

УДК 004.4:681.322

## КИРИЛИЦЯ В L<sup>A</sup>T<sub>E</sub>X. ПРОБЛЕМИ. УСПІХИ. ПРАКТИКА

**Азаренков В.І.**

к.т.н., доцент, Кафедра системного аналізу  
та інформаційно-аналітичних технологій,

Національний технічний університет «Харківський політехнічний інститут»

***Анотація.** порушено питання кирилізації L<sup>A</sup>T<sub>E</sub>X. Показано проблеми та існуючі шляхи їх вирішення. Проведено аналіз найбільш популярних підходів до набору та верстки наукових текстів у кирилиці. З відкритих джерел зібрано, перекладено, сформовано та запропоновано посібник з підготовки текстів у L<sup>A</sup>T<sub>E</sub>X із урахуванням національних правил верстки та видання наукової літератури.*

***Ключові слова:** T<sub>E</sub>X, L<sup>A</sup>T<sub>E</sub>X, N<sub>C</sub>S<sub>E</sub>T<sub>E</sub>X, N<sub>C</sub>STOOLS, N<sub>C</sub>CCLASS, КИРИЛІЗАЦІЯ, ВЕРСТКА, СИСТЕМА ВЕРСТКИ.*

### **Вступ**

Навіщо потрібен L<sup>A</sup>T<sub>E</sub>X? Цей інструмент використовується повсюди для створення наукових документів, написання книг, а також багатьох інших форм публікацій. Він дозволяє не тільки створювати гарно оформлені документи, а й дає користувачам можливість дуже швидко реалізовувати такі складні елементи друкованого набору, як математичні вирази, таблиці, посилання і бібліографії, отримуючи узгоджену розмітку за всіма розділами.

Переваги використання L<sup>A</sup>T<sub>E</sub>X настільки великі, що протягом 30 років він залишається найактуальнішим інструментом для оформлення технічно складних за версткою робіт у багатьох областях.

Видавнича система L<sup>A</sup>T<sub>E</sub>X дає змогу отримати документ високої якості за розумний проміжок часу навіть без залучення фахівця-верстальника. При цьому для створення складних текстів не буде потрібно багато часу на вивчення можливостей L<sup>A</sup>T<sub>E</sub>X і оформлення потрібного документа високої якості. На відміну від редакторів та інших програм верстки, L<sup>A</sup>T<sub>E</sub>X від самого початку орієнтований на автоматичне дотримання всіх правил набору і друкарських норм (встановлених за замовчуванням у пакеті, що допускають, якщо є бажання, будь-які корекції; водночас кожна істотна зміна в тексті й посиланнях приводить до автоматичного переверстування тексту всього документа без вашого втручання) та створення естетично оформлених документів. Верстка за його допомогою забирає у кілька разів менше часу та зусиль, ніж та ж сама робота, виконана у звичайних редакторах, водночас даючи істотно кращі (в плані акуратності, оформлення, легкості читання і т.д.) результати.

## Мета та задачі дослідження

Основним завданням даної роботи є аналіз порівняння різних версій пакету  $\text{\LaTeX}$  щодо повної його кирилізації під українські стандарти набору тексту та складання короткого керівництва для його широкого використання.

## Основна частина

Здавалося б, можна просто зосередитися тільки на наборі тексту. Але виявилось, що від самого початку існує маленька перешкода, що призводить до великих наслідків. Причому не для всіх, а лише для тих, хто пише кирилицею.

Спочатку  $\text{\TeX}$  був створений для мов, що використовують латинський алфавіт. Пізніше кількість можливих для використання мов у пакеті значно зросла. Проблеми виникли з кириличними шрифтами через особливості їхнього кодування (розташування в таблиці кодування) і написання. Основна проблема при створенні україномовних документів у  $\text{\LaTeX}$  – це мала кількість якісних шрифтів із підтримкою кирилиці. Створений пізніше пакет  $\text{\TeX Live 2013}$  частково розв'язує цю проблему, а розроблені пакети  $\text{\XeTeX}$  та  $\text{\XeLaTeX}$ , використовуючи Unicode кодування, домоглися повного вирішення мовних шрифтових проблем кодування, але повністю не змогли розв'язати завдання верстки з урахуванням наших національних особливостей оформлення тексту. Проблема наявності різноманітності накреслення українських літер залишилася. Код букви є, а реалізація її накреслення різними шрифтами, зокрема й математичними, – відсутня. Але це ще не все.

Документи, отримані в американському, європейському та українському наборах, серйозно відрізняються; ці відмінності визначаються як національними традиціями, так і державними стандартами. Відмінності стосуються насамперед наборів не лише шрифтів, а й повноцінних наборів символів для української мови: №, %, двох видів тире і дефіса, двох різновидів лапок тощо. Необхідно враховувати таблиці та правила переносів, особливості написання математичних символів і функцій, назв і оформлення окремих частин тексту, заголовків, розділів і підрозділів, їх нумерацію, підписуноків написи та назви таблиць, пропуски, відступи й розділові знаки та багато чого іншого.

Здавалося б, у чому проблема? Вона і виїденого яйця не варта. На жаль, ні. Коли  $\text{\TeX}$  тільки з'явився, все було просто: його текстові шрифти складалися з 128 символів, містили символи кодування ASCII і невелику кількість додаткових символів. Тепер же в  $\text{\LaTeX}$ -і підтримуються десятки мов, та забезпечити точний збіг вхідного коду символу з його кодом у шрифті  $\text{\TeX}$ 'а вже не вдається. З цієї причини доводиться використовувати пакет **inputenc**, призначений для встановлення відповідності між кодами не-ASCII-символів (символів кирилиці), які використовуються під час набору тексту, і їхніми кодами в текстових шрифтах.

Пакет `inputenc` перетворює кожен не-ASCII-символ на команду. Коли  $\TeX$  трансліює такий документ, команди, пов'язані з не-ASCII-символами, підставляють замість себе коди, що використовуються у шрифтах  $\TeX$ -а. У разі використання кирилиці ситуація ще більше ускладнюється, оскільки є не тільки різноманіття вхідних кодувань, вихідні кодування шрифтів із кирилицею так само різняться. Тому процес зіставлення вхідного коду і коду в шрифті стає двокроковим: під час трансляції символ кирилиці замінюється на спеціальну команду, а потім ця команда підставляє замість себе код шрифту за правилами, визначеними пакетом **fontenc**. Наприклад, слово «Задача» (**Zadaha**) перетворюється в процесі обробки на послідовність команд

`\CYRZ\cyra\cyrd\cyra\cyrch\cyra.`

Ця особливість не дає змоги нормально реалізувати пошук, сортувати предметний покажчик, що містить терміни українською мовою: українські літери впорядковуватимуться не за алфавітом, а за іменами відповідних їм команд...! І якщо літери кирилиці працюють як команди, то їх не можна використовувати в іменах міток при перехресних посиланнях. На доповнення до цього в стандартних математичних шрифтах  $\TeX$ -а символи кирилиці, на жаль, теж відсутні. У результаті в математичних формулах вони зникають. Це дуже підступна помилка, причому виявити її візуально неможливо, а знайти такий символ у тексті документа вкрай важко. І це далеко не всі складнощі з кирилицею. Їх в міру прояву намагаються долати різними шляхами.

Але якщо система верстки така недосконала, то чому її так люблять використовувати для створення документів? Переваг у неї досить багато, але я назву найголовніші:

- $\LaTeX$  це система, що безплатно розповсюджується і не має на сьогодні програмних помилок;
- системні вимоги:  $\LaTeX$  працює на всьому комп'ютерному залізі (навіть застарілому), що «ворушиться», і в будь-якій операційній системі;
- стабільність, повна сумісність з усіма операційними системами, незалежно від своїх версій;
- тільки набираючи самостійно ( $\LaTeX$  це дозволяє всім охочим без сторонніх верстальників) можна бути абсолютно впевненим в очікуваному результаті і одразу його виправити;
- документ  $\LaTeX$  – це текст. Уся робота з  $\LaTeX$  може виконуватися без відриву рук від клавіатури. Згадайте, скільки разів Ви помилялися в значеннях у формулах та були змушені перевіряти весь документ MS Word у пошуках формул, де зустрічається це значення, і виправляти його вручну. У  $\LaTeX$  потрібно лише вказати масову заміну значень. У всьому документі все буде виправлено автоматично;
- $\LaTeX$  формули – це також текст. І можливості набору складних формул у  $\LaTeX$  НЕ ОБМЕЖЕНІ;

– одна з найвагоміших причин, через яку багато хто використовує  $\text{\LaTeX}$ , полягає у відокремленні змісту документа від його стилю. Це означає, що після написання тексту, можна з легкістю змінювати його зовнішній вигляд. Аналогічним чином, можна створити один стиль документа і використовувати його для стандартизації зовнішнього вигляду інших. Це дає змогу науковим журналам створювати шаблони для пропонованих на розгляд матеріалів. Такі шаблони мають задану розмітку, внаслідок чого додати залишається лише зміст. Насправді існують сотні подібних шаблонів, починаючи з різноманітних резюме і закінчуючи презентаціями слайдів;

– і т.д., переваги можна перераховувати ще дуже довго...

Як видно з наведених переваг, за таку систему верстки варто поборотися.

Зазначимо основні моменти, які у тому чи іншому вигляді мають бути присутніми у кирилізації.

1. Підключити схему національних переносів.
2. Розробити та підключити кириличні шрифти.
3. Ввести кириличні ялинки « » та лапки „ “.
4. Визначити знак номера №, що у англійській мові відсутній.
5. Ввести більш коротке, порівняно з англійським, тире – з відповідними відбівками з обох боків.
6. Перевизначити команди секціонування для отримання національних назв: Глава, Параграф, Розділ, Вступ, Передмова, Зміст тощо.
7. Крім  $\tan$ ,  $\sinh$  тощо, ввести визначення математичних функцій  $\text{tg}$ ,  $\text{sh}$  тощо.
8. Перевизначити команду `\today` для правильного отримання дати українською мовою.
9. Додати команди `\Asbuk` та `\asbuk` для нумерації літерами українського алфавіту.

Сподіваюся, я розвіяв думку про те, що для українізації пакета достатньо підключити українські шрифти. Зрозуміло, робота з кирилізації цим не обмежується: потрібно пристосувати програму *bibtex* для отримання списку літератури національною мовою з відповідним чином оформленими бібліографічними описами та впорядкуванням як за країною, так і англійською абеткою; потрібно українізувати програму *makeindex* для отримання предметного та іменного покажчика (якщо йдеться про книгу); потрібно відповідно до вітчизняних традицій переробити стандартні класи документів, зокрема *article* і *amsart*, і багато іншого, про що в цьому короткому аналізі нам не вдасться розповісти через брак місця. З усіх питань, що виникають у вас, рекомендуємо звертатися до списку розсилки групи користувачів кириличного  $\text{\TeX}$ -а *CyrTeX*. Щоб підписатися на нього, потрібно надіслати e-mail на адресу [CyrTeX-ru-on@vsu.ru](mailto:CyrTeX-ru-on@vsu.ru) із зазначенням „Subscribe“ у полі *Subject*.

Навіть опанувавши озвучену вище інформацію, не можна одразу зрозуміти, в яких пакетах розширення системи  $\text{\LaTeX}$  потрібно готувати свою статтю українською мовою, щоб отримати максимально правильну українізацію

оригінал макета (мається на увазі повне дотримання національних правил виконання та підготовки видання до друку). Виникає питання, а чому я одразу не уточнив і не перерахував цю інформацію?

На жаль, розбрід та хитання характерні зараз для всіх сфер нашої української дійсності. Не зуміли домовитися і розробники українізації L<sup>A</sup>T<sub>E</sub>X-а: версій з'явилося багато – начебто це добре, але вони не сумісні, неповні і не можуть доповнювати одна одну (вирішують різні й одночасно однакові завдання) та не призначені для спільної роботи – це погано. Тому не буду їх перераховувати, тому, що до перерахування потрібно буде додати досить повний опис можливостей кожної з них. Цю інформацію легко отримати в Інтернеті.

Одне можу одразу поради: перш, ніж відправляти статтю в той чи інший вітчизняний чи зарубіжний журнал, з'ясуйте, у якому конкретно (з якою кирилізацією) пакеті L<sup>A</sup>T<sub>E</sub>X потрібно її оформляти. На жаль, поки що не існує єдиного кирилізованого та вузько-спеціалізованого українізованого пакета L<sup>A</sup>T<sub>E</sub>X, яким користувалися б у всіх видавництвах. Якщо конкретна кирилізація видавництвами не обговорюється, ви готуєте статтю за тією, в якій звикли працювати; якщо кирилізація вказана, то попросіть її у видавництва разом із інструкцією користувача.

Наведемо опис модифікацій L<sup>A</sup>T<sub>E</sub>X, що досить часто використовуються, – від більш старих до більш нових.

1. **rlatex Шеня-Львівського**. У лістинг L<sup>A</sup>T<sub>E</sub>X необхідно додати до заголовка файлу

`\usepackage{rusccorr}` та компілювати командою `rlatex`.

*Переваги*. Розумно і повно описано у підручниках L<sup>A</sup>T<sub>E</sub>X російською мовою. У пакеті, що розповсюджується, коректно вирішено більшість (але не всі) проблем кирилізації. Стабільно однаково працює на різних дистрибутивах з 1993 року і до цього часу.

*Недоліки*. Необхідні неочевидні дії при встановленні – треба встановлювати спеціальні додатки (`rusccorr` та ін), будуть складнощі при використанні пакета `hyperref`. Якщо у вас не встановлено цю русифікацію, то не потрібно намагатися її додавати до наявних. Відсилаючи комусь файл у цій кирилізації, обов'язково попередьте його.

2. **rusllh Черепанова**. У лістинг першим рядком файлу додати

`%& --translate-file=win2t2 або koi2t2, або alt2t2`

у залежності від кодування, що використовується, при цьому треба перевірити доступність відповідних файлів з розширенням `tcx`); у заголовок додати

`\usepackage{rusllh}`

і, за бажанням,

`\usepackage{ruscorr}`.

*Переваги.* Можна без особливих зусиль використати з будь-яким дистрибутивом. Кириличні літери можна використовувати в іменах команд, міток тощо. Українські літери розумно записуються у файли типу .aux, .toc, .idx тощо.

*Недоліки.* Необхідно використати нестандартний пакет. Кодування utf-8 не підтримується. Можливі помилки з українськими літерами всередині математичних формул. Не повністю працює пакет hyperref (принаймні проблеми з bookmark-ами гарантовані). Можна використовувати цю кирилізацію, якщо вам треба працювати з файлом, де явно використані українські літери в іменах команд або міток. В інших випадках спеціального виграшу вона швидше не приносить. А проблеми будуть.

3. **babel+inputenc.** Використання – до заголовка файлу додати

```
\usepackage[utf8]{inputenc}
\usepackage[ukrainian]{babel}
```

та компілювати командою *pdflatex* або *latex*.

*Переваги.* Стандартний спосіб, який працює out-of-the-box з будь-яким із нині поширених дистрибутивів. Можна користуватися майже будь-яким українським кодуванням (потрібно лише акуратно його вказати пакету inputenc). Сумісно з усіма стандартними пакетами (в т.ч. з hyperref).

*Недоліки.* Українські літери стають активними символами (і їх не можна використовувати у назвах міток, команд тощо). Українські літери у файлах типу .aux, .toc, .idx тощо стають нечитаними (принаймні без окремих програм, що їх перекодують).

4. **X<sub>Y</sub>LaTeX.** Використання – до заголовка файлу додати

```
\usepackage{polyglossia}
\setmainlanguage{rusian}
\newfontfamily\russianfont{PT Sans}
```

(остання команда варіюється в залежності від бажаного шрифту); тим, хто звик до babel, варто додати ще

```
\setkeys{ukrainian}{babelshorthands=true}.
```

Компілювати командою *xelatex*.

*Переваги.* Працює out-of-the-box із досить сучасними дистрибутивами. Українські літери можна використовувати в іменах міток тощо. Можна використовувати будь-який OpenType або TrueType шрифт, наявний у системі.

*Недоліки.* Працює лише з кодуванням utf8. Активно розробляється і, мабуть, не зовсім усталений (радимо стежити за новизною дистрибутива!). Відсутня можливість компіляції PostScript (відповідно, не можна використовувати засоби типу ps2ps). Зі «старими» версіями LaTeX не працює.

Тим, хто не впевнений, яка кирилізація йому потрібна, спільнота радить скористатися варіантом № 3 (3. babel+inputenc). Якщо ж необхідно скомпілювати документ із великою кількістю міток з українськими іменами – найпростіше змусити працювати кирилізацію № 2 О. Черепанова.

Це деякий далеко неповний список видань і пакетів кирилізації (*не сумісних між собою*), що використовувались і використовуються редакціями наукових журналів на сьогоднішній день. Аналіз даних можливостей кирилізації підготовленої до видання літератури з відкритих джерел показав, що найбільш повно та кардинально питання кирилізації L<sup>A</sup>T<sub>E</sub>X-а вдалося вирішити нашому співвітчизнику програмісту та працівнику інформаційно-видавничої сфери доктору фізико-математичних наук Олександру Йосиповичу Роженку шляхом розробки та включення до дистрибутиву L<sup>A</sup>T<sub>E</sub>X-а цілої системи макросів N<sub>C</sub>C-L<sup>A</sup>T<sub>E</sub>X, що є **ncctools** – колекцією загальних пакетів для стандартного L<sup>A</sup>T<sub>E</sub>X: *fterpackage*, *dcounter*, *desclist*, *extdash*, *manyfoot*, *mboxfill*, *nccbbb*, *nccboxes*, *ncccomma*, *ncccropbox*, *ncccropmark*, *nccfancyhdr*, *nccfloats*, *nccfoots*, *N<sub>C</sub>C-L<sup>A</sup>T<sub>E</sub>X*, *nccmath*, *nccparskip*, *nccpic*, *nccrules*, *nccsect*, *nccstretch*, *nccthm*, *ncctools*, *textarea*, *tocenter*, *topsection*, *watermark* [1]. По суті, N<sub>C</sub>C-L<sup>A</sup>T<sub>E</sub>X – розширений клас загального призначення – надає платформу для заміни загального класу стандартних класів статей, книг та звітів, а також додає клас «Препринт», деякі з яких також можуть використовуватися зі стандартними класами. Пакет *ncslatex* також завантажує багато стандартних пакетів і вимагає останньої версії комплекту *ncctools*. При використанні цих макросів практично всі питання щодо кирилізації знімаються автоматично.

Залишається ще непорозуміння: для вивчення L<sup>A</sup>T<sub>E</sub>X-а ми користуємося перекладними книжками з латинських мов, для яких немає проблем із кирилицею, бо вони нею не користуються, а отже, вони ж і не користуються макросами N<sub>C</sub>C, включеними до складу дистрибутивів цієї системи. І, як наслідок, не озвучують, не вчать та не описують їх.

Зі сказаного випливає несподіваний і чудовий висновок: дистрибутиви L<sup>A</sup>T<sub>E</sub>X-а, якими ми з вами користуємося в максимально-можливому об'ємі кирилізовані! І виникає ще один, але сумний висновок: а нам про це ніхто не розповів.

Аналіз доступної літератури про L<sup>A</sup>T<sub>E</sub>X показав, що за описом макросів N<sub>C</sub>C-L<sup>A</sup>T<sub>E</sub>X було видано лише три (!!!) книги малими тиражами [3-4], які миттєво розійшлися і стали бібліографічною рідкістю. Існує розрізнений комплект англomовної документації на сайтах CTAN [1] із кількома оновленнями під сучасний L<sup>A</sup>T<sub>E</sub>X. І все. Вважаючи це несправедливим і не допустимим приховання таких можливостей видавничої системи, ми на підставі доступного матеріалу переклали (переклад виявився дуже складним через використання специфічної термінології) та сформуливали україномовний опис

пакета  $\text{NCC-LAT}_{\text{E}}\text{X}$ , викладеного нижче, тим самим внесли свої п'ять копійок у широке практичне використання кирилізованого пакета.

Посібник за останньою версією пакета  $\text{NCC-LAT}_{\text{E}}\text{X}$  описує лише відмінність його від стандартного  $\text{LAT}_{\text{E}}\text{X}$  і короткий опис застосування макросу в сучасному  $\text{LAT}_{\text{E}}\text{X}$ . У ньому, за твердженням розробника, повністю збережено наступність попередніх версій пакета, тобто будь-які інструкції старих реалізацій  $\text{NCC}$  у  $\text{LAT}_{\text{E}}\text{X}$  працюватимуть нарівні з новими. Наш виклад матеріалу неспроможний претендувати на повноту викладу керівництва з **ncclatex**, так як ми не знайшли цілісного опису всіх макросів та загальної технології їх застосування у доступному друку. Хочеться сподіватися, що він з'явиться найближчим часом.

## 1 $\text{NCC-LAT}_{\text{E}}\text{X}$

Замовлення класу  $\text{NCC}$  здійснюється командою

$\backslash\text{documentclass}[\text{Список опцій}]{\text{ncc}}.$

Перша та головна відмінність від «стандартного»  $\text{LAT}_{\text{E}}\text{X}$  полягає в тому, що підготовка статей, книг і звітів у  $\text{NCC-LAT}_{\text{E}}\text{X}$  здійснюється за допомогою спеціальних опцій, які називаються стилями, класу  $\text{ncc}$  замість використання різних звичайних класів. Клас  $\text{NCC}$  містить 4 стилі дизайну документів, що замовляються такими опціями:

- `article` – стаття;
- `preprint` – препринт. Команда  $\backslash\text{preprint}\{\text{номер}\}$  задає номер препринта. Титул оформляється на окремій сторінці;
- `book` – монографія. Команда  $\backslash\text{bookeditor}\{\text{текст}\}$  дозволяє вставити текст під назвою книги (зазвичай це наукові редактори книги);
- `report` – звіт. Еквівалентний стилю `book` з опцією `oneside`.

Опції `titlepage` і `notitlepage` не використовуються в класі  $\text{ncc}$ . Зовнішній вигляд заголовка на окремій сторінці визначається використовуваним стилем. Стиль статті вказується заголовком, що біжить, а інші визначають заголовок на окремій сторінці. Структура `abstract` визначена у всіх стилях. Вона налаштовується на окремій сторінці, якщо заголовок підготовлений на окремій сторінці.

У всіх стилях оформлення документа, крім `article`, використовується стиль оформлення сторінки `headings`. У стилі `article` використовується `myheadings`.

За замовчуванням встановлюються опції `10pt`, `a4paper`, `article`, `twoside`, `onecolumn`, `final`, `openany`.

Клас встановлює  $\backslash\text{unitlength}$  рівним 1 мм (тобто всі координати в структурі `picture` обчислюються в міліметрах).



## 1.1 Макет сторінки

Клас *NCC підтримує* всі стандартні опції замовлення розміру сторінки: *a4paper*, *a5paper*, *b5paper*, *letterpaper*, *legalpaper*, *executivepaper*, *landscape*. Додатково є опція *a5a4paper*, що застосовується для друку тексту формату A5 на принтері формату A4, що має центровану кишеню введення.

На відміну від стандартних класів, опції замовлення розміру сторінки не змінюють розмірів текстового поля. Розміри текстового поля визначаються за базовим кеглем документа (табл. 1).

Таблиця 1 – Розміри текстового поля

| Розмір пункту, пт | Ширина тексту, мм | Висота тексту, мм |
|-------------------|-------------------|-------------------|
| 10                | 110               | 157               |
| 11                | 126.5             | 199               |
| 12                | 145               | 233               |
| 14                | 160               | 240               |

Щоб перерахувати розміри текстового поля на повну сторінку, використовуйте опцію *fittopage*. Текстове поле вирівнюється по центру сторінки, беручи до уваги також поле заголовка, але відступ зверху коригується так, щоб він становив не більше 1,5 дюймів. Якщо ви хочете центрувати текстове поле з урахуванням розмірів поля маргінальних виносков і нижнього колонтитула сторінки, використовуйте команду

`\ToCenter[hfm]{textwidth}{textheight}`

у преамбулі документа. Необов'язковий параметр *[hfm]* містить будь-яку комбінацію з цих трьох літер, що означають назви полів, які мають бути враховані під час центрування тексту: *h* – враховувати колонтитул (header), *f* – враховувати нижній колонтитул (footer), *m* – враховувати поле маргінальних виносков. За допомогою цієї команди ви можете змінити ширину і висоту текстового поля та відцентрувати результат на сторінці.

Інший спосіб змінити ширину і висоту – використовувати команду `\FromMargins`. Наприклад, така команда

`\FromMargins[hf]{20mm}{10mm}{25mm}{15mm}`

обчислює розміри текстового поля і відступи таким чином, щоб забезпечити відстань 20 мм від лівого краю сторінки, відстань 10 мм від правого краю, відстань 25 мм від верху і відстань 15 мм від низу, припускаючи, що використовуються заголовок і нижній колонтитул. Під час двостороннього друку відстань ліворуч і праворуч міняється місцями для парних сторінок.

## 1.2 Параметри розміру базового кегля

Поряд зі стандартними параметрами розміру базового кегля 10pt, 11pt і 12pt, вводиться параметр розміру 14pt. У 14 pt розміри шрифту `\huge` і `\Huge` дорівнюють `\LARGE`.

Іноді стандартні розміри шрифтів, що використовуються в командах розмітки розділів, дещо більші, ніж необхідно. Цей випадок виникає, коли розмір сторінки досить малий або розмір базового шрифту занадто великий. У цьому випадку можна використовувати опцію `small`, щоб зменшити розмір шрифту, який використовується в командах розмітки розділу. Застосовується параметр `14pt`.

### 1.3 Інші параметри

Клас `NCS` підтримує такі стандартні опції: `oneside`, `twoside`, `draft`, `final`, `openright` (у стилі `book` новий розділ починається з непарної сторінки, у стилі `article` порожня парна сторінка додається в кінець статті за потреби), `openany` (немає вирівнювання по непарній сторінці), `onecolumn`, `twocolumn`, `openbib`, `fleqn`, `leqno`.

Опція `russian` замовляє кириличний стиль оформлення заголовків, завантажує пакет багатомовної підтримки `Babel` з опцією `russian`, вмикає `\frenchspacing` (без додаткових відбитків після кінця речення) і перевизначає команди `\alph`, `\Alph` на алфавітну нумерацію кирилицею (нумерація латиницею зберігається в командах `\alphlatin`, `\Alphlatin`).

### 1.4 Пакети, що завантажуються ncc класом

Клас `ncc` завантажує безліч пакетів із пакетів `nclatex` і `ncctools`. Наприклад:

- `ncclatex` – є ядром `NCS-LATEX`;

- `nccltrus` – завантажується, коли використовується опція `russian`.

Перевизначає назви підписів на кирилиці;

- `nccold` – надає деякі застарілі команди зі старого `NCS-LATEX`, а саме `\no`, `\eref`, `\Eq`, `\Eqs`, `\Eqalign`, `\tbox`, `\bbox`, `\tbbbox`, `\emline` і `\emlinewidth`;

- `dcounter` – забезпечує динамічні лічильники;

- `desclist` – покращує структуру `description` і надає структуру `desclist` з кращим контролем маркерів в описах;

- `extdash` – надає команди швидкого доступу `\-/`, `\--`, `\---`, `\=/`, `\==` і, and `\===` для кращого управління переносами складних слів;

- `nccboxes` – визначає додаткові бокси;

- `nccfloats` – надає команди `\minifig`, `\fig`, `\figs`, `\minitabl`, `\tabl` і `\tabls`, які охоплюють стандартні плаваючі фігури `LATEX` і готують плаваючі матеріали в стилі `NCS-LATEX` (текст у плаваючих фігурах `NCS-LATEX` розміщують по центру, готують його у форматі `\footnotesize`). Нові функції `\sidefig` і `\sidetabl` спрощують створення плаваючих міні-фігур, розміщених на зовнішній стороні сторінки;

- `nccfoots` – надає команди `\Footnote`, `\Footnotemark` і `\Footnotetext` для ручного керування маркерами виносок;

- `nccmath` – розширює пакет **`amsmath`** командами `NCS-LATEX` для підготовки виключних рівнянь (`\eq`, `\eqs`, `\eqalign`), перевизначає структуру

`eqnarray` для правильної роботи разом з виключними рівняннями *AMS*, а також вводить структури `darray`, `fleqn` і `seqn`;

- `nccpic` – є оболонкою для пакета **graphicx** (надає списки розширень для корисних dvi-драйверів, вводить команду `\putimage` для сумісності зі старим *NCC-LATEX* і визначає команду `\ipic`);

- `nccsect` – забезпечує поліпшене управління розділами, заголовками та змістами;

- `nccthm` – теореми в стилі *NCC-LATEX* (настроюваний макет, безліч нових функцій);

- `toccenter` – надає команди `\ToCenter` і `\FromMargins`;

- `watermark` – надає водяні знаки.

Клас **ncc** завантажує деякі стандартні пакети *LATEX*. Це **amsmath**, **graphicx** і **makeidx**. Пакет **babel** також завантажується, якщо використовується опція `ruussian`.

### 1.5 Нові команди ncc класу

Ми описуємо тут нові команди, представлені тільки в класі **ncc**. Інші нові команди представлені в пакетах збірок *ncclatex* і *ncctools* (більшість із них є спільними для класу **ncc** і пакета **nclatex**).

`\partmark{текст}` використовується для керування мітками верхнього і нижнього колонтитула після нової частини.

`\openrightorany` виконує команду `\clearpage` або `\cleardoublepage` залежно від режиму, обраного в опціях класу (`openright` або `openany`). Якщо створюється порожня парна сторінка, то текст у її колонтитулі буде порожній.

`\AuthorBeforeTitle` and `\TitleBeforeAuthor` керують порядком виведення назви та списку авторів командою `\maketitle`. Використовуються тільки в преамбулі документа. У стилі *article* спочатку йде назва, а потім автори. В інших стилях усе навпаки.

`\noeqbreak`, `\alloweqbreak` керують можливістю розриву сторінки перед виключною формулою. Перша команда забороняє розрив, а друга дозволяє. За замовчуванням діє `\alloweqbreak`.

`\SetTOCStyle{стиль}` задає стиль оформлення змісту, списку рисунків і списку таблиць. Використовується в преамбулі документа. Стиль за замовчуванням порожній. Наприклад, щоб вивести зміст дрібним шрифтом `\small`, слід скористатися командою `\SetTOCStyle{\small}`.

`\setyear{рік}`, `\theyear`: перша команда задає рік, а друга – друкує заданий рік. Команда `\setyear` не змінює лічильник `\year`. За замовчуванням команда `\theyear` друкує поточний рік.

`biblist` – це нова структура:

```
\begin{biblist}[початковий номер]{nprototin} . . . \end{biblist}.
```

Задає друк чистого бібліографічного списку без заголовка. Перший необов'язковий параметр встановлює початкове значення лічильника бібліографії. Стиль оформлення бібліографії задається командою `\bibliststyle` (за замовчуванням її значення `\small`). Структура `thebibliography` створена на основі `biblist`.

Додано структуру `theglossary`. Команди `\printindex`, `\see` і `\seealso` завантажуються пакетом **makeidx**. Введено команду `\printglossary`.

Лічильники `figure` і `table` оголошуються динамічно. Стиль їхньої нумерації встановлюється командою `\countstyle` у преамбулі документа. За замовчуванням у стилях `article` і `preprint` встановлено `\countstyle{}`, а в стилях `book` і `report` – `\countstyle{chapter}`. Лічильник рівнянь також управляється командою `\countstyle`.

**Команди, специфічні для стилю `article`.** Три нові команди.

`\setseries{назва}`, `\setvolume{номер тому}` и `\setissue{номер випуску}` керують серією, томом і номером збірника, у якому видається стаття. Ці команди зазвичай задаються в локальному файлі **nccadd.sty**, який автоматично завантажується в стилі `article`. Виведення цієї інформації виконується відповідно командами `\theseries`, `\thevolume`, `\theissue`. У режимі `openright` наприкінці статті за потреби додається порожня парна сторінка.

**Команди, специфічні для стилю `preprint`.** Команда `\preprint{Preprint No}` задає номер препринта. Якщо номер задано, то під титулом препринта буде додано текст.

### Preprint number

**Команди, специфічні для стилів `book` і `report`.** Можна використовувати команди `\frontmatter`, `\mainmatter`, і `\backmatter`.

Команда `\bookeditor{текст}` дає змогу вставляти фрагмент тексту на титульній сторінці під заголовком монографії та списком авторів. Можна керувати способом виведення тега назви розділу в колонтитулі та змісті книги. За замовчуванням цей тег складається з номера розділу. Але можна зробити, щоб перед номером ішов префікс «Розділ». Для цього служить команда `\ChapterPrefixStyle{список}`, у списку якої через кому можна вказувати два слова: `header` і `toc`. Зазначивши `header`, Ви вмикаєте виведення префікса в колонтитулі, а вказавши `toc`, задаєте виведення префікса в змісті.

## 1.6 Використання незвичайних заголовків із класом `ncc`

Заголовки в класі **ncc** підтримуються дещо інакше, ніж у стандартних класах. Тому, якщо ви хочете використовувати незвичайні заголовки разом із класом **ncc**, введіть

```
\usepackage[список стилів сторінки]{ncchdr}
```

у преамбулу документа. Параметр *список стилів сторінки* (*list-of-page-styles*) повинен містити список стилів сторінки, які необхідно перевизначити, тобто щось із `empty`, `plain`, `headings` і `myheadings`. Останній з перерахованих стилів заодно встановлюється як поточний стиль.

Примітка: стилі `headings` і `myheadings` мають спільну основу в макросі `\NCC`. Зазначивши один із них у параметрах завантаження пакета, Ви автоматично замовляєте лінійку і для іншого. Додатковий незвичайний стиль заголовка сторінки поставляється з пакетом **ncchdr**. Він використовує три знаки заголовка, а саме `\lefttitlemark`, `\titlemark` (у центрі) і `\righttitlemark`. Щоб завантажити цей стиль, використовуйте опцію `title` у пакеті.

Пакет заснований на пакеті **nccfancyhdr** із набору *ncctools*.

## 2 Пакет **ncclatex**

Більша частина класу **ncc** завантажена в пакет **ncclatex**. Щоб забезпечити майже всю функціональність `\NCC-LATEX` за допомогою стандартних класів `LATEX`, ви можете використовувати цей пакет. Пакет також завантажує безліч пакетів. Легше сказати, які пакети він не завантажує зі списку, зазначеного в розділі 1.4. Не завантажуються лише три з них, а саме **nccold**, **tocenter** і **watermark**.

Якщо ви хочете передати додаткові опції пакета, завантаженого з **ncclatex**, задайте його командою `\usepackage` перед завантаженням **ncclatex**.

Пакет **ncclatex** має дві опції: `small` і `ukrainian`. Використання опції `small` забезпечує дрібніший шрифт у командах розмітки секцій. Опція `ukrainian` намагається завантажити пакет `babel` з необов'язковим параметром `[ukrainian]`, перевизначає підписи математичних тверджень українською мовою, завантажуючи пакет `pscltrua`, та перевизначає команди `\alph` та `\Alph` для створення українських буквено-цифрових чисел. Оригінальні латинські змінні буквено-цифрової нумерації зберігаються в `Alphatin` та `Alplatin`. Пакет встановлює `\unitlength` рівним 1 мм (міліметри – одиниці за замовчуванням у `\NCC-LATEX`).

### 2.1 Розділи, заголовки та зміст у `\NCC-LATEX`

#### 2.1.1 Розділи

Команди розмітки розділів рівня 0 і вище перевизначаються за допомогою пакета **nccsect** з *ncctools* (команда рівня 0 – `\chapter` або `\part` залежно від використовуваного класу; команди вищих рівнів – `\section`, `\subsection`, . . . . , `\subparagraph`). Усі команди розмітки секцій поділяються на дві категорії: виключні секції (до `\subsubsubsection`) і активні секції (`\paragraph` і `\subparagraph`).

Пакет використовує плаваюче горизонтальне вирівнювання для виключних секцій, яким управляють за допомогою

`\sectionstyle{стиль},`

де *стиль* – один із таких варіантів:

- `hangindent`      стандартний стиль  $\text{\LaTeX}$ ;
- `hangindent*`      аналогічно попередньому, але без правого вирівнювання;
- `parindent`          рубрики вимкнені на величину параграфного відступу;
- `parindent*`          аналогічно попередньому, але без правого вирівнювання;
- `center`              рубрики центруються.

Команди `\indentaftersection` і `\noindentaftersection` керують відступом після секцій. Пакет **nccsect** встановлює стиль і абзац на стандартні значення  $\text{\LaTeX}$  за замовчуванням, `\sectionstyle{hangindent}` і `\noindentaftersection`. Але пакет **ncclatex** використовує `\sectionstyle{hangindent*}`.

Пакет **nccsect** істотно покращує управління номерами секцій, тегів назви розділів і записом в аух-файл:

- якщо Ви не хочете поміщати текст рубрики в колонтитул, напишіть `\norunninghead` перед командою рубрикації;
- якщо Ви хочете замінити текст, що йде у колонтитул, на інший текст, використовуйте команду `\runninghead{інший текст};`
- якщо Ви хочете зробити нумеровану рубрику, яка з'явиться в колонтитулі та змісті документа, скористайтесь модифікатором `\noheadingtag`. Зверніть увагу, що текст використаної команди потрапляє в тег назви розділу і в аух-файл. Це простіший спосіб створення розділу без номера, який має з'явитися в змісті та в заголовку;
- якщо Ви хочете оформити рубрику з нестандартним номером і помістити її в колонтитул і зміст, скористайтесь командою `\headingtag{інший номер}`. Зверніть увагу, що замінений номер іде і в тег назви, розділу, і в аух-файл. Він повинен містити тільки надійні команди. Крихкі команди в ньому мають бути захищені командою `\protect`. Лічильник рубрики в цьому випадку не змінюється;
- якщо Ви хочете повністю підмінити оформлення номера рубрики або підпису (включно з префіксом і суфіксом, що автоматично вставляються командами розмітки розділів і підписів) на власне оформлення, скористайтесь командою `\headingtag*{інший номер}`. При цьому текст рубрики не надходитиме в колонтитул і зміст;
- якщо Ви хочете заборонити запис рубрики в зміст, підпис або аух-файл, використовуйте модифікатор `\skipwritingtonaux`.

### 2.1.2 Заголовки

Команда `\caption`, корисна для плаваючих елементів, також перевизначена в пакеті **nccsect**. Метод створення підпису внутрішньо відрізняється від стандартного методу. У ньому можна використовувати команди переведення рядка, але

переконайтеся, що переведення рядка в підписі не йде в аух-файл (використовуйте необов'язковий параметр з `caption` для визначення аргументу, що йде в аух-файл). Команду `\centering` можна використовувати в заголовках без побоювань (вона автоматично пригнічується під час запису в аух-файл). Команда `\caption*` дає змогу створити підпис без номерного тега (слова «Рисунок» або «Таблиця» належать до тега!). Текст такого напису не потрапляє в аух-файл.

Пакет визначає *різні* команди підписів для таблиць і рисунків. Назва таблиці зазвичай розташовується над таблицею. Тому вертикальний пропуск після назви таблиці необхідний, а пропуск перед назвою таблиці не потрібний. Підпис до рисунка зазвичай розташовується під рисунком. У цьому випадку вертикальні пропуски відрізняються: пропуск перед рисунком необхідний, а пропуск після рисунка не потрібен. Саме з цієї причини підписи до таблиць і рисунків оформляють по-різному.

### 2.1.3 Зміст

Управління змістами також перевизначено в пакеті **nccsect**. Кожен пункт змісту задається двома параметрами: лівим полем і значенням відступу. Пакет **nccsect** обчислює ліве поле для пункту змісту шляхом підсумовування пропусків полів, визначених у пунктах змісту нижчих рівнів. Наприклад, якщо набирається пункт 2-го рівня, то його ліве поле дорівнюватиме сумі відступів, визначених у пунктах 0-го і 1-го рівнів. Це набагато гнучкіший спосіб, ніж стандартний, у якому ліве поле кодувалося тільки в `toc`-команді.

Пропуск полів і пропуск відступу визначаються за допомогою методу прототипу. Цей метод забезпечує дуже простий спосіб повторного оголошення змісту. Наприклад, якщо у вашій книзі не більше 9 розділів, а в деяких розділах більше 9 підрозділів, ви можете переформулювати пункт змісту підрозділу таким чином:

$$\backslash\text{DeclareTOCEntry}\{2\}\{\}\{9.99\}\{9.99\}.$$

Ця команда містить такі параметри: рівень пункту, дію, яка застосовується перед пунктом (наприклад, можна вставити вертикальний пропуск), префікс перед номером пункту (наприклад, знак розділу), прототип числа для обчислення відступу, а також стиль, який застосовується до пункту змісту. Шостий необов'язковий параметр у квадратних дужках використовується для розрахунку пропуску полів для пункту наступного рівня. Якщо він опущений, то пропуск дорівнює значенню відступу для цього пункту.

### 2.1.4 Команди персоналізації

На закінчення ми покажемо команди персоналізації з їхніми значеннями за замовчуванням для пакета **ncclatex**:

– `\SectionTagSuffix{\hskip .6em}` – це суфікс, що вставляється після тега номера розділу (за винятком розділів 0-го рівня). Числа в розділах у **NCC- $\text{\LaTeX}$**  закінчуються десятковою крапкою;

– `\CaptionTagSuffix{\hspace .6em plus .2em minus .1em}` – це суфікс, що вставляється після тега номера підпису;

– `\NumberlineSuffix{\hspace .6em}{\hspace .4em}` описує 2 суфікси: перший використовується під час розрахунку відступів змісту, а останній вставляють після цифри в абзацах змісту. Останній суфікс зазвичай вужчий, ніж перший. Це дозволяє зменшити відстань між номером і текстом, якщо ширина номера дещо перевищує ширину прототипу. Примітка: команда `\numberline` зі стандартного L<sup>A</sup>T<sub>E</sub>X модифікована в пакеті **nccsect**, щоб запобігти накладенню тексту на число, якщо число ширше, ніж значення відступу;

– `\PnumPrototype{99}` – це прототип номера сторінки, що використовується для розрахунку правого поля в змісті. За замовчуванням передбачається, що книга містить не більше 99 сторінок. Щоб змінити ширину номера сторінки на 3-значний прототип, використовуйте команду `\PnumPrototype{999}`.

## 2.2 Теореми в N<sub>C</sub>C-L<sup>A</sup>T<sub>E</sub>X

Пакет **ncclatex** визначає низку математичних операторів, використовуючи пакет **nccthm** з пакета *ncctools*. Для пояснення можливостей пакета **nccthm** необхідний короткий вступ. У цьому розділі для простоти ми називаємо всі математичні твердження теоремами.

### 2.2.1 Огляд

У математичних манускриптах корисні два способи нумерації теорем. У звичайному режимі номер іде після назви теореми (наприклад, **Theorem 1**, **Example 1**, **Theorem 2**, тощо). У цьому режимі теореми з різними назвами зазвичай нумеруються незалежно одна від одної. Інший режим вимагає наявності номера перед назвою теореми. Ми називаємо цей режим стилем *apar*. У стилі *apar* теореми з різними назвами нумеруються послідовно (наприклад, **1 Theorem**, **2 Example**, **3 Theorem**, тощо). Обидва режими можна змішувати в одному манускрипті.

Теореми з різними заголовками можуть бути показані по-різному. Наприклад, твердження «Теорема» показується з напівжирним заголовком і курсивним тілом (comment після заголовка оформляється звичайним шрифтом). А твердження «Приклад» може бути показано з курсивним заголовком і звичайним шрифтом. Подання нових теорем ми називаємо *типом теореми*. Пакет **nccthm** надає два типи теорем за замовчуванням – *theorem* і *remark*. Можна створювати нові типи теорем і перевизначати наявні.

Наступним параметром теорем є стиль відступів. Теорема може відкривати новий абзац без відступів (поведінка за замовчуванням) і з абзацними відступами. Номер теореми в стилі *apar* може відображатися на полях. Стиль відступів регулюється опціями *indent*, *noindent*, *margin* і *nomargin* пакета **nccthm**.



Останнім параметром теорем є стиль розриву. Переведення рядка може бути використано після заголовка теореми (включно з коментарем). Стиль розриву і відсутності розриву може бути закодований у теоремах під час їхнього визначення за допомогою перемикачів `\TheoremBreakStyle` і `\TheoremNoBreakStyle` перед визначенням або перевизначенням теорем (на відміну від стандартного  $\text{\LaTeX}$ , для перевизначення теорем передбачена команда `\renewtheorem`). Стиль розриву окремої теореми може бути змінений на льоту за допомогою команд `\breakafterheader` і `\nobreakafterheader`, вставлених перед теоремою.

Усі описані властивості теорем забезпечуються пакетом **nccthm**. Більше того, лічильники, що використовуються в теоремах, оголошені як динамічні лічильники (див. пакет **dcouter** з *ncctools*). Динамічний лічильник створюється під час першого використання. Таким чином, простір лічильних регістрів  $\text{\TeX}$  не зайнятий марними лічильниками (це дуже важливо, оскільки кількість лічильників обмежена). Використовуючи динамічні лічильники, автор пакета може без проблем підготувати кілька зумовлених теорем. Оскільки динамічний лічильник створюється при першому використанні, користувач може контролювати стиль нумерації для нього. Для цього використовується команда `\countstyle{базовий лічильник}`. Базовий лічильник (*base-counter*) – це ім'я лічильника, якому має бути підпорядкований новостворений динамічний лічильник. Порожній параметр означає відсутність підпорядкування. Таким чином, зміна стилю нумерації для всіх теорем у манускрипті виконується всього однією командою! Якщо деякі лічильники повинні мати певний стиль рахунку, ви можете створити їх вручну або за допомогою команди `\countstyle` з необов'язковим параметром

$$\backslash\countstyle[\text{list-of-counters}]\{\text{base-counter}\}.$$

Ця команда створює нові лічильники зі списку, розділеного комами, і підпорядковує всі лічильники в списку базовому лічильнику (*base-counter*). Її можна використовувати і для наявних лічильників, якщо ви хочете *змінити* їхню підпорядкованість іншому лічильнику. Наприклад, використовуючи

$$\backslash\countstyle[\text{section}]\{\}$$

у книзі ми відкидаємо підпорядкування лічильника підрозділів лічильнику розділів.

### 2.2.2 Попередньо визначені теореми

Пакет **ncclatex** визначає *вісім* теорем у *трьох* формах кожна: теореми з автоматичною нумерацією, теореми з ручною нумерацією і теореми в стилі *apar*. Усі теореми можна використовувати двома способами: як структури і як команди. У командному стилі вони повинні завершуватися іншою теоремою або командою `\qef`, або `\qed`. Команда `\qef` завершує абзац, вставляє

відповідний вертикальний пробіл і повертає форму шрифту до звичайної. Команда `\qed` зазвичай завершує доведення теореми. Вона друкує символ `qed` (білий квадрат), вирівняний на праве поле, і викликає команду `\qef`. Команда `\qed*` друкує тільки символ `qed`.

### 2.2.3 Теореми з автоматичною нумерацією

Ми показуємо використання цих теорем у стилі оточення.

```
\begin{theorem}[Comment]    body    \end{theorem},
\begin{lemma}[Comment]      body    \end{lemma},
\begin{proposition}[Comment] body    \end{proposition},
\begin{corollary}[Comment]   body    \end{corollary},
\begin{definition}[Comment]  body    \end{definition},
\begin{statement}[Comment]   body    \end{statement},
\begin{example}[Comment]     body    \end{example},
\begin{remark}[Comment]      body    \end{remark}.
```

У структурному стилі кінець теореми еквівалентний команді `\qef`. Імена лічильників збігаються з іменами структур. Перші 4 структури готуються з використанням типу `theorem`, а останні 4 структури – з використанням типу `remark`.

### 2.2.4 Теореми з ручною нумерацією

Ми показуємо використання цих теорем у стилі оточення `also`.

```
\begin{Theorem}{Number}[Comment]    body    \end{Theorem},
\begin{Lemma}{Number}[Comment]      body    \end{Lemma},
\begin{Proposition}{Number}[Comment] body    \end{Proposition},
\begin{Corollary}{Number}[Comment]   body    \end{Corollary},
\begin{Definition}{Number}[Comment]  body    \end{Definition},
\begin{Statement}{Number}[Comment]   body    \end{Statement},
\begin{Example}{Number}[Comment]     body    \end{Example},
\begin{Remark}{Number}[Comment]      body    \end{Remark}.
```

Якщо параметр *Number* порожній, то друкується твердження без номера. Ці команди взагалі не використовують лічильники. Тип оформлення цих тверджень аналогічний попередньому.

Кожен тип теореми має відповідну команду `\like type`, що використовується для підготовки теорем цього типу. Якщо необхідно оформити нестандартне твердження як теорему, то можна скористатися командою `\liketheorem`:

```
\liketheorem{Title}{Number}[Comment] body \qef.
```

Для оформлення твердження як зауваження, є команда `\likeremark`.

```
\likeremark{ Title}{Number}[Comment] body \qef.
```

### 2.2.5 Теореми в стилі **apar**

Ми показуємо застосування цих теорем у командному стилі.

```
\atheorem[Comment]    body    \qef,
\alemma[Comment]      body    \qef,
\aproposition[Comment] body    \qef,
\acorollary[Comment]  body    \qef,
\adefinition[Comment] body    \qef,
\astatement[Comment]  body    \qef,
\anexample[Comment]   body    \qef,
\aremark[Comment]     body    \qef.
```

Усі ці твердження нумеруються за допомогою динамічного лічильника **apar**. Типи подання для цих теорем такі самі, як і для відповідних теорем з автоматичною нумерацією. Зіркоподібні форми команд `\liketheorem` і `\likeremark` використовуються для підготовки варіантів математичних тверджень. Вони не мають параметра номера:

```
\liketheorem*{Title}[Comment] body \qef,
\likeremark*{Title}[Comment] body \qef.
```

Є ще команда `\apar`, що дає змогу почати новий нумерований параграф із заданим заголовком або без нього:

```
\apar[Header].
```

Він починає новий абзац, пронумерований наступним значенням лічильника **apar**. Заголовок виділено жирним шрифтом.

### 2.2.6 Команда доведень і підтримка різних **qed** символів

Оформлення доведень виконується за допомогою команди `\proof[Comment]`. Завершується доведення зазвичай командою `\qed`. Наприклад, команда

```
\proof[теореми 1] Деякий текст \qed
```

виглядає наступним чином: **Доведення теореми 1.** Деякий текст.

Поряд з `\qed`, команда `\qedsymbol` може бути використана в тому випадку, коли доведення завершується виключною формулою. Вона набирається за допомогою команди  $\tag*{\qedsymbol}$  всередині виключної формули.

Пакет **nccthm** тепер дає два варіанти символу **qed** – білий і чорний. Вони запитуються через опції пакета `whiteqed` і `blackqed` відповідно. Коли використовується опція, для неї створюються ще дві команди: `\option` і `\optionsymbol`. Наприклад, опція `whiteqed` генерує команди `\whiteqed` і `\whiteqedsymbol`, а опція `blackqed` генерує команди `\blackqed` і `\blackqedsymbol`. Остання опція в списку опцій описує символ **qed** за

замовчуванням. Якщо опції не використовуються, генеруються тільки команди `\qed` і `\qedsymbol`. Інші `qed` символи також вітаються.

### 2.2.7 Команди персоналізації

На закінчення ми покажемо команди персоналізації з їхніми значеннями за замовчуванням для пакета **ncclatex**.

`\TheoremCommentDelimiters{}}` описує роздільники, що вставляються до і після коментаря теореми. У пакеті **nccthm** використовуються круглі дужки, а в пакеті **ncclatex** роздільники порожні.

`\AfterTheoremHeaderChar{.}` описує символ, що вставляється наприкінці заголовка теореми і доведення.

`\AfterTheoremHeaderSkip{\hskip .7em plus .2em minus .1em}` описує горизонтальний пропуск, вставлений після заголовка теореми і доведення.

`\AparStyleParameters{\bfseries}{\bfseries}{.\enskip}` описує параметри стилю для підготовки `apar` теореми: перший параметр – стиль, який використовується в команді `\apar`, другий і третій параметри описують префікс і суфікс, які вставляють до та після команди `\theapar` під час верстки.

`\ProofStyleParameters{\bfseries}{\proofname}` описує параметри стилю, які використовуються в команді `\proof`: перший – стиль шрифту, останній – заголовок доведення (його закодовано тут у команді `\proofname` з визначенням за замовчуванням `\newcommand\proofname{Proof}`).

### 2.3 Математичне розширення

У старому **NCC-L<sup>A</sup>T<sub>E</sub>X** для підготовки виключних формул використовувалися стандартні структури **L<sup>A</sup>T<sub>E</sub>X**, а для підтримки ручної нумерації були розроблені деякі команди розширення. У новому випуску **NCC-L<sup>A</sup>T<sub>E</sub>X** було вирішено використати чудове математичне розширення, що поставляється з пакетом **amsmath**, і дещо удосконалити його. Розширення пакета *AMS* представлено пакетом **nccmath**. Пакет **nccmath** передає всі опції пакета **amsmath**.

Було зроблено два великих розширення:

– структура **eqnarray** в **amsmath** залишена без змін, оскільки існують альтернативні середовища *AMS*. Ми перевизначили `eqnarray` для роботи в стилі *AMS*. У ньому було зроблено такі поліпшення: тег рівняння готується так само, як у виключних формулах *AMS* (дозволено `\tag` і `\tag*`); дозволено команду `\displaybreak`; міжстовпчикова відстань зменшена до відстані між звичайними та реляційними математичними символами; центральне поле підготовлено в `\displaystyle`;

– у старому стилі **NCC-L<sup>A</sup>T<sub>E</sub>X** був еквівалент структури `array`, що називався `darray`, який призначався для оформлення масиву формул у стилі `\displaystyle`. У новій реалізації ця структура теж є, але вона повністю

перероблена і виконана незалежною від `array` (для сумісності з іншими пакетами, в яких структура `array` перевизначається). Нова імплементація дещо відрізняється від старої: у параметрі специфікації стовпчиків у структурі `darray` можна використовувати тільки команди `l`, `c`, `r`, `@`, `*`; міжстовпчикова відстань зменшена та дорівнює величині відступу, характерного для символів операцій порівняння; величини відступів перед першим і після останнього стовпчика за замовчуванням дорівнюють нулю.

### 2.3.1 Еквіваленти виключних формул у $\text{\LaTeX}$

Наступні еквіваленти  $\text{\LaTeX}$  надаються разом із **nccmath**:

```
\eq{formula} = \begin{equation} formula \end{equation},
\eq*{formula} = \begin{equation*} formula \end{equation*},
\eqs{formulas} = \begin{eqnarray} formulas \end{eqnarray},
\eqs*{formulas} = \begin{eqnarray*} formulas \end{eqnarray*},
\eqalign{formulas} = \begin{equation} \begin{darray}{rcl} formulas
\end{darray} \end{equation},
\eqalign*{formulas} = \begin{equation*} \begin{darray}{rcl} formulas
\end{darray} \end{equation*}.
```

Команди `\eqs` і `\eqs*` мають додатковий параметр, що керує відстанню між стовпцями масиву формул. Наприклад, команда

```
\eqs[0mm]{&& -\Delta u = f, \quad && u|_{\Gamma} = 0,}
```

обнуляє міжстовпцеву відстань, оскільки два перші стовпці в цій формулі не використовуються. Структура `eqnarray` не має додаткового параметра.

Команди `\eqalign` і `\eqalign*` також мають додатковий параметр. Він означає специфікацію стовпців у формулі:

```
\eqalign{formulas} = \eqalign[rcl]{formulas}.
```

### 2.3.2 Зауваження щодо виключних рівнянь

Стиль вирівнювання у виключних рівняннях (крім низькорівневого `$$` у  $\text{\TeX}$ ) можна змінити за допомогою таких структур:

```
\begin{fleqn}[margin] body \end{fleqn},
\begin{ceqn} body \end{ceqn}.
```

Структура `fleqn` встановлює вирівнювання по лівому краю з лівим відступом, заданим необов'язковим параметром (за замовчуванням відступ дорівнює `0pt`). Структура `ceqn` задає вирівнювання по центру.

Команда `\intertext` з **amsmath** тепер має необов'язковий параметр,

```
\intertext[skip]{текст між формулами}.
```

Пропуск – це значення вертикального пропуску, що вставляється до і після тексту. Якщо він опущений, задаються стандартні пробіли.

У  $\text{\LaTeX}$  не рекомендується починати виключну формулу з нового абзацу (іншими словами, у вертикальному режимі). Це пов'язано з тим, що алгоритм  $\text{\TeX}$  вставляє порожній абзац перед такою формулою. Щоб уникнути вставки порожнього абзацу, Дональд Кнут рекомендував використовувати команду  $\text{\backslash noindent}$  перед формулою відображення, що починається у вертикальному режимі. Цей трюк використовується в пакеті **nccmath**. Усі середовища відображення дещо коригуються, якщо в їхній початок вставити команду  $\text{\backslash NCC@ignorepar}$ . Ця команда робить більше, ніж просто застосовує цей трюк. Вона забезпечує заборону вертикального пробілу перед виключною формулою, якщо вона відкриває міні-сторінку. Крім того підтримує використання короткого пропуску над виключною структурою за запитом користувача.

У  $\text{\TeX}$  використовуються два типи пропусків над виключними формулами: звичайний пропуск, визначений у реєстрі  $\text{\backslash abovedisplayskip}$ , і короткий пропуск, визначений у реєстрі  $\text{\backslash abovedisplayshortskip}$ . Коли набирається виключна формула,  $\text{\TeX}$  вирішує, який пропуск вставити у залежності від ширини формули, її стилю (відцентрована або вирівняна за лівим краєм, нумерована зліва чи справа) і ширини решти тексту в останньому рядку попереднього абзацу. Але цей алгоритм працює тільки для звичайних формул. Він не працює в багаторядкових формулах, підготовлених за допомогою команди  $\text{\backslash halign}$ . Тому в деяких випадках потрібна ручна заміна звичайного пропуску на короткий. Для цього в пакет **nccmath** введено команду  $\text{\backslash useshortskip}$ . Вона змушує використовувати короткий пропуск у наступній за ним виключній формулі.

### 2.3.3 Інші математичні команди

Наступні математичні команди перейшли зі старого макросу  $\text{\NCC-L\TeX}$ .

$\text{\backslash nr}$  має точно такий самий синтаксис, як команда  $\text{\backslash}$ , але вставляє додаткову відстань у  $0.5ex$ . Ця команда часто використовується в масивах виключних формул для додаткового розсування деяких рядків.

$\text{\backslash mrel}\{рядки\}$  створює новий математичний символ із категорії символів відношення за допомогою одностовпцевої таблиці.

Наприклад,

$$\text{\backslash mrel}\{<\backslash [-.7ex]>\}$$

виводить  $\leq$ .

$\text{\backslash underrel}\{base\}\{bottom\}$  служить двійником до команди  $\text{\backslash overrel}$ .

Наприклад,

$$\text{\backslash A}\text{\backslash underrel}\{\text{\backslash longrightarrow}\}\{x\text{ to }0\}\text{\backslash B}\text{\backslash}$$
 виводить  $A \xrightarrow{x \rightarrow 0} B$ .

## 2.4 Рисунки та таблиці в $\text{\LaTeX}$

У стандартних класах  $\text{\LaTeX}$  плаваюче оформлення рисунків і таблиць виконується в структурах `figure` і `table`. Ці структури забезпечують плаваюче розміщення, але не задають стиль оформлення. У макросі  $\text{\LaTeX}$  стиль оформлення задається спеціальними засобами в основному за допомогою надбудови додаткових команд над командами плаваючого оформлення. Ці команди впроваджені в пакет **nccfloats** з пакета **ncctools**.

### 2.4.1 Базові команди

Команда `\FloatingStyle{style}` визначає стиль оформлення малюнків і таблиць. Він впливає на матеріал, підготовлений за допомогою команд, описаних нижче. За замовчуванням у макросі  $\text{\LaTeX}$  встановлено стиль

`\FloatingStyle{\footnotesize\centering}`.

Базові команди `\minifig` і `\minitabl` забезпечують підготовку матеріалу у вигляді міні-сторінки і дозволяють використання команди `\caption` всередині. Синтаксис цих команд відповідає розширеному синтаксису команди `\parbox`:

`\minifig[pos][height][inner-pos]{width}{body},`  
`\minitabl[pos][height][inner-pos]{width}{body}.`

Параметр *pos* задає вертикальне вирівнювання (t, b або c) щодо навколишнього тексту; параметр *height* задає необхідну висоту міні-сторінки; параметр *inner-pos* визначає вертикальне вирівнювання вмісту всередині міні-сторінки (t, b, c або s); нарешті, параметр *width* задає ширину міні-сторінки. Тіло плаваючого елемента оформляють відповідно до заданого стилю, воно може містити команду `\caption` всередині.

Решта команд плаваючих розширень  $\text{\LaTeX}$  засновані на `\minifig` и `\minitabl`.

### 2.4.2 Оформлення з обтіканням

Для невеликих малюнків і таблиць краще вставляти їх усередину тексту, а не використовувати плаваючий механізм. За правилами української поліграфії в режимі обтікання рисунок має розташовуватися на зовнішньому боці сторінки (зліва – на парній і праворуч – на непарній сторінці за двостороннього друку). За одностороннього друку рисунки і таблиці мають займати правий бік аркуша паперу. Таке розміщення забезпечується такими командами:

`\sidefig[pos](w1)(w2){figure body}{text body},`  
`\sidefig*[pos](w1)(w2){figure body}{text body},`  
`\sidetabl[pos](w1)(w2){table body}{text body},`  
`\sidetabl*[pos](w1)(w2){table body}{text body}.`

Для простоти надалі ми використовуємо термін *minifloat* для невеликого ілюстративного матеріалу (рисунок або таблиця), розуміючи, однак, що це зовсім не float. Він вставляється в основний потік поруч із рамкою абзацу, зазначеною в останньому параметрі описаних вище команд.

Команди без зірочки розміщують minifloat поруч із зазначеним текстом на зовнішній стороні сторінки (праворуч для непарної сторінки та ліворуч для парної). У двоколонковому або одноколонковому режимі minifloat завжди розташовується праворуч. Форми із зірочками забезпечують зворотне розташування. За замовчуванням minifloat центрується по вертикалі відносно тексту, а команду `\strut` вставляють на початку та наприкінці *text body*, щоб забезпечити нормальний інтерліньяж із рядками тексту, що обтікають таку команду зверху та знизу.

Усі параметри в квадратних і круглих дужках необов'язкові. Їхнє призначення:

- *pos* визначає вертикальне вирівнювання плаваючого елемента щодо тексту (t, b або c; за замовчуванням c). Може також містити додаткові літери: j означає розтяжку останнього рядка тексту (використовується, якщо параграф продовжується поза елементом з обтіканням), n скасовує вставку розпірок (вони повинні вставлятися вручну, якщо необхідно);

- *w1* задає ширину плаваючого елемента (число) в міліметрах (в одиницях `\unitlength`, які в макросі `NCS` дорівнюють міліметрам);

- *w2* задає ширину текстового боксу в міліметрах.

Якщо обидва параметри ширини опущені, то ширини тексту і плаваючого елемента дорівнюватимуть  $(\text{\code{\linewidth}}-1.5\text{em})/2$ . Якщо *w2* відсутній, то ширина текстового боксу обчислюється як  $\text{\code{\linewidth}}-w1-1.5\text{em}$ .

Під час підготовки плаваючих елементів з обтіканням іноді потрібно знати, де саме опиниться плаваючий елемент, – зліва чи справа, і виконувати дещо різне оформлення в різних випадках. Команда

$$\text{\code{\ifleftsidefloat}}\{left\text{-}clause\}\{right\text{-}clause\}$$

дає таку інформацію. Це корисно: параметри `\sidefig` або `\sidetabl` обробляють лівий пункт, якщо інше речення minifloat розміщено ліворуч або праворуч.

### 2.4.3 Плаваюче оформлення рисунка або таблиці

$$\begin{aligned} &\text{\code{\fig}}[placement](w)\{body\}, \\ &\text{\code{\fig*}}[placement](w)\{body\}, \\ &\text{\code{\tabl}}[placement](w)\{body\}, \\ &\text{\code{\tabl*}}[placement](w)\{body\}. \end{aligned}$$

*placement* – це параметр розміщення плаваючого елемента, який описує місце, де може з'являтися плаваючий елемент. Значенням за замовчуванням є `htp` (тут, або вгорі наступної сторінки, або на сторінці тільки з плаваючими числами). Необов'язковий параметр (*w*) визначає ширину прямокутника, що



займає `float` (ширина вкладеного) `\minifig` або `\minitabl`). Якщо його опустити, то максимальна ширина `float` дорівнює `\linewidth`. Команди `\fig` і `\tabl` включають середовища фігури та таблиці відповідно. Їхнє місце розташування огортають відповідні навколишні середовища.

Команди `\fig` і `\tabl` суть надбудови над структурами `figure` і `table`. Їхні `*`-форми є надбудовами над `*`-формами цих структур, тобто використовуються під час оформлення плаваючих елементів, що займають всю ширину сторінки у двохстовпцевій моді.

#### 2.4.4 Плаваюче оформлення пари рисунків або таблиць

```
\figs[placement](w1)(w2){body1}{body2},
\figs*[placement](w1)(w2){body1}{body2},
\tabs[placement](w1)(w2){body1}{body2},
\tabs*[placement](w1)(w2){body1}{body2}.
```

Ці команди розміщують дві фігури або таблиці поруч. *Body1* є тілом лівої фігури або таблиці, а *body2* – тілом правої фігури або таблиці. Інші параметри не є обов'язковими. Параметр *placement* має той самий сенс, що й вище, а параметри (*w1*) і (*w2*) задають ширину лівого і правого плаваючого елементів. Якщо обидва параметри опущені, ліве і праве поля матимуть ширину, яка дорівнює  $(\text{\linewidth}-1\text{em})/2$ . Якщо параметр (*w2*) опущено, праве поле займатиме залишок горизонтального простору мінус `1em`. Якщо вказані обидва параметри, простір, що залишився, буде вставлено між комірками. Якщо загальна ширина лівого і правого плаваючого елементів перевищує `\linewidth`, вони будуть перекриватися в середині рядка (між ними буде вставлено від'ємне значення горизонтального простору).

У команді `\tabs` поля лівого і правого тіла вирівнюються зверху, а в команді `\figs` використовується нижнє вирівнювання.

#### 2.4.5 Графіка в $\text{\LaTeX}$

Пакет **nccpic** із *ncctools* реалізовано на базі стандартного пакета **graphicx**.

Основною метою пакета **nccpic** є підготовка списку графічних розширень залежно від `dvi`-драйвера, що використовується з пакетом **graphicx**. Це дає змогу не вказувати розширення файлу в команді `\includegraphics`. Коли знаходиться файл без розширення, ця команда послідовно перебирає розширення зі списку, поки не буде знайдений відповідний файл.

Використовуючи цю можливість, ви можете підтримувати множинний висновок з  $\text{\LaTeX}$  з мінімальними змінами у вихідних кодах `.tex`. Єдине, що потрібно, це підготувати кілька версій для всіх графічних файлів, що викликаються в документі. Наприклад, програмі `dvips` подобаються файли `.eps`, а програмі попереднього перегляду `Yap` – файли `.bmp` або `.eps`, `pdflatex` –

файли .png. Щоб задовольнити їхні потреби, ви можете підготувати .eps, версії .bmp і .png для всіх зображень і передати необхідний параметр пакета **nccpic**.

Список графічних розширень тепер адаптовано для більшості популярних драйверів, а саме dvips, dvipdf, і dvipdfm, а також для використання з pdftex. Більше персоналізації вітається!

Наступна мета пакета **nccpic** – регулювання розміщення графічних файлів у файловій системі. Занадто незручно, коли рисунки зберігаються разом із вихідними файлами .tex. Ми пропонуємо зберігати графічні файли в підкаталозі graphics каталогу вихідних кодів. Для підтримки пошуку в цьому сховищі шлях до графіки налаштовується в пакеті **nccpic**.

У пакеті **nccpic** представлено такі команди:

- `\ipic{filename}` завантажує файл з іменем *filename.pic* з піддиректорії graphics. У старій версії макроса **NCC** .pic файли або містили картинки, підготовлені в **TEXCADi**, або команду `\putimage` вставки графічного об'єкта;

- `\putimage(x, y)[x_r, y_r](x_s, y_s){filename}` емулює команду вставки графіків у стилі старого макроса **NCC**. Більш повний опис команди знаходиться в пакеті **nccpic**;

- `\draftgraphics`, `\finalgraphics` перемикають режим виведення графічних об'єктів. У режимі `\draftgraphics` замість графічного об'єкта малюється прямокутник потрібних розмірів, а в режимі `\finalgraphics` графічний файл вставляється в текст документа. Зазначимо, що опції `draft` і `final` команди `\documentclass` включають режими `\draftgraphics` і `\finalgraphics` відповідно. Саме тому останні команди було додано в макрос **NCC**, щоб можна було гнучкіше керувати режимом виведення графічних об'єктів. Зазначимо також, що в драфтовому режимі команда `\putimage` не перевіряє, чи існує вказаний графічний файл чи ні.

*Примітка: Якщо Ви плануєте конвертувати документ у формат pdf і хочете намалювати малюнок у **TEXCADi**, то не використовуйте режим `emlines` у ньому, оскільки транслятор **pdflatex** не розуміє графіки **emTEX**.*

## 2.5 Додаткові бокси

Макрос **NCC** завантажує пакет **nccboxes** із комплекту *ncctools*, що визначає додаткові бокси:

- `\jhbox{прототип}[align]{текст}` розміщує *текст* у боксі, ширину якого задано параметром *прототип*. Додатковий параметр *align* задає вирівнювання (l, c, r або s; за замовчуванням c);

- `\jvbox{прототип}[align]{текст}` зсуває текст по вертикалі, використовуючи для вирівнювання «підпорку», визначену вертикальними розмірами *прототипу*. Додатковий параметр *align* задає вертикальне вирівнювання відносно підпорки (t, c або b; за замовчуванням c): t означає

вирівнювання висоти (верху) тексту і підпорки, *b* – вирівнювання глибини (низу) тексту і підпорки, *c* – центрування тексту відносно підпорки. Результируюча висота і глибина боксу обчислюються як максимум між відповідними параметрами підпорки і вирівняного по ній тексту;

– `\jparbox{прототип}[align]{ширина}{тіло}` формує параграфний бокс потрібної *ширини* та вирівнює його вертикально відносно *прототипу* у такий самий спосіб, як і в команді `\jvbox`;

– `\rbox[align]{тіло}` реалізує просту одностовпцеву таблицю без обведення і відступів. Параметр *align* може складатися з двох літер, що визначають горизонтальне вирівнювання тексту всередині стовпця (*l*, *c* або *r*) і вертикальне вирівнювання таблиці щодо навколишнього тексту (*t*, *c* або *b*). Вирівнювання по центру встановлено за замовчуванням. Наприклад, команда `\rbox[l]{...}` еквівалентна команді

$$\backslash begin\{tabular\} [t]{@{}l@{}}...\backslash end\{tabular\}$$

– з одним маленьким винятком, що відстань між рядками в таблиці `\rbox` не залежить від значення `\arraystretch`;

– `\addbox{зверху}{знизу}{текст}` створює бокс, висота якого збільшена на величину, задану параметром *зверху*, глибина якого збільшена на величину *знизу*. Наприклад, `\addbox{.5ex}{.5ex}{text}` збільшує висоту і глибину боксу на *0.5ex*

$$\backslash picbox\{body\} = \backslash begin\{picture\}(0,0)(0,0) \textit{body} \backslash end\{picture\}.$$

Для підготовки «красивих» таблиць можна використовувати такі команди:

– `\Strut/число/` задає підпорку, висота і глибина якої обчислюються на основі прототипу (літери *A*) наступним чином: якщо *число* >0, то висота підпорки збільшується на величину інтерліньяжу (відстані між базовими лініями сусідніх рядків тексту), помножену на це число; якщо *число* <0, то глибина підпорки збільшується на величину інтерліньяжу, помножену на модуль цього числа. Наприклад, `\Strut/1/` вставляє підпорку, висота якої перевищує висоту літери *A* на величину одного рядка. Команда `\Strut` без параметра еквівалентна команді `\Strut/0/`;

– `\tstrut`, `\bstrut`, `\tbstrut` вставляють підпорки, висота і глибина яких перевищує розміри літери *A* на спеціальну величину. Цю величину обчислюють так, щоб повна висота (висота + глибина) підпорки `\tbstrut` була в 1.5 більшою за величину інтерліньяжу;

– `\cbox/число/[align]{тіло}` створює бокс, чиє тіло є одностовпцевою таблицею. Його висота і глибина збільшуються за допомогою `\tstrut` на початку і `\bstrut` в кінці тіла. Горизонтальне вирівнювання (*l*, *c* або *r*) усередині стовпця і вертикальне вирівнювання щодо підпорки (*t*, *c* або *b*) задаються параметром *align*. За замовчуванням використовується центроване вирівнювання. Результируючий бокс

вертикально вирівнюється відносно підпорки `\Strut/число/` за допомогою команди `\jvbox`. Команда `\cbox*` додатково обнуляє висоту і глибину результуючого боксу (використовується для оформлення мултистовпчикових комірок таблиць).

Команда `\cbox` застосовується в «заголовку» таблиць. Спільне виведення тіла команди визначається командою `\cboxstyle`. У макросі  $\mathcal{NCS}$  ця команда перевизначена таким чином:

```
\renewcommand\cboxstyle{\scriptsize}.
```

Продемонструємо використання підпорок і команди `\cbox` на прикладі таблиці (табл. 2) яка була зроблена таким чином:

```
\begin{center}
\renewcommand\cboxstyle{\small\bf}
\setlength{\tabcolsep}{10pt}
\begin{tabular}{|l|c|c|c|}\hline
\cbox*/-1.5/{Vertically\spanned head} & \cbox{Simple head}
&\multicolumn{2}{c|}{\cbox{Very long head\of two
lines}}\ccline{2-4}
&\cbox{subhead} &\cbox{subhead} &\cbox{subhead}\hline
\Strut/1/ Text & field & field & field \\\
Text & field & field & field \\\
\bstrut Text & field & field & field \\\hline
\end{tabular}
\end{center}
```

Таблиця 2 – Використання підпорок і команди `\cbox`

| Вертикально<br>розміщений<br>заголовок | Простий<br>заголовок | Дуже довгий заголовок із двох ліній |              |
|----------------------------------------|----------------------|-------------------------------------|--------------|
|                                        | підзаголовок         | підзаголовок                        | підзаголовок |
| Text                                   | field                | field                               | field        |
| Text                                   | field                | field                               | field        |
| Text                                   | field                | field                               | field        |

*Примітка:* також можна використовувати команди `\tbox`, `\bbox`, `\tbbox` зі старого макросу  $\mathcal{NCS}$ . Однак вони вважаються застарілими і тому рекомендується замість них використовувати команди `\strut`, `\strut`, `\strut`.

## 2.6 Різне

Опишемо команди, що задаються самим пакетом **ncclatex**.

`\acknow` починає новий параграф із заголовком *Acknowledgements* або *Подяки* (у українському стилі).

`\cref{мітка}` виводить посилання на номер бібліографічного елемента без квадратних дужок. Використовується при підготовці посилань, що містять складні перерахунки, наприклад, посилань виду «[2-4, 7]».

`\mor{name}` використовується для кодування нестандартних математичних операцій типу  $\sin$ . Наприклад, `$k=\mor{sgn}x$` виводить  $k = \operatorname{sgn} x$  із правильним вибором шрифту математичної операції та відступів. Ця команда еквівалентна команді  $\textit{AMS} \backslash \operatorname{operatorname}$  з пакета **amsmath**.

`\tg`, `\ctg`, `\arctg` і `\arcsctg` виконують математичні операції для тангенса, котангенса та їхніх обернених функцій у українській друкарській традиції.

`\No` виводить символ №. Якщо опція `russian` не використовується, то символ номера емулюється текстом «No.».

`\tl{текст}`, `\tc{текст}`, `\tr{текст}` вимкнення тексту вліво, по центру і вправо за допомогою пружинок `\hspace*{\fill}`. Ці команди використовуються в таблицях, коли вирівнювання в комірці має відрізнятися від вирівнювання в усьому стовпці.

Назви доведень, зумовлених теорем і підтверджень оголошуються командами із суфіксом `name`. Наприклад, заголовок доведення задається командою `\proofname`, заголовок леми – командою `\lemmaname`, а заголовок підтвердження – командою `\acknowname`. Усі вони можуть бути перевизначені для забезпечення інтернаціоналізації (опція `ukrainian` перевизначає ці заголовки українською мовою).

### 3 Клас **sibjnm**

Цей клас описує стиль, який використовується в журналі *Siberian Journal of Numerical Mathematics (SibJNM)*, <http://www.sccc.ru/SibJNM/>, e-mail: [sibjnm@oapmg.sccc.ru](mailto:sibjnm@oapmg.sccc.ru). Рекомендовано використовувати цей клас для подання статті в цей журнал.

Клас **sibjnm** заснований на класі **ncc**. Він встановлює такі параметри класу **ncc**: `a4paper`, `11pt`, `article`, `twoside` і `onecolumn`. Користувач може керувати такими параметрами: `draft`, `final`, `openany`, `openright` і `russian`. Інші варіанти заборонені.

**SibJNM** – двомовний журнал. Стаття може бути підготовлена англійською мовою або на кирилиці. Анотація подається обома мовами. Перед анотаціями необхідно використовувати одну з команд предметної класифікації: `\UDC{indices}` або `\AmSclassification{indices}`. Перша команда встановлює класифікаційні індекси UDC, а остання – класифікаційні індекси  $\textit{AMS}$  теми статті.

Кириломовний реферат має йти відразу після команди `\maketitle`:

`\begin{Rabstract}{Authors}{Title} body \end{Rabstract}`.

Реферат на англійській повинен йти після кириломовного:

`\begin{Eabstract}{Authors}{Title} body \end{Eabstract}`.

Текстове поле статті **SibJNM** – 155×225 мм.

## Результати досліджень

Аналіз доступної літератури показав наступне.

1. Формально кирилізація  $\text{\LaTeX}$  нині існує. Варіантів кирилізації багато, але майже всі вони не повні, тобто завдання кирилізації не вирішено до кінця.

2. Наслідок п. 1 – усі запропоновані варіанти кирилізації не сумісні, тому що одержані додаванням різних макросів, тобто вирішують різні завдання різними способами. Інакше кажучи, реально кирилізації немає.

3. Найбільш повно та кардинально питання кирилізації  $\text{\LaTeX}$ -а вдалося вирішити О.Й. Роженку шляхом розробки та включення до дистрибутиву  $\text{\LaTeX}$ -а цілої системи макросів  $\text{\LaTeX}$ : *ncctools* – колекція загальних пакетів для стандартного  $\text{\LaTeX}$ : *fterpackage*, *dcounter*, *desclist*, *extdash*, *manyfoot*, *mboxfill*, *nccbbb*, *nccboxes*, *ncccomma*, *ncccropbox*, *ncccropmark*, *nccfancyhdr*, *nccfloats*, *nccfoots*, *\LaTeX*, *nccmath*, *nccparskip*, *nccpic*, *nccrules*, *nccsect*, *nccstretch*, *nccthm*, *ncctools*, *textarea*, *tocenter*, *topsection*, *watermark*.

По суті,  $\text{\LaTeX}$  – розширений клас загального призначення – надає платформу для заміни загального класу стандартних класів статей, книг та звітів, деякі з них також можуть використовуватися зі стандартними класами. Пакет *ncclatex* теж завантажує багато стандартних пакетів і вимагає останньої версії комплекту *ncctools*.

При використанні цих макросів практично всі питання щодо кирилізації знімаються автоматично.

4. Верстка в пакеті  $\text{\LaTeX}$  нічого не змінює у стандартному процесі верстки тексту класичним  $\text{\LaTeX}$ .

Прикладом може служити наступні лістинг та скріншот (рис. 1).

```
\documentclass[14pt]{ncc}
\ToCenter[h]{160mm}{250mm}
\usepackage[headings]{ncchdr}
%\usepackage{latexsym}
\usepackage[english, ukrainian]{babel}
\renewtheorem{theorem}{Теорема}
\renewtheorem{definition}{Визначення}
\renewtheorem{notation}{Примітка}

\begin{document}
\title{КОНСТРУКЦІЯ ПОВНИХ-ПРОСТИХ\
ДИСТРИБУТИВНИХ РЕШТОК (ГРАТОК)}
\author{Георгій~А. Менухін\thanks{Дослідження підтримано NSF під
номером гранту 23466.}\
Кафедра комп'ютерних наук\
Віннебаго, Мінесота 23714\
menuhin@cc.uwinnebago.edu}
```

\date{15 березня 1999 року}  
 \maketitle

\begin{abstract}

У цій замітці ми доведемо, що існують *\emph{повні-прості дистрибутивні решітки}*,

тобто повні дистрибутивні решітки,  
 в яких є лише дві повні конгруенції.

\end{abstract}

\section{Введення}\label{S:intro}

У цій записці ми доведемо такий результат:

\begin{theorem}

Існує нескінченна повна дистрибутивна решітка  $K$   
 лише з двома тривіальними повними співвідношеннями конгруенції.

\end{theorem}

...

# КОНСТРУКЦІЯ ПОВНИХ-ПРОСТИХ ДИСТРИБУТИВНИХ РЕШІТОК (ГРАТОК)

Георгій А. Менухін\*  
 Кафедра комп'ютерних наук  
 Віннебаго, Міннесота 23714  
 menuhin@cc.uwinnebago.edu

**Анотація.** У цій замітці ми доведемо, що існують *повні-прості дистрибутивні решітки*, тобто повні дистрибутивні решітки, в яких є лише дві повні конгруенції.

## 1. Введення

У цій записці ми доведемо такий результат:

**Теорема 1.** *Існує нескінченна повна дистрибутивна решітка  $K$  лише з двома тривіальними повними співвідношеннями конгруенції.*

## Висновки

При необхідності набирати кириличний текст слід виконувати наступне.

1. Перш, ніж відправляти статтю до того чи іншого вітчизняного чи зарубіжного журналу, з'ясуйте, у якому конкретно (з якою кирилізацією) пакеті  $\text{\LaTeX}$  потрібно її оформляти. Якщо конкретна кирилізація видавництвами не обговорюється, ви готуєте статтю у тій, у якій звикли працювати; якщо кирилізація вказана, то запитайте її у видавництва разом із інструкцією користувача.

2. Під час використання кількох пакетів кирилізації переконайтеся, що вони сумісні.

3. При використанні  $\text{\LaTeX}$  практично всі питання щодо кирилізації знімаються автоматично.

Список літератури.

1. Rozhenko, A.I. CTAN Comprehensive TEX Archive Network. <https://ctan.org/author/rozhenko>.