4cm);
      % The actual calendar
      \calendar at (\the\mycount:5.4cm) [
          dates=\the\year-\month-01 to \the\year-\month-last]
      if (day of month=1) {\large\color{\monthcolor!50!black}
  \tikzmonthcode}
      if (Saturday,Sunday) [red]
  \holiday\birthday
      if (all) {
      % Again, compute angle
```

```
        \mycount=1
        \advance\mycount by -\pgfcalendarcurrentday
        \multiply\mycount by 11
        \advance\mycount by 90
        \pgftransformshift{\pgfpointpolar{\mycount}{1.2cm}}};}
\end{tikzpicture}
\end{document}
```

Результат компіляції



10.2.11 Приклад №4

Наведемо календар у вигляді стрічки з [13].

```
\begin{tikzpicture}[every day/.style={anchor=mid}]
  \small\sffamily
  \colorlet{darkgreen}{green!50!black}
  \calendar[dates=2022-01-01 to 2022-12-31,week list,
            month label left vertical,month yshift=0pt,
            month text=\textcolor{blue}{\%mt}]
           if (Saturday,Sunday) [red]
% Свята
if(equals=2022-01-07, equals=2022-03-08, equals=2022-05-09,
equals=2022-06-28,equals=2022-07-28,equals=2022-06-24,
equals=2022-10-14) [red]
% Понеділки після свят
if(equals=2022-04-25, equals=2022-05-02,equals=2022-06-13,
equals=2022-12-26) [red]
% Дні народження
if(equals=2022-02-21, equals=2022-03-09,equals=2022-03-11,
equals=2022-04-12,equals=2022-04-12,equals=2022-04-22,
equals=2022-09-08,equals=2022-10-06)
{\fill[green] (0,0) circle (6pt);};
\end{tikzpicture}
```

Результат компіляції

January						1	2
	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
February	31	1	2	3	4	5	6
	7	8	9	10	11	12	13
	14	15	16	17	18	19	20
	21	22	23	24	25	26	27
March	28	1	2	3	4	5	6
	7	8	9	10	11	12	13
	14	15	16	17	18	19	20
	21	22	23	24	25	26	27
April	28	29	30	31	1	2	3
	4	5	6	7	8	9	10
	11	12	13	14	15	16	17
	18	19	20	21	22	23	24
May	25	26	27	28	29	30	1
	2	3	4	5	6	7	8
	9	10	11	12	13	14	15
	16	17	18	19	20	21	22
June	23	24	25	26	27	28	29
	30	31	1	2	3	4	5
	6	7	8	9	10	11	12
	13	14	15	16	17	18	19
July	20	21	22	23	24	25	26
	27	28	29	30	1	2	3
	4	5	6	7	8	9	10
	11	12	13	14	15	16	17
August	18	19	20	21	22	23	24
	25	26	27	28	29	30	31
	1	2	3	4	5	6	7
	8	9	10	11	12	13	14
September	15	16	17	18	19	20	21
	22	23	24	25	26	27	28
	29	30	31	1	2	3	4
	5	6	7	8	9	10	11
October	12	13	14	15	16	17	18
	19	20	21	22	23	24	25
	26	27	28	29	30	1	2
	3	4	5	6	7	8	9
November	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
	31	1	2	3	4	5	6
December	7	8	9	10	11	12	13
	14	15	16	17	18	19	20
	21	22	23	24	25	26	27
	28	29	30	1	2	3	4
	5	6	7	8	9	10	11
	12	13	14	15	16	17	18
	19	20	21	22	23	24	25
	26	27	28	29	30	31	

10.2.12 Приклад №5

Наведемо місячний календар з сайту www.texample.net/tikz/examples/all.

```
% Birthday calendar
% Author: Hakon Malmedal
\documentclass[fontsize=20pt]{scrartcl}
\usepackage[ukrainian]{babel}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage[margin=1cm,a4paper,landscape]{geometry}
\pagestyle{empty}
\usepackage{graphicx}
\usepackage{tikz}
\usetikzlibrary{calendar,fit}
\usepackage{expl3,xparse}
\usepackage{verbatim}

%% Adapted from http://tex.stackexchange.com/a/10199/4771
%% Визначаємо назву днів тижня
\makeatletter%
\tikzoption{day headings}{\tikzstyle{day heading}=[#1]}
\tikzstyle{day heading}=[]
\tikzstyle{day letter headings}=[
       execute before day scope={ \ifdate{day of month=1}{%
         \pgfmathsetlength{\pgf@ya}{\tikz@lib@cal@yshift}%
         \pgfmathsetlength\pgf@xa{\tikz@lib@cal@xshift}%
         \pgftransformyshift{-\pgf@ya}
         \foreach \d/\l in {0/Понеділок,1/Вівторок,2/Середа,
 3/Четвер,4/П'ятниця,5/{\color{red}Субота},
 6/{\color{red}Неділя}} {
           \pgf@xa=\d\pgf@xa%
           \pgftransformxshift{\pgf@xa-\cellwidth/2}%
           \pgftransformyshift{\pgf@ya}%
           \node[above=-0.5ex,day heading]{\l};%
         }
       }{}%
     }%
]
\makeatother%
%% End
```

```
\def\year{2022} %Визначаємо рік
%% Adapted from pgf source
%% Визначаємо назву місяців
\def\pgfcalendarmonthname#1{%
     \translate{\ifcase#1\or Січень\or Лютий\or Березень\or
 Квітень\or Травень\or Червень\or Липень\or Серпень\or
 Вересень\or Жовтень\or Листопад\or Грудень\fi}%
}
%% Визначаємо святкові, вихідні дні та дні народження
\def\zzz{if (Sunday, Saturday,
         equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-02,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=10-14,
         equals=08-24,
         equals=12-25,
         equals=12-26) [red]
% Державні свята
     if (equals=01-01,
         equals=01-07,
         equals=03-08,
         equals=05-01,
         equals=05-09,
         equals=06-28,
         equals=07-28,
         equals=08-24,
         equals=12-25,
% Релігійні свята
         equals=04-24,
         equals=04-24+49,
% Понеділки після свят
 equals=04-25,
 equals=04-25+49,
 equals=05-02,
 equals=12-26) [flag-flying day, red]
```

```
     if (equals=01-01) [observance=Новий Рік]
     if (equals=01-07) [observance=Юдо-християнське Різдво]
     if (equals=03-08) [observance=Міжнародний жіночий день]
     if (equals=05-01) [observance=День солідарності трудящих]
     if (equals=05-02) [observance=Понеділок після Дня солідарності
 трудящих]
     if (equals=05-09) [observance=День Перемоги]
     if (equals=06-28) [observance=День Конституції]
     if (equals=07-28) [observance=День Української Державності]
     if (equals=08-24) [observance=День  Незалежності]
     if (equals=10-14) [observance=День захисників і захисниць
 вітчизни]
     if (equals=12-25) [observance=Різдво]
 if (equals=12-26) [observance=Понеділок після Різдва]
     if (equals=04-24) [observance=Великдень]
     if (equals=04-24+1) [observance=Понеділок після Великодня]
     if (equals=04-24+49) [observance=Трійця]
     if (equals=04-24+50) [observance=Понеділок після Трійці]
% Дні народження
 if (equals=02-21) [anniversary={Олександ}{1954},cyan]
     if (equals=03-09) [anniversary={Оксана}{1981},cyan]
     if (equals=03-11) [anniversary={Люда}{1972},cyan]
     if (equals=03-18) [anniversary={Тома}{1956},cyan]
     if (equals=04-12) [anniversary={Віта}{1994},cyan]
     if (equals=04-22) [anniversary={Сергій}{2001},cyan]
     if (equals=09-08) [anniversary={Таня}{1983},cyan]
     if (equals=10-06) [anniversary={Сергій}{1991},cyan]
    ;}

\makeatletter
\tikzstyle{month label above centered}=[%
     execute before day scope={%
       \ifdate{day of month=1}{%
         {
           \pgfmathsetlength{\pgf@xa}{\tikz@lib@cal@xshift}%
           \pgf@xb=\tikz@lib@cal@width\pgf@xa%
           \advance\pgf@xb by-\pgf@xa%
           \pgf@xb=.5\pgf@xb%
           \pgftransformxshift{\pgf@xb}%
           \pgftransformxshift{-\cellwidth/2}%
```

```
            \pgfmathsetlength{\pgf@y}{\tikz@lib@cal@yshift}%
            \pgftransformyshift{0.333\pgf@y}
            \tikzmonthcode%
          }
        }{}},
      every month/.append style={anchor=base,blue}
]
\makeatother
%% End

\ExplSyntaxOn

%% Adapted from http://tex.stackexchange.com/a/56214/4771
%%
% first of all we define the user level commands
\NewDocumentCommand{\addtext}{ m }{ \bdaycal_input_add:n { #1 } }
\NewDocumentCommand{\addtextyear}{ mm }{ \bdaycal_input_add_y:nn
{ #1 } { #2 } }
\NewDocumentCommand{\showtext}{ }{ \bdaycal_output_direct: }

% allocate variable:
% a sequence for global storage of the inputs;
\seq_new:N \g_bdaycal_input_seq

% store globally an input in the sequence
\cs_new:Npn \bdaycal_input_add:n #1
    {
     \seq_gput_right:Nn \g_bdaycal_input_seq { #1 }
    }

\cs_new:Npn \bdaycal_input_add_y:nn #1 #2
    {
     \seq_gput_right:Nn \g_bdaycal_input_seq { #1 ~
 ( \int_to_arabic:n
 { \pgfcalendarifdateyear - #2 } ) }
    }

% how to output in direct order; we simply do a mapping function
calling
% \bdaycal_print:n after incrementing the counter
```

```
\cs_new_protected:Npn \bdaycal_output_direct:
    {
     \seq_map_inline:Nn \g_bdaycal_input_seq
      {
       \bdaycal_print:n { ##1 }
      }
     \seq_gclear:N \g_bdaycal_input_seq
    }

% the printing macro; change here for adapting to your wishes
\cs_new:Npn \bdaycal_print:n #1
    {
     #1 \par
    }
%% End

% Прапор
\newsavebox{\flagNO}
\savebox{\flagNO}{
\begin{tikzpicture}
    % \fill[roed] rectangle (6pt,6pt);
    % \fill[roed,yshift=10pt] rectangle (6pt,6pt);
    % \fill[blue,xshift=10pt] rectangle (12pt,6pt);
    % \fill[roed,xshift=10pt,yshift=10pt] rectangle (12pt,6pt);
    % \fill[blaa,yshift=7pt] rectangle (22pt,2pt);
    % \fill[blaa,xshift=7pt] rectangle (2pt,16pt);
  \fill[yellow] rectangle (22pt,8pt);
  \fill[blue,yshift=8pt] rectangle (22pt,8pt);
\end{tikzpicture}
}

\pgfkeys{/tikz/week~number/.code =
     {
       \week_number:nnn {
         \pgfcalendarifdateyear } {
         \pgfcalendarifdatemonth } {
         \pgfcalendarifdateday }
       \addtext{\int_to_arabic:n { \l_week_number_W_int } }
     }
}
```

```
\ExplSyntaxOff

\pgfkeys{/tikz/flag-flying day/.code =
     {
       \draw (-\cellwidth,0) node [above right,font=\Huge]
       {\resizebox{!}{0.8ex}{\usebox{\flagNO}}};
     }
}

\pgfkeys{/tikz/observance/.code =
     {
       \addtext{#1}
     }
}

\pgfkeys{/tikz/anniversary/.code 2 args=\addtextyear{#1}{#2}}

\pgfkeys{/tikz/day code =
     {
       \node (lower right) at (0,0) [above left,font=\Huge]
   {\tikzdaytext};
       \node (upper left) at (-\cellwidth,\cellheight)
       [below right,align=left,text width=\cellwidth-\pgflinewidth,
       font=\tiny,black] {\showtext};
       \node (lower left) at (-\cellwidth,0) {};
       \node[rounded corners, draw,
             fit=(lower right) (upper left) (lower left),
             inner sep=1mm] {};
     }
}

\pgfkeys{/tikz/inner sep = 0pt}

\pgfkeys{/tikz/day xshift=\cellwidth+2mm+2mm}

\pgfkeys{/tikz/day yshift=\cellheight+2mm+2mm}

\newlength{\cellheight}
\setlength{\cellheight}{25mm}
```

```
\newlength{\cellwidth}
\setlength{\cellwidth}{35mm}

\begin{document}

\centering

\begin{tikzpicture}[thick]
     \calendar[dates=\year-04-01 to \year-04-last,
               week list,
               month label above centered,
               month text=\textbf{\%mt \%y0},
               day headings={font=\footnotesize},
               day letter headings]
     \zzz
\end{tikzpicture}

\end{document}
```

Результат компіляції

Квітень 2022

Понеділок	Вівторок	Середа	Четвер	П’ятниця	Субота	Неділя
				1	2	3
4	5	6	7	8	9	10
11	Віта (28) 12	13	14	15	16	17
18	19	20	21	Сергій (21) 22	23	Великдень  24
Понеділок після Великодня  25	26	27	28	29	30	

11 Робота над помилками

Вихідний текст документу може знаходитися у кількох файлах, які викликають з кореневого файлу командами **input** чи **include**. Під час обробки документу TEX інформує користувача про те, який файл він обробляє. Під час відкриття нового файлу на екран виводиться відкриваюча кругла дужка, за якою йде повне ім'я файлу, далі видаються номери сторінок, що сформовано TEX'ом з інформації, яка міститься в цьому файлі. Нарешті, після закінчення обробки файлу на екран виводиться закриваюча дужка. Дужки, які оточують ім'я кореневого файлу, міститимуть вкладені дужки з повними іменами всіх файлів, що використовують, і списками сторінок, що ними створено. Таким чином, користувач має можливість відслідковувати, який саме файл обробляють у певний момент [8].

Через те, що LATEX 2_ε запускається з командного рядка консолі, то й повідомлення виводяться на консоль. Найчастіше ці повідомлення містять попередження й повідомлення про помилки. Якщо LATEX 2_ε запущено з програми-оболонки типу **texworks**, результат відображається в окремому вікні. Попередження не переривають роботу компілятора, і на сучасних комп'ютерах прочитати їх важко через високу швидкість зміни повідомлень. Крім консолі, повідомлення записуються у LOG-файл, і після закінчення роботи компілятора їх можна там переглянути. Зустрівши помилку, компілятор призупиняє роботу й виводить на консоль чи в окреме вікно оболонки повідомлення, яке має вигляд:

```
! LaTeX Error: Environment tbular undefined.

See the LaTeX manual or LaTeX Companion for explanation.
Type H <return> for immediate help.
...

l.103 \begin{tbular}
                    {lrccc}
?
```

Це повідомлення про помилку та її короткий опис, номер рядка вхідного TEX-документа (у прикладі вказано рядок 103 вхідного документу), в якому знайдено помилку, й запит оператору про дії у вигляді знаку запитання (?). У цьому ж повідомленні виводиться помилковий рядок документу зі вставленим символом «перевод рядка» у тому місці, яке компілятор не зміг опрацювати. Після виводу повідомлення про помилку компілятор очікує дії оператора [5].

Помилки можуть бути виправлені як під час компіляції, так і у вихідному документі. Оскільки перша помилка може призводити до появи наступних, то процес правки повинен здійснюватися поступово, починаючи від першої. По-

милки на етапі компіляції можна і проігнорувати, але при цьому документ на екрані буде відображено невірно.

Після завершення трансляції документу (якщо помилок не виявлено) вікно буде закрите. Під час виявлення помилок трансляція документу призупиняється [2].

LaTeX 2_ε має досить розвинені можливості опрацювання помилок. Якщо помилка є простою, то можна вказати компілятору ігнорувати її, натиснувши клавішу **Enter**, після чого робота компілятора буде продовжуватися. Якщо помилка є серйозною, процес переривають, натиснувши комбінацію клавіш **Ctr-C** чи **Ctr-Break** або кнопку у програмі-оболонці. Крім того, перервати роботу компілятора після повідомлення про помилку можна, якщо ввести у будь-якому регістрі літеру **x** й натиснути **Enter**. Виправивши помилку за відомим номером рядка у TeX-документі та описом помилки, компілятор запускають знову. Здебільшого випадків під час роботи в програмі-оболонці цього достатньо для відлагоджування та виправлення TeX-документу.

В процесі роботи LaTeX 2_ε записує проміжні результати у AUX- та OUT-файли, які використовують під час підготовки документу. Крім того, проміжні результати можуть записуватися й в інші допоміжні файли. Часто під час виникнення помилки у ці файли заносяться хибні проміжні результати. Це призводять до появи дивних, на перший погляд, повідомлень про помилки. Краще за все вилучити всі допоміжні файли після кожної помилки чи, як мінімум, прибрати AUX- та OUT-файли.

Літера **s** переводить LaTeX 2_ε в режим **Scroll**, за якого зупинка роботи під час винекнення помилки не відбувається. Повідомлення про помилки виводяться на консоль і у LOG-файл, але робота продовжується. Компілятор зупиниться, коли не вдасться знайти частину TeX-документу.

Літера **r** (режим **Run**) змусить компілятор працювати без зупинки за всіх помилкових ситуаціях.

Літера **q** (**Run Quietly**) переводить компілятор у пакетний режим. У цей же режим його можна перевести командою **batchmode**, яку розташовують на початку TeX-документу. У цьому режимі компілятор працює без зупинки в усіх помилкових ситуаціях й нічого не виводить на консоль, продовжуючи виводити повідомлення в LOG-файл.

Літера **i** (режим **Insert**) дозволяє ввести з клавіатури символ, якого не вистачає, чи слово й продовжити роботу. Введена у відповідь на запит цифра змусить компілятор пропустити відповідну кількість символів й продовжити роботу.

Набравши літеру **h** (**Help**), можна отримати розширений опис помилки та рекомендації з її усунення. У нашому прикладі це виглядатиме так:

```
Your command was ignored.
```

```
Type I <command> <return> to replace it with another command,
or <return> to continue without it.
```

У відповідь на введений знак запитання LaTeX 2$_\varepsilon$ виведе коротку підказку про режими роботи.

Іноді замість знаку запитання оператору видається зірочка. Найчастіше це означає, що програма не знайшла команду завершення оточення **document** (**end{document}**). Для завершення роботи можна ввести команду **stop** і натиснути клавішу **Enter**.

На відміну від помилок, попередження не викликають зупинок роботи. Всі попередження також супроводжуються повідомленнями із зазначенням номеру рядка вхідного TeX-документу. Більша частина попереджень пов'язана з незаповненими чи переповненими боксами. Такі повідомлення здебільшого означають, що LaTeX 2$_\varepsilon$ не зміг підібрати правильний розмір боксу.

```
Overfull \hbox (34.91pt too wide) in paragraph at lines 26--33
...
```

Переповнені бокси зручно переглядати в режимі **draft** для стандартних класів LaTeX 2$_\varepsilon$ [5].

Помилки, що найчастіше зустрічаються й розпізнаються TeX'ом і LaTeX'ом:

- невідповідність відкриваючих та закриваючих групуючих дужок чи дужок, які виділяють аргумент команди;
- пропущено один з роздільників математичної моди;
- команду використано у неприпустимій для неї моді;
- пропущено команду початку чи кінця документу.

11.1 Повідомленя про помилки, які видає LaTeX

Нижче наведено список повідомлень, які видає LaTeX під час знаходження помилок. Повідомлення наведено у абетковому (за латиницею) порядку.

- **Bad \line or \vector argument**
  Перший аргумент команд **line** чи **vector**, що використовують всередині командних дужок **picture**, має неправильні компоненти.
- **Bad math environment delimiter**
  Пропущено один з розділювачів, що обрамляють вирази у математичній моді: $, $$, \[, \], \(чи \).
- **Bad use of **
  Команда \\використовується між абзацами, де вона не має сенсу.

- **\begin{...} ended by \end{...}**
  LaTeX знайшов тільки кінець чи тільки початок командних дужок. Можлива причина — друкарська помилка в імені, а також пропуск чи зайва вставка **\begin** чи **\end** командних дужок.

- **Can be used only in preamble**
  Команду LaTeX'у, яка може з'являтися тільки в преамбулі документу, використано після команди **\begin{document}**. До команд преамбули відносяться: **\documentclass, \nofile, \includeonly, \makeindex** та **\makeglossary**. Та ж діагностика з'являється за наявності зайвої команди **\begin{document}**.

- **Command name ... already used**
  Спроба використати одну з команд визначення нової команди чи командної дужки **\newlist**, **\newtheorem**, **\newcommand**, чи **\newlength** з іменем, яке вже використали раніше. Якщо ви хочете переозначити вже визначену команду, замість **\newcommand** варто використати **\renewcommand**.

- **Command name name not defined as a math alphabet** Команда **name** не визначена як математична абетка. Повідомлення викликано помилкою в шрифтовому пакеті. Зверніться до автора пакету.

- **Command name not provided in base NFSS**
  Команду **name** не реалізовано у базовій схемі NFSS. **NFSS (New Font Selection Scheme, Нова схема вибору шрифтів)** виключила низку символів, що були в LaTeX 2.09, з формату LaTeX 2_ε. Помилка генерується під час використання однією з команд **\mho \Join \Box \Diamond \leadsto \sqsubset \sqsupset \lhd \unlhd \rhd \unrhd** які тепер визначені в пакеті **latexsym**. Додайте в преамбулу декларацію **\usepackage {latexsym}**.

- **Corrupted NFSS tables**
  Зіпсовано таблиці NFSS. Зіпсовано систему завантаження шрифтів. Наприклад, вилучено якийсь файл визначення шрифтів (з розширенням fd). Перевстановіть LaTeX чи пакет, що викликав помилку.

- **Counter too large**
  LaTeX намагається полічити, відмічаючи пункти літерами, і при цьому кількість пунктів перевищує 26 (кількість літер в англійській абетці). Можлива також помилка в роботі з іншими лічильниками.

- **Encoding scheme ENC undefined**
  Схема кодування з іменем ENC невідома. Визначте кодування, вказавши ENC у необов'язкрвому аргументі **\usepackage** під час завантажен-

ня пакету **fontenc**. Завантаженню пакета **fontenc** має передувати завантаження шрифтового пакету, який втклікав помилку. Якщо у вихідному тексті документа використовують шрифтові команди низького рівня типу **fontencoding**, слід перевірити правильність написання ENC.

- **Environment ... undefined**
  Використовуються командні дужки з невідомим іменем.

- **File ended while scanning use of \end**
  Команду не завершено

- **File file not found**
  Файл file не знайдено. LaTeX намагається прочитати файл, якого не існує. Якщо файл має розширенння **tex**, тоді його намагються прочитати команди **input** чи **include**; якщо він має розширенння **cls**, тоді вказано неіснуючий клас друкованого документу в декларації **documentclass**; якщо він має розширенння **sty**, тоді вказано неіснуючий пакет в декларації **usepackage**. LaTeXпризуриняє роботу й очикує вводу з клавиатури правильного імені файлу; під час натискання клавіши **Enter** робота буде продовжена без повторення спроби знайти файл.

- **Float(s) lost**
  Командні дужки плаваючих елементів (**figure, table**) чи команда **marginpar**, що розташовані всередині боксу. Якщо в тексті багато плаваючих елементів, LaTeX можливо, виявить цю помилку, далеко порсунувшись від команди, яка її містить, і знайти відповідне місце в тексті буде нелегко. Один з таких елементів буде загублено, але не обов'язково той, що помилково розташовано в боксі.

- **Font family ENC+family unknown**
  Сім'я шрифтів ENC+family невідома. Виявлено помилку у шрифтовому пакеті. Зверніться до автора пакету.

- **Font name not found**
  Шрифт name незнайдено. Помилка у файлах визначення шрифтів. Зверніться до розробника.

- **Illegal character in array arg**
  В аргументі командних дужок **array** чи **tabular** або у другому аргументі команди **multicolumn**, яку використовують у цих командних дужках, знайдено заборонений символ.

- **\include cannot be nested**
  Команда **include**, що використовують у файлі, який міститься у кореневому вхідному файлі за допомогою іншої команди **include**

- **LaTeX 2ε command command in LaTeX 2.09 document**
  Команда command версії LaTeX 2ε використана в документі LaTeX 2.09. Необхідно замінити її командою, яка існує у версії LaTeX 2.09, або виконати компіляцію документу в режимі LaTeX 2ε.

- **Lonely \item–perhaps a missing list environment**
  Поодиноку команду **\item** можливо пропущено процедура складання списку. Команда **\item** знаходиться поза процедурою складання списку.

- **Math alphabet identifier alphabet is undefined in math version ver**
  Ідентифікатор математичної абетки alphabet не визначено у версії ver. Помилка у шрифтовому пакеті. Зверніться до розробоника.

- **Math version version is not defined**
  Математична версія version не визначена. Помилка у шрифтовому пакеті. Зверніться до розробоника.

- **Misplaced alignment tab character &**
  Зайвий символ табуляції &. Символ & використано у звичайому тексті. Він може використовуватися у процедурах **array** і **tabular**. Можливо, варто ндрукувати **\&**.

- **Missing number, treated as zero**
  Пропущено число, замінено на нуль. TeX, можливо, виконував команду LaTeX'у, яка очікувала як аргумент параметр довжини й не отримала його. Можливо, пропущено аргумент чи квадратна дужка на початку чи в кінці необов'язкового аргументу. Цю помилку також викликає вставка команды **\protect** або перед командною довжиною, чи перед командою **\value**, яка створює число.

- **Missing \begin{document}**
  Пропущено команду **\begin{document}**, або в преамбулі міститься текст чи команди, які формують текст.

- **Missing p-arg in preamble**
  В аргументі командних дужок **array** чи **tabular**, або у другому аргументі команди **\multicolumn**, яку використовують всередині цих командних дужок, знайдено символ **p** без наступного за ним виразу.

- **Missing @-exp in preamble**
  В аргументі командних дужок **array** чи **tabular**, або у другому аргументі команди **\multicolumn**, яку використовують всередині цих командних дужок, знайдено символ **@** без наступного за ним @-виразу.

- **NFSS release 1 command name found**
  Виявлено команду name першого випуску NFSS. Це повідомлення може

бути як попередженням, так і інформацією про помилку. Команда з іменем name використовувалася у першому випуску нової схеми вибору шрифтів (New Font Selection Scheme), але пізніше була замінена іншою. У більшості випадків компілятор здатен підставити замість застарілої команди її більш новий аналог, однак в у будь-якому випадку необхідно внести виправлення у вихідний текст документу чи отримати більш нову версію пакету, який містить команду name.

- **No counter counter defined**
  Лічильник counter не визначено. Використане неіснуюче ім'я лічильника у обов'язковому аргументі команди **setcounter** чи **addtocounter** чи у необов'язковому аргументі декларацій **newcounter** чи **newtheorem**. Однак, якщо цю помилку виявлено під час читання aux-файлу, тоді, можливо, її викликано використанням декларації **newcounter** у файлі, який міститься в документі через команду **include**.

- **No declaration for shape shape**
  Немає декларації для накреслення shape. Виконано підстановку шрифту, для якого не визначено накреслення **shape**.

- **No such counter**
  Визначення неіснуючого лічильника в одній з команд, що працює з лічильниками, наприклад **setcounter** чи **addtocounter**. Якщо помилка з'являється під час обробки файлу з розширенням .aux, це може означати, що визначення лічильника командою **newcounter** знаходиться поза преамбулою (тобто після **begin{document}**).

- **Not in outer par mode**
  Командні дужки плаваючих елементів (**figure, table**) або команда **marginpar** використовуються в математичній моді чи в боксі. Плаваючі елементи можна використовувати тільки у зовнішній абзацній моді.

- **Option clash for package package**
  Конфлікт опцій для пакету **package**. Вказаний пакет завантажено двічі з різними опціями. Натиснувши **h** + **Enter**, можна отримати список опцій під час кожного завантаження. Вихід полягає в тому, щоб завантажити пакет відразу з усіма необхідними опціями. Варто мати на увазі, що пакети можуть завантажувати інші пакети, тому вказану помилку можна отримати без явного виклику двічі одного й того ж пакету.

- **\pushtabs and \poptabs don't match**
  В межах командних дужок **tabbing** виявлено команду **poptabs** без попередньої команди **pushtabs**; або в момент обробки команди **end{tabbing}** виявилося, що одна чи кілька команд **pushtabs** є зайвими.

- **Something's wrong–perhaps a missing \item**
  Швидше за все, пропущено команду **item** в командних дужках, що утворюють список. Таке повідомлення видається, якщо пропущено аргумент в командних дужках **thebibliography**.

- **Suggested extra height (14454.0pt) dangerously large**
  В **enlargethispage** вказане надто велике число

- **Symbol font name is not defined**
  Символьний шрифт name не визначено. Помилка у шрифтовому пакеті. Зверніться до автора.

- **Tab overflow**
  Перевищено границю для кількості позицій табулятора. Збільшення цієї границі можна досягнути тільки модифікацією LaTeX'у.

- **There's no line here to end**
  Команди **newline** або \\ використовуються між абзацами, де вони не мають сенсу. Для отримання додаткових проміжків між абзацами варто використовувати команди **vspace** або **vskip**.

- **Text for \verb command ended by end of line**
  Команду **verb** розпочато, але не завершено у поточному рядку. Слід перевірити, чи не забуто другий символ, який визначає межі аргументу команди **verb**.

- **This is a LaTeX bug**
  Це наслідок помилки в самому LaTeX'і. У такому випадку рекомендується показати документ експерту.

- **This NFSS system isn't set up properly**
  Систему NFSS налаштовано неправильно. В системі завантаження шрифтів виявлено фатальну помилку. Помилка виникає, якщо в кінці логічного ланцюжка підстановки відсутніх шрифтів виявлено також відсутній шрифт. Перевстановіть LaTeX чи зверніться до розробника шрифтового пакету, який викликав помилку.

- **Too deeply nested**
  В документі більше шести вкладених списків або більше чотирьох вкладених списків з автоматичним перерахуванням. (Командні дужки **quote** і **verse** також розглядаються як списки.)

- **Too many }'s**
  Відкрита фігурна дужка не закрита або закрита не за правилами

- **Too many columns in eqnarray environment**
  Забагато колонок у процедурі **eqnarray**. Процедура **eqnarray** містить три роздільника колонок & без проміжної команди \\переходу на новий рядок.

- **Too many unprocessed floats**
  LaTeX'у не вистачило пам'яті для плаваючих елементів, які створюються командними дужками **figure, table** чи командою **marginpar**. Помилка виникає або під час спроби створити забагато таких об'єктів на одній сторінці, або коли в командних дужках **figure** чи **table** задано параметри розташування, які перевищують можливості однієї сторінки.

- **Undefined tab position**
  Спроба пересунення табулятора на невизначену позицію під час використання однієї з наступних команд: \>, \+, \- чи \<.

- **\usepackage before \documentclass**
  Використання пакету перед класом документа

- **Undefined size function name**
  Шрифтову розмірну функцію name не визначено. Помилка у шрифтовому пакеті. Зверніться до автора пакету.

- **Unknown option option for package package**
  Невідома опція option для пакету package. Перевірте необов'язковий аргумент декларації **protectusepackage**.

- **You can't use macro parameter character # in mode mode**
  Ви не можете використовувати символ макропараметру # в моді mode. Спецсимвол # виявлено у звичайному тексті. Можливо, варто надрукувати \#.

- **\< in mid line**
  Команда \< з'явилася в середині рядка в командних дужках **tabular**. Ця команда може знаходитися тільки на початку рядка.

11.2 Повідомлення про помилки, які видає TeX

Під час обробки файлу LaTeX'ом низка помилок діагностується самим процесором TeX. Тут наведено список повідомлень про помилки, які видає TeX.

- **Counter too large**
  Сноски нумерують літерами або спеціальними символами, кількість яких є обмеженою. Ця помилка може виникнути, наприклад, під час використання надто великої кількості команд **thanks**.

- **Double subscript**
  В математичній формулі виявлено неоднозначну комбінацію для отримання подвійного нижнього індексу.

- **Double superscript**
  В математичній формулі виявлено неоднозначну комбінацію для отримання подвійного верхнього індексу.

- **Extra alignment tab has been changed to \cr**
  Кількість символів & в рядку масиву чи таблиці перевищує задану кількість стовбчиків.

- **Extra }, or forgotten $**
  Групуючі дужки не відповідають одна одній, або пропущено (чи задано зайвий) обмежувач виразу в математичній моді.

- **Font ... not loaded. Not enough room left**
  Документ використовує більше шрифтів, ніж дозволяє TEX. Можна спробувати опрацювати документ частинами.

- **I can't find file ...**
  TEXне може зайти вказаний вхідний файл. Це може бути результатом помилки під час вводу імені файлу та запуску TEX'а або під час зазначення неіснуючого файлу в команді **input** або **documentclass**. Якщо файл не знайдено, TEXвидає запрошення:
  Please type another input file name:
  і очікує на ввід відповідного імені файлу.

- **Illegal unit of measure (pt inserted)**
  Під час визначення довжини вказана безрозмірна величина.

- **Illegal parameter number in definition of ...**
  Під час використання команд **newcommand** чи **renewcommand** здійснено посилання на аргумент, номер якого більший за кількість аргументів, що визначено в цій команді. Якщо замість трьох крапок у повідомленні вказано команду **@gtempa**, це може означати, що в аргумент команди **label** записано символ #.

- **Misplaced alignment tab character &**
  Використання спеціального символу & у звичайному тексті, а не в таблиці.

- **Missing control sequence inserted**
  У визначенні нових команд за допомогою **newcommand**, **renewcommand** чи **newlength** існує таке, коли перший аргумент не є правильно заданим іменем команди.

- **Missing number, treated as zero**
  Не задано числове значення параметру (розмірного чи безрозмірного). Ймовірніше, пропущено обов'язковий аргумент команди чи квадратні дужки в тексті розташовано так, що були визнані за початок необов'язкового аргументу.
- **Missing { inserted**
- **Missing } inserted**
  Пропущено одну з групуючих дужок.
- **Missing $ inserted**
  Ряд команд TEX'у можуть використовуватися лише у математичній моді. Зустрічаючи такі команди поза математичною модою, TEXвидає це повідомлення про помилку і при цьому автоматично перемикається у математичну моду.
- **Not a letter**
  Помилка в аргументі команди **hyphenation**.
- **Runaway argument?**
  Кількість дужок та їх розташування не відповідає правилам оформлення в LATEX
- **Paragraph ended before ... was complete**
  Зареєстровано ознаку кінця абзацу (команда **par** або порожній рядок) в аргументі команди, де не може бути кілька абзаців. Причиною помилки може бути також відсутність закриваючої дужки в кінці аргумента чи розділювача в команді **verb**.
- **TeX capacity exceeded, sorry [...]**
  TEX'у не вистачило пам'яті, і він завершив роботу. В квадратних дужках вказують параметр, який визначає можливі причини помилки, а саме:
  - **buffer size** В аргументах команд рубрикації чи команд **caption,** **addcontentsline** або **addtocontents** задано надто довгий фрагмент тексту. Повідомлення може видаватися під час роботи команди **end{document}**, якщо подібна помилка мала міце під час виконання команд **tableofcontents,** **listoffigures** чи **listoftables**.
  - **exception dictionary** TEX'у задано забагато винятків для переносу слів командами **hyphenation**. Вилучіть з цих команд слова, що рідко зустрічаються у тексті, використовуючи команду \- для визначення в них місця переносу.
  - **hash size** Визначено забагато команд і(або) міток для перехресних посилань.

– **main memory size** Створено настільки складну сторінку, що TEX не в змозі зберігати всю інформацію, яка є необхідною для її формування. Для того, щоб виявити причину, за якою переповнюється пам'ять, рекомендується вставити команду **clearpage** перед місцем виникнення помилки. Потім, можливо, доведеться змінити вихідний файл у місці збою, спростивши побудову сторінок (наприклад, посунути якмога далі одне від одного рисунки і таблиці).

– **pool size** Можливо, використано забагато перехресних посилань і(або) визначено забагато нових команд. Точніше, у міток і команд надто велика сумарна довжина у символах. Спробуйте зробити імена ваших команд і міток корошими. Така діагностика може також з'явитися, якщо пропущено закриваючу дужку, яка відділяє аргументи таких команд, як **setcounter,** **newenvironment** чи **newtheorem**.

– **save size** Команди, командні дужки і групи визначень мають надто глибоку вкладеність. Найчастіше це наслідок зациклювання команд, коли команда викликає сама себе, можливо через кілька рівнів.

- **Text line contains an invalid character**
  Рядок містить заборонені символи. Ймовірно, текст має неправильне кодування.

- **Undefined control sequence**
  Використано невизначену команду. Найімовірніша причина — наявність помилки в імені команди чи імені макропакету. Можливо, не задано файл опису стиля, який містить визначення цієї команди.

- **Use of command doesn't match its definition**
  Використання **command** не відповідає її визначенню. Якщо в **command** вказано одну з команд малювання, слід перевірити синтаксис її аргументів (він відрізняється від синтаксису інших команд). Якщо в **command** вказано команду **@array**, то помилка знаходиться у @-виразі в аргументі процедури **array** чи **tabular**; зокрема, ламкі команди у @- виразі мають бути захищені командою **protect**. Ця помилка може також викликатися незахищеною (за допомогою **protect**) командою з необов'язковим аргументом у рухомому аргументі іншої команди (причому в command може бути вказано ім'я зовсім іншої команди).

- **You can't use 'macro parameter character # in ... mode**
  Використання спеціального символу # у звичайному тексті. Вочевидь, потрібно ввести \#.

11.3 Попереджувальні повідомлення LaTeX'у

На відміну від процедури видачі повідомлення про помилку, під час видачі попереджувальних повідомлень LaTeX не призупиняє роботу. Всі LaTeX'івські попередження починаються з тексту: **LaTeX Warning**. Нижче наводяться у абетковому порядку попередження, що видає LaTeX .

- **Citation ... on page ... undefined**
  Мітка роботи, що цитують в команді **cite**, яку використовують під час побудови списку літератури, не визначена командою **bibitem**.

- **Command name name invalid in math mode**
  Команда з ім'ям **name** не діє в математичній моді. Це попередження може бути також повідомленням про помилку. Воно визначає, що має місце спроба використати в математичному режимі команду, яка призначена для застосування тільки у тексті.

- **Encoding ENC1 has changed to ENC2 for ...**
  Кодування ENC1 замінено на кодування ENC2 для .... У визначенні символьного шрифта використано різні кдуавння. Результатом може бути відмінність математичних символів, які друкує одна й та ж команда (яку визначено за допомогою **DeclareMathSymbol**) у різних математичних версіях.

- **External font font loaded for size size** Зовнішній шрифт **font** завантажено для розміру size. Замість потрібного шрифта з розміром **size** завантажено шрифт **font** фіксованого розміру за правилом підстановки, яке задано в таблицях NFSS (у файлах визначення шрифтів з росширенням fd).

- **Float too large for page by size**
  Плаваючий об'єкт більший за сторінку на **size**. Таблиця чи рисунок виходить за нижню межу сторінки на розмір, що вказано в одиницях **pt**.

- **FontDef file: file...**
  Файл визначення шрифтів: file .... Завантажено файл з іменем file, де задано відповіднісит команд перемикання шрифтів певним зовнішнім шрифтам.

- **Font shape shape in size size not available**
  Шрифт накреслення shape розміром size є недоступним. Вказаний шрифт відсутній. Наступний рядок повідомлення покаже, який шрифт будет використано для заміщення відсутнього. Якщо це повідомленняе викликано декларацією **boldmath**, то відсутній шрифт, можливо, реально не використовують у документі через те, що **boldmath** намагається одночасно

завантаджити математичні шрифти для основного розміру, для індексів та індексів в індексах.

- **Font shape shape1 undefined. Using shape2 instead**
  Шрифт накречлення shape1 не визначено. Замість нього використовують shape2. Система завантаження шрифтів прийняла рішення про заміну відсутнього накреслення іншим. Правила підбору шрифтів, які замінюють, визначено у файлах визначення шрифтів (з розширенням fd).

- **Label '...' multiply defined**
  Кілька команд **label** чи **bibitem** мають один і той же аргумент.

- **Label(s) may have changed. Rerun to get cross-references right**
  Числа, які виводяться командами **ref,** **pageref** або **cite**, можуть виявитися хибними. Рекомендується виконати ще один прогін через LaTeX.

- **Marginpar on page ... moved**
  Зноска на полях, що створюється командою **marginpar**, буде зсунута вниз, щоб попередити вивід її поверх попередньої зноски. Таким чином, ця зноска не виявиться напроти того рядка, на якому зустілася команда **marginpar**.

- **Missing character: There is no glyph in font font!**
  Пропущено символ: немає символу **glyph** ц шрифті **font**! У вказаному шрифті відсутній вказаний символ.

- **No \author given**
  Не задана команда **author**. Команді **maketitle** не передує команда **author**.

- **No \title given**
  Не задано команду **title**. Команді **maketitle** не передує **title**.

- **Optional argument of \twocolumn too tall on page page**
  Необов'язковий аргумент команди **twocolumn** на стор. page надто великий. Значення необов'язкового аргументу команди **twocolumn** перевищує висоту боксу, який міг б вміститися на сторінці.

- **Oval too small**
  Команді **oval**, яку використовують у командних дужках **picture**, доручено намалювати настільки маленький овал, що виконати це неможливо. Тому овал буде дещо більший за заданий.

- **Overwriting font in version version**
  Перевизначено font у версії version. Змінено накреслення символьного шрифту чи математичної абетки font у версії version.

- **Redeclaring math alphabet alphabet**
  Перевизначено математичну абетку alphabet. Змінено шрифт, який використовує математична абетка alphabet. Можливо, зміну здійснено завантаженим пакетом.

- **Redeclaring math symbol symbol**
  Перевизначено математичний символ symbol. Змінено команду **symbol**, яка друкує один з математичних символів. Можливо, зміну здійснено завантаженим пакетом.

- **Redeclaring math version version**
  Перевизначено математичну версію version. Змінено версію version математичних шрифтів. Можливо, зміну здійснено завантаженим пакетом.

- **Redeclaring symbol font name**
  Перевизначено символьний шрифт name. Змінено символьний шрифт з іменем name в усіх математичних версіях. Можливо, зміну здійснено завантаженим пакетом.

- **Reference key on page page undefined**
  Посилання key на сторінці page не визначено. Аргумент команди **ref** чи **pageref** не визначено командою **label**.

- **Size substitution with differences to size have occurred**
  Здійснено підстановку шрифтів с невідповідністю розмірів до size. Виявлено хоча б одну підміну шрифтів з суттєвою різницею у розмірах. Величина size означає максимальну різницю для всієї множини здійснених підмін.

- **Some font shapes were not available, defaults substituted**
  Шрифти певних накреслень недоступні, замінені тими, що використовують за замовченням. Це повідомлення виводиться в кінці компіляції документу, якщо здійснювалася автоматична підстановка відсутніх шрифтів.

- **There were multiply-defined labels**
  Були багаторазово визначені мітки. Це попередження друкується під час завершення компіляції, якщо дві команди **label** використовували одну й ту ж мітку.

- **There were undefined references or citations**
  Були невизначені мітки чи посилання на літературу. Це попередження друкується під час завершення компіляції, якщо були знайдені команди **ref** або **cite**, які не мають відповідних **label** чи **bibitem**.

- **Try loading font information for ENC+family**
  Пошук інформації для завантаження шрифтів сім'ї **family** у кодуванні

ENC. Це попередження записується у файл протоколу у всіх випадках, коли компілятор намагається завантажити файл визначення шрифтів для заданої комбінації. Файл має розширення fd, а його ім'я отримуємо склеювання слів ENC і family. і

- **You have requested release date1 of package, but only release date2 is available**
  Запитана версія package, яку датовано date1, але доступна тільки версія від date2.) Отримайте новішу версію класу чи пакету package, який викликав це повідомлення.

11.4 Попереджувальні повідомлення TEX'у

TEX свої попередження ніяк не передує і роботу не призупиняє. Нижче наведено попередження, які видає TEX.

- **Overfull \hbox ...**
  Текст не вміщюється в горизонтальний бокс. Найчастіше така ситуація виникає, коли TEX не може вдало розбити абзац на рядки. Виправити ситуацію можна кількома способами:
  - задати додаткові місця дозволених переносів слів;
  - поставити примусовий чи кращий розрив рядків у певних місцях;
  - якщо таких попереджень виникає забагато, можна збільшити ширину сторінки. У цьому повідомленні виводиться також інформація про те, як переповнений бокс. Якщо це переповнення не дуже велике, то повідомлення можна ігнорувати.
- **Overfull \vbox ...**
  TEX не може знайти вдалого місця для переходу на новую сторінку, тому на сторінку виводиться текст, що перевищує розмір сторінки. Рекомендується використати відповідні команди для визначення найбільш вдалих місць для переходу на наступну сторінку.
- **Underfull \hbox**
  Горизонтальний бокс майже або геть не заповнений. Таке повідомлення може бути викликане, наприклад, використанням команди **\linebreak** в рядку, що заповнений менше, ніж наполовину.
- **Underfull \vbox**
  TEX не може знайти вдалого місця для переходу на нову сторінку, й створює сторінку з недостатньою кількістю тексту на ній. В такому випадку рекомендується використовувати відповідні команди для позначення відповідних місць для переходу на наступну сторінку.

Перелік джерел посилань

1. Рудик О. Б. Встановлення і перші кроки використання LaTeX / О. Б. Рудик // Комп'ютер у школі та сім'ї, № 1, 2012. — С. 47–51

2. Ткачук В. М. Практикум на ЕОМ, Частина 1. Видавнича система LaTeX / В. М. Ткачук., О. М. Ткачук. — Івано-Франківськ : Видавництво Прикарпатського національного університету імені Василя Стефаника, 2012. – 178 с.

3. Подошвелев Ю. Г. Система LaTeX. Інтерактивний електронний посібник / Ю. Г. Подошвелев. — Полтава : Полтавський національний педагогічний університет імені В.Г. Короленка, 2016. – 189 с.

4. Г. М. Губаль Математичні тексти та рисунки в системі LaTeX// Комп'ютерно-інтегровані технології: освіта, наука, виробництво, вип. № 32, 2018. — С. 90 – 94

5. Морозов Д. К. Подготовка документов в издательской системе Латех / Д. К. Морозов, А. Я. Пархоменко. — Ярославль : ЯрГУ им. П. Г. Демидова, 2011. — 96 с.

6. Львовский С. М. Набор и вёрстка в системе LaTeX/ С. М. Львовский — Москва, 2003, — 448 с.

7. Балдин Е. М. Компьютерная типография LATEX / Е. М. Балдин — Новосибирск, 2008, 2012, 2013. — 308 с

8. Грицаенко И. А. LaTeX. Руководство для пользователей. Ч. 1 / И. А. Грицаенко, С. В. Клименко. — 114 с.

9. Вишневський В. В., Романенко Т. М. Ітераційний алгоритм побудови кривої Безьє // Матеріали Сьомої Всеукраїнської Міжнародної конференції «Оброблення сигналів і зображень та розпізнавання образів Укробраз2004», Киів — 2004 — С. 205 – 208

10. Рябошапка Д. С., Барановская Л. В. Кривые Безье // Третя міжнародна науково-практична конференція «Математика в сучасному технічному університеті» 26–27 грудня 2014 року, Київ С. 100 – 101

11. Дєєв С. С., Крівцов В. В. До питання використання кривих Без'є зв'язаними дугами та параметричними поліномами при викладанні курсу «Машинна графіка» // Вісник Національного університету водного господарства та природокористування Випуск 3(59) 2012 р. Серія «Технічні науки» — С. 3 – 10

12. Рудик О. Б. Вектона графіка в Latex засобами tikz / О. Б. Рудик // Комп'ютер у школі та сім'ї, № 8, 2012. — С. 35–38

13. Tantau T. The TikZ and PGF Packages. Manual for version 2.10 Institut für Theoretische Informatik, Universität zu Lübeck. October 25, 2010 (http://sourceforge.net/projects/pgf).

Предметний покажчик

Електронне навчальне видання

LaTeX в дії.

Методичні рекомендації

з використання видавничої системи LaTeX

для студентів, науковців, викладачів

Упорядник: НІКІТЕНКО Олександр Миколайович

Відповідальний випусковий І.В. Руженцев

Редактор Т.С. Малюк

Комп'ютерна верстка О.М. Нікітенко

План 2018 (друге півріччя), поз. 17

Підп. до друку 04.07.2018. Формат 60 × 84 1/16.
Умов. друк. арк. 7,4. Облік. вид. арк. 6,7.
Зам. № 2—17

ХНУРЕ, Україна, 61166, Харків, просп. Науки, 14

Відредаговано у редакційно-видавничому відділі ХНУРЕ.
61166, Харків, просп. Науки, 14