



Операційні системи

Лекція 12

Файлові системи



План лекції

- Основні поняття про файли і файлові системи
- Імена файлів
- Типи файлів
- Логічна організація файлів
- Файлові операції
- Міжпроцесова взаємодія через файлову систему
- Загальна модель файлової системи
- Фізична організація файлів



Основні означення

- **Файл** – це набір даних, до якого можна звертатись за іменем
 - Файли є найпоширенішим засобом організації доступу до інформації, що зберігається в енергонезалежній пам'яті
- **Файлова система** – це підсистема ОС, яка призначена для того, щоби забезпечити користувачеві зручний інтерфейс для роботи з даними, що зберігаються в енергонезалежній пам'яті (на диску), і забезпечити спільне використання файлів кількома користувачами і процесами
 - Файлова система надає прикладним програмам абстракцію файлу
- До складу файлової системи входять:
 - сукупність усіх файлів;
 - набори структур даних, що застосовуються для керування файлами:
 - каталоги,
 - дескриптори,
 - таблиці розподілу дискового простору;
 - комплекс системних програмних засобів, що реалізують керування файлами, зокрема створення, видалення, зчитування, записування, іменування, пошук та інші операції



Імена файлів

- Для користувачів – символічні імена
 - Обмеження на алфавіт
 - Які символи припускаються (ASCII, Unicode)
 - Чутливість до регістра (myfile.txt vs MYFILE.TXT)
 - Наявність і формат розширення
 - У деяких системах розширення обов'язкове (система і прикладні програми розрізняють типи файлів за розширенням)
 - Обмеження довжини імені
 - DOS (FAT) – 8.3
 - UNIX System V – 14
 - Windows (NTFS) – 255
- Унікальне ім'я
 - У більшості систем – повний шлях до файлу + ім'я файлу
 - Система не обов'язково працює із символічними іменами
 - Наприклад, у UNIX за іменем файлу визначають його індексний дескриптор (i-node)



Типи файлів

- Звичайні файли
 - Текстові – рядки символів (деякі символи можуть мати спеціальне значення – кінець рядка, кінець файлу)
 - Бінарні – послідовність байт (біт)
 - Окремий тип – виконувані файли
- Файли-каталоги (**directory**)
 - З одного боку це група файлів
 - З іншого боку, це спеціальний файл, який містить інформацію про файли, що до нього входять
- Спеціальні файли
 - Файли пристроїв
 - Блок-орієнтовані пристрої – файли із прямим доступом
 - Байт-орієнтовані (символьні) пристрої – файли із послідовним доступом
 - Дозволяють замінити спеціалізовані операції введення-виведення на стандартні операції зчитування і записування у файл



Каталоги

- Каталог містить список файлів і встановлює відповідність між файлами та їхніми характеристиками (атрибутами)
 - Ім'я
 - Тип (бінарний/символьний, каталог, зв'язок, спеціальний)
 - Розмір файлу
 - Атрибути безпеки (власник, read only, hidden, system, temporary, атрибути доступу)
 - Часові атрибути (час створення, час останньої модифікації)
- Каталоги можуть містити
 - усю інформацію (MS-DOS)
 - лише посилання на таблиці (UNIX)
- Каталоги можуть утворювати ієрархічну структуру
 - дерево (MS-DOS)
 - мережа (UNIX)
- **Шлях до файлу** – перелік каталогів через які можна отримати доступ до файлу
 - Може бути абсолютним і відносним



Розділи

- *Розділ* (*partition*) – частина фізичного дискового простору, що призначена для розміщення на ній структури одної файлової системи і з логічної точки зору розглядається як єдине ціле
- Розділ – це логічний пристрій, що з погляду ОС функціонує як окремий диск
 - Може відповідати усьому фізичному диску
 - Найчастіше відповідає частині фізичного диску
- Кожний розділ може мати свою файлову систему і, можливо, використовуватись різними ОС
- Існують два підходи до реалізації взаємозв'язку розділів і структури каталогів файлової системи
 - Розділи із встановленими на них файловими системами об'єднуються (монтуються) в єдине дерево каталогів
 - Характерно для UNIX-подібних систем
 - Приховує від користувача структуру розділів
 - Кожний розділ є окремим логічним томом, що має власне позначення (C:, D: тощо), власний кореневий каталог, власне дерево каталогів



Зв'язки (або посилання)

- **Жорсткі зв'язки (hard links)**
 - Встановлення жорсткого зв'язку – це фактично надання файлу додаткового імені (у більшості випадків, воно знаходиться в іншому каталозі)
 - Усі жорсткі зв'язки посилаються на один і той самий файл з його атрибутами (повинна підтримуватись таблиця дескрипторів)
 - Усі жорсткі зв'язки рівноправні
 - Спроба видалення файлу приводить до видалення лише окремого його імені і зменшенні на одиницю кількості імен, сам файл видаляється лише після видалення останнього з імен
- **Символічні зв'язки (symbolic links)**
 - Символічний зв'язок – це спеціальний файл, який містить ім'я файлу, на який він вказує
 - Видалення зв'язку ніяк не впливає на файл
 - Видалення або переміщення файлу зробить зв'язок недійсним (відновлення файлу поновить зв'язок)



Логічна організація файлів

- Структура, з якою має справу програміст
- Елемент логічної структури файлу називають *логічним записом*
 - Логічний запис – це найменший елемент даних, яким може оперувати програміст під час обміну із зовнішнім пристроєм
 - Фізичний обмін з пристроєм, як правило, здійснюється більшими порціями (блоками)
- Логічні записи можуть бути
 - Фіксованої довжини
 - Змінної довжини
 - З використанням індексних таблиць
- Для ідентифікації логічні записи можуть мати спеціальне поле, яке називають *ключем*
- У сучасних файлових системах файли, як правило, мають найпростішу структуру у вигляді послідовності однобайтових записів
 - Довжина логічного запису фіксована і дорівнює 1 байту,
 - Ключ відсутній



Файлові операції (1)

- Відкриття файлу
 - Необхідне для того, щоби розпочати роботу із файлом
 - Зазвичай передбачає завантаження у пам'ять дескриптора файлу, який визначає його атрибути і місце розташування на диску
- Закриття файлу
 - Структуру, створену під час відкриття файлу, видаляють з пам'яті
 - Усі зміни, що не були збережені, записують на диск
- Створення файлу
 - На диску створюють файл нульової довжини
 - При цьому його автоматично відкривають
- Видалення файлу
 - Структури, що описують файл (індексний дескриптор, запис у каталозі) вилучають або помічають як недійсні
 - Дисковий простір, який займав файл, помічають як вільний (але, як правило, не очищають)
 - Як правило, ця операція недопустима для відкритих файлів



Файлові операції (2)

- Читання з файлу
 - Пересилання певної кількості байтів із файлу, починаючи з поточної позиції, у заздалегідь виділений буфер пам'яті режиму користувача
- Записування у файл
 - Здійснюють з поточної позиції
 - Якщо у цій позиції вже є записані дані, їх перезаписують
 - Дані пересилають із заздалегідь виділеного буфера
 - Операція може змінити розмір файлу
- Переміщення покажчика
 - Покажчик поточної позиції можна встановити на будь-яке місце всередині файлу, або безпосередньо за його кінець
- Отримання і призначення атрибутів файлу
 - Операції дозволяють зчитати усі або деякі атрибути файлу і встановити їх нові значення



Операції над каталогами

- Створення нового каталогу
 - Новий створений каталог, як правило, порожній (у деяких реалізаціях у нього одразу додають елементи “.” та “..”)
- Видалення каталогу
 - На рівні системного виклику ця операція дозволена лише для порожнього каталогу
- Відкриття і закриття каталогу
 - Каталог, подібно до звичайного файлу, має бути відкритим перед використанням і закритим після використання
 - Деякі операції, пов’язані з доступом до елементів каталогу, допустимі лише для відкритих каталогів
- Зчитування елементів каталогу
 - Зчитує один елемент каталогу і переміщує поточну позицію на один елемент
- Перехід у початок каталогу
 - Переміщує поточну позицію на початок каталогу



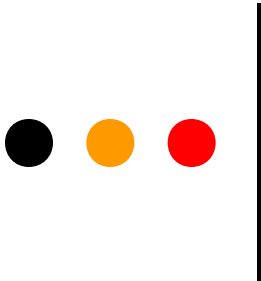
Відображення файлів у пам'ять

- Для відображення файлу у пам'ять застосовуються спеціальні системні виклики
 - У POSIX – `mmap()`, `munmap()`
 - Перед виконанням відображення файл має бути відкритим
 - У визначену частину адресного простору процесу відображають заданий файл або його частину
 - Після виконання цього виклику доступ до такої пам'яті спричинятиме прямий доступ до вмісту цього файлу
 - У разі закриття файлу або завершення процесу модифіковану інформацію зберігають у файлі на диску
- Кілька процесів можуть відобразити той самий файл у свої адресні простори
 - Таким чином можна реалізувати обмін даними між процесами
- Переваги відображення файлів у пам'ять
 - Лише один системний виклик
 - Прямий доступ до будь-якої ділянки (не потрібно переміщати покажчик у файлі)
 - Не потрібно копіювати дані між системною пам'яттю і буфером режиму користувача
- Недоліки
 - Не можна змінити розмір файлу!
 - Файл може бути більшим за обсяг віртуального адресного простору!



Міжпроцесова взаємодія через файлову систему

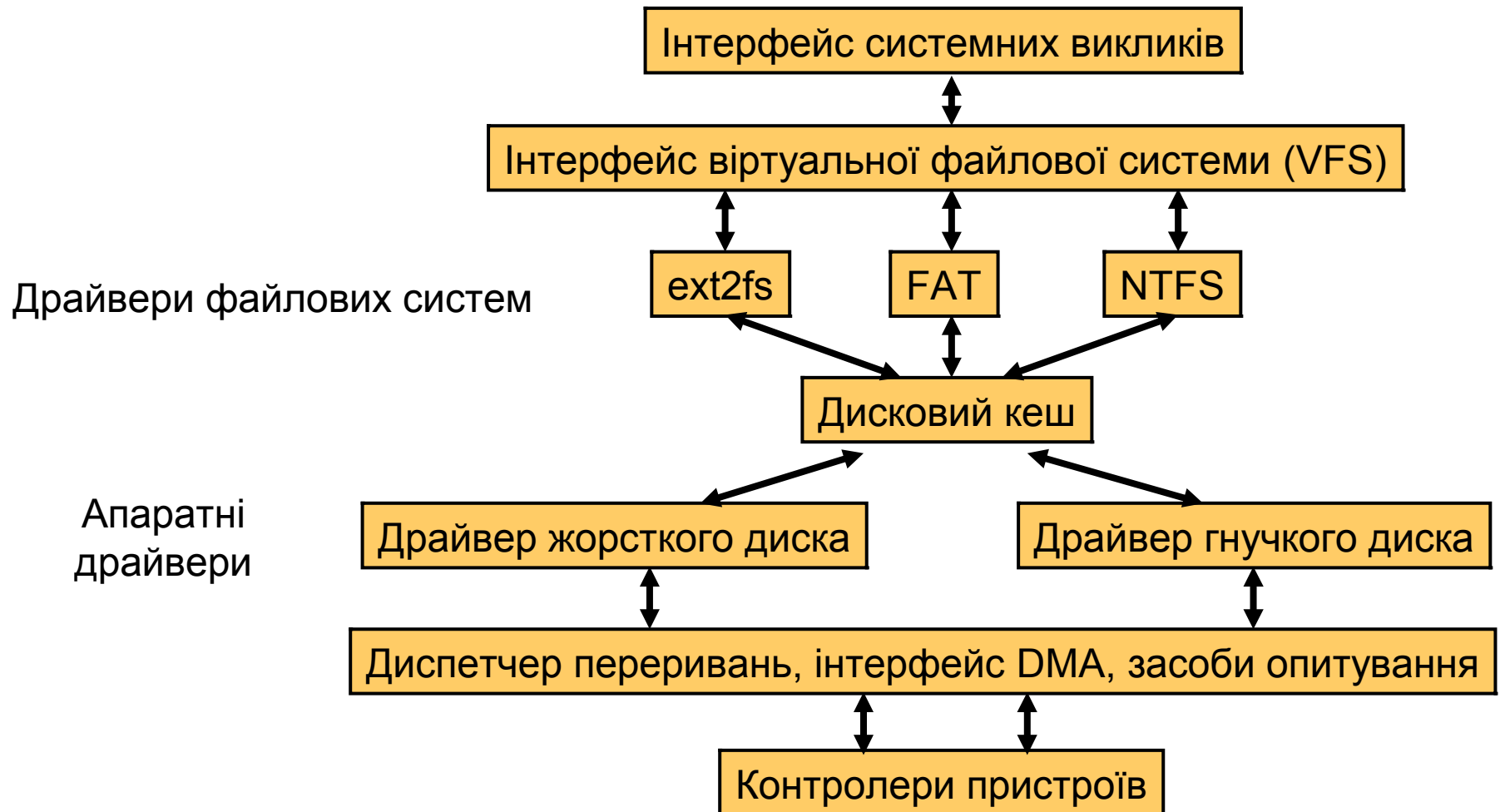
- Файлові блокування (file locks)
 - Засіб синхронізації процесів, які намагаються здійснити доступ до одного файлу
 - Консультативне, або кооперативне блокування (advisory lock)
 - Процес перед здійсненням операції перевіряє наявність блокування
 - У разі блокування – відмовляється від операції
 - За відсутності блокування – сам блокує
 - Якщо здійснює операцію без перевірки, то її дозволяють
 - Обов'язкове блокування (mandatory lock)
 - Здійснюється на рівні ядра
 - Небезпечно!
- Поіменовані канали
 - У POSIX – одnobічні FIFO канали
 - У Win32 – двобічний обмін повідомленнями



Загальна модель файлової системи

- Символьний рівень
 - За символьним іменем файлу визначається його унікальне ім'я
- Базовий рівень
 - За унікальним іменем файлу визначаються його характеристики (атрибути)
 - Перевіряються права доступу
 - При відкриванні файлу його атрибути переміщуються в оперативну пам'ять
- Логічний рівень
 - Визначаються координати логічного запису у файлі
- Фізичний рівень
 - Визначається номер блока

Багаторівнева структура



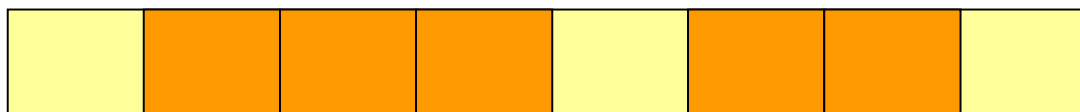


Фізична організація файлів

- Описує правила розміщення файлу на пристрої зовнішньої пам'яті
- Файл складається з фізичних одиниць – блоків
 - Блок – найменша одиниця, якою здійснюється обмін із зовнішнім пристроєм
 - У деяких файлових системах замість терміна “блок” вживають термін “кластер”
- Схеми фізичної організації
 - Неперервне розміщення файлів
 - Розміщення файлів зв'язними списками
 - Прості зв'язні списки
 - Зв'язні списки з таблицею розміщення файлів
 - Індексоване розміщення файлів

Неперервне розміщення

- Кожному файлу надають послідовність блоків диска, що утворюють єдину неперервну ділянку
- Переваги:
 - Простота та ефективність
 - Для знаходження будь-якого блоку достатньо задати лише перший блок
 - Дуже швидкий доступ
- Недоліки:
 - Під час створення файлу часто невідома його довжина, відповідно, невідомо, яку ділянку треба для нього зарезервувати
 - Велика зовнішня фрагментація
- Приклади реалізації:
 - HPFS (OS/2)
 - Файлова система компакт-дисків

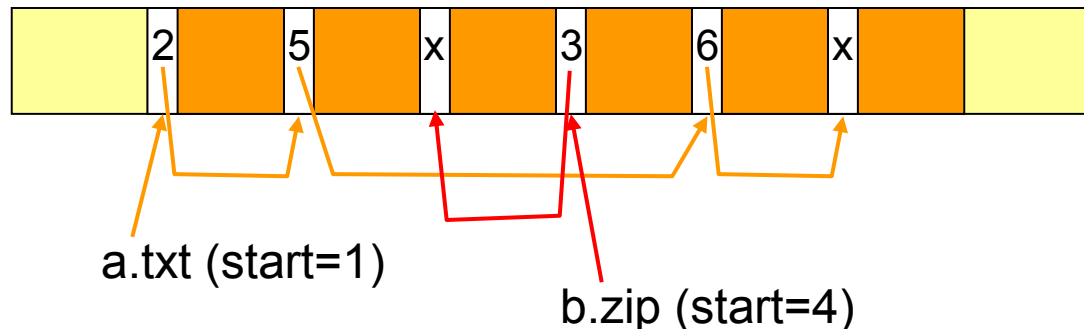


a.txt (start=1, len=3)

b.zip (start=5, len=2)

Прості зв'язні списки

- На початку кожного блока міститься показчик на наступний блок
- Переваги:
 - Достатньо задати початковий блок
 - Відсутня зовнішня фрагментація
 - Файл може змінювати довжину шляхом додавання блоків
- Недоліки:
 - Складність доступу до довільного блоку (необхідно прочитати усі попередні блоки)
 - Обсяг даних у блоці не дорівнює 2^n , оскільки частину блока виділяють для показчика на наступний блок





Зв'язні списки з таблицею розміщення файлів

- Усі посилання на номери блоків зберігають в окремій ділянці файлової системи, формуючи таблицю розміщення файлів (File Allocation Table, FAT)
- Кожний елемент такої таблиці відповідає блоку (кластеру) на диску, і може містити:
 - Номер наступного кластера
 - Індикатор кінця файлу
 - Ознаку вільного кластера
- Переваги:
 - Для доступу до довільного кластеру зчитують не попередні кластери, а лише елементи FAT
 - Розміри FAT дозволяють кешувати її у оперативній пам'яті
- Обмеження:
 - Маленькі кластери → багато елементів → завеликий обсяг FAT
 - Великі кластери → збільшення непродуктивних витрат дискового простору для малих файлів

Використання таблиці розміщення файлу

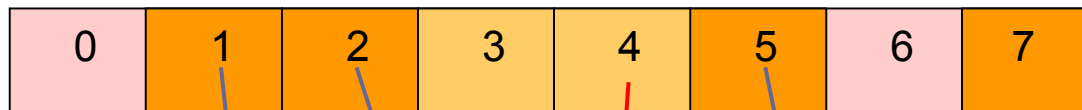
Каталог

a.txt	1
b.zip	4

FAT

0	Вільний
1	2
2	5
3	EOF
4	3
5	7
6	Вільний
7	EOF

Кластери на диску

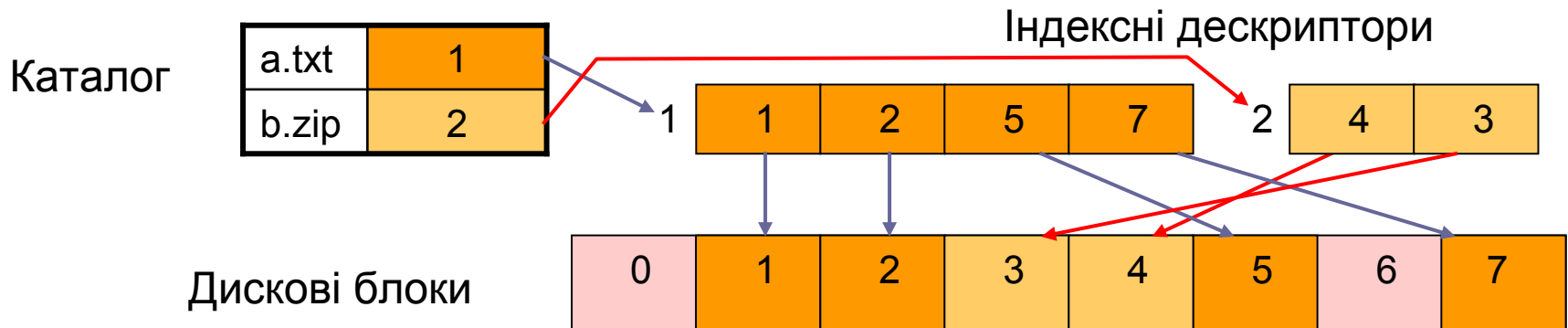


a.txt (start=1)

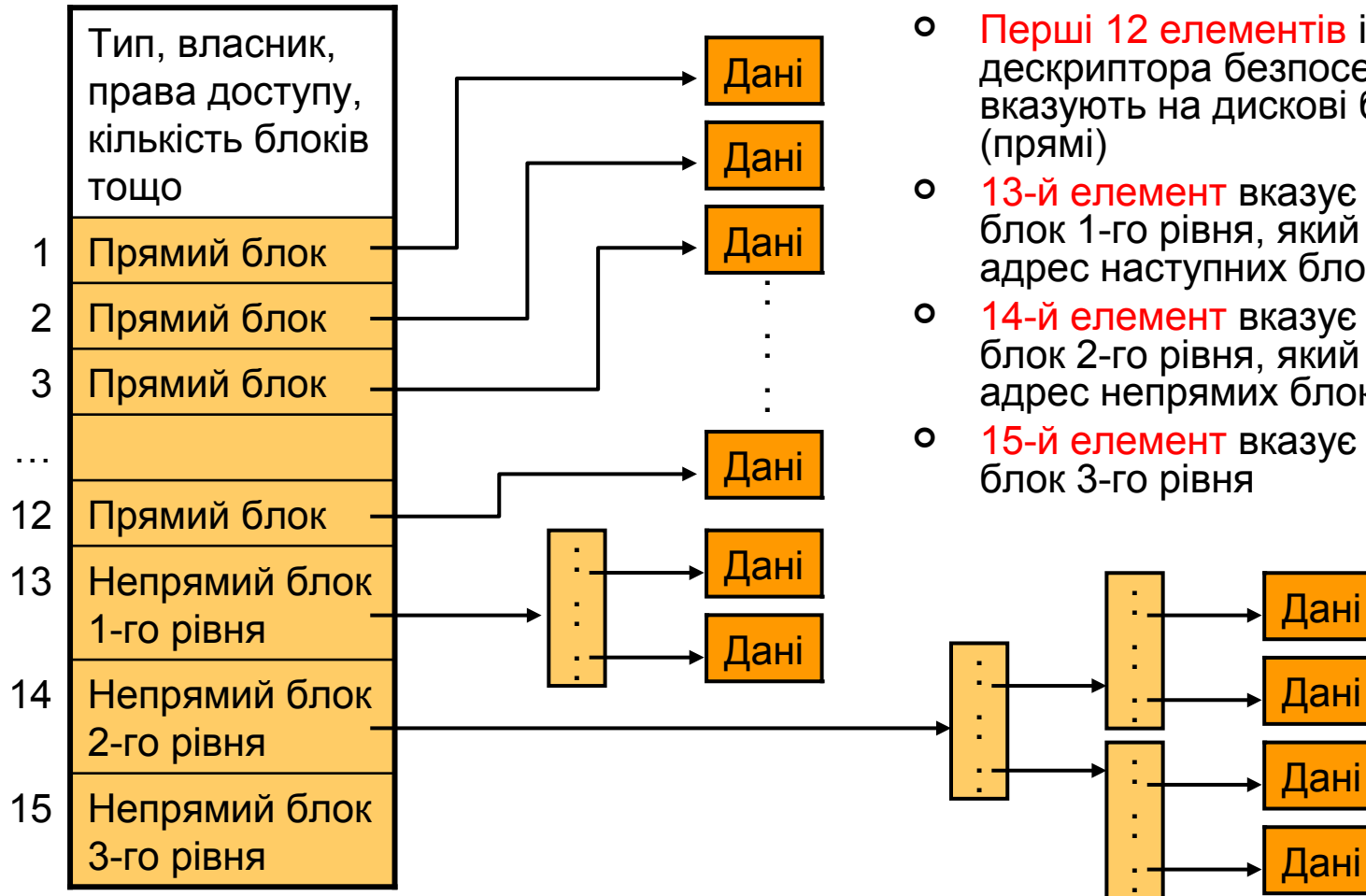
b.zip (start=4)

Індексоване розміщення

- У спеціальній структурі, пов'язаній з файлом (індексний дескриптор, i-node), безпосередньо описують розміщення усіх блоків файлу
 - У найпростішому варіанті індексний дескриптор – це масив, у якому перелічені адреси (номери) усіх блоків цього файлу
- Ефективно здійснюється як послідовний, так і випадковий доступ
 - Для підвищення ефективності індексний дескриптор повністю завантажують у пам'ять
- Основною проблемою є вибір розміру і структури індексного дескриптора



Приклад реалізації індексованого розміщення – файлові системи UNIX



- **Перші 12 елементів** індексного дескриптора безпосередньо вказують на дискові блоки з даними (прямі)
- **13-й елемент** вказує на непрямий блок 1-го рівня, який містить масив адрес наступних блоків файлу
- **14-й елемент** вказує на непрямий блок 2-го рівня, який містить масив адрес непрямих блоків 1-го рівня
- **15-й елемент** вказує на непрямий блок 3-го рівня