# Датотечни системи

## Оперативни системи 2024

проф. д-р Димитар Трајанов,
проф. д-р Невена Ацковска,
проф. д-р Боро Јакимовски,
проф д-р Весна Димитрова,
проф. д-р Игор Мишковски,
проф. д-р Сашо Граматиков,
вонр. проф. д-р Милош Јовановиќ,
вонр. проф. д-р Ристе Стојанов,
доц. д-р Костадин Мишев

Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ
И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

# Датотечни системи

- Што сè е потребно за долготрајно складирање на информација:
  - Мора да може да се складира **голема количина** на информација.
  - Информацијата мора **да постои и по завршувањето** на процесот кој ја користи.
  - Повеќе процеси мора да можат **истовремено да пристапуваат** до таа информација.

# Датотечни системи (2)

- Замислете го дискот како линеарна секвенца од блокови со фиксна големина, кој подржува читање и запишување на блокови.  Прашања:
  - Како да се најде саканата информација?
  - Како еден корисник да не може да пристапи до информацијата на друг корисник?
  - Како да се знае кои блокови се слободни?

# Датотеки

- Процесите (нишки), адресните простори и датотеките се најважните концепти во ОС
- Датотеките се логичка единица на информација креирана од процесите
  - Слично на адресните простори
- Датотечен систем
  - Управува со датотеки: како се структурирани, именувани, пристап, користење, заштита, имплементација, и др.

# Имиња на датотеки

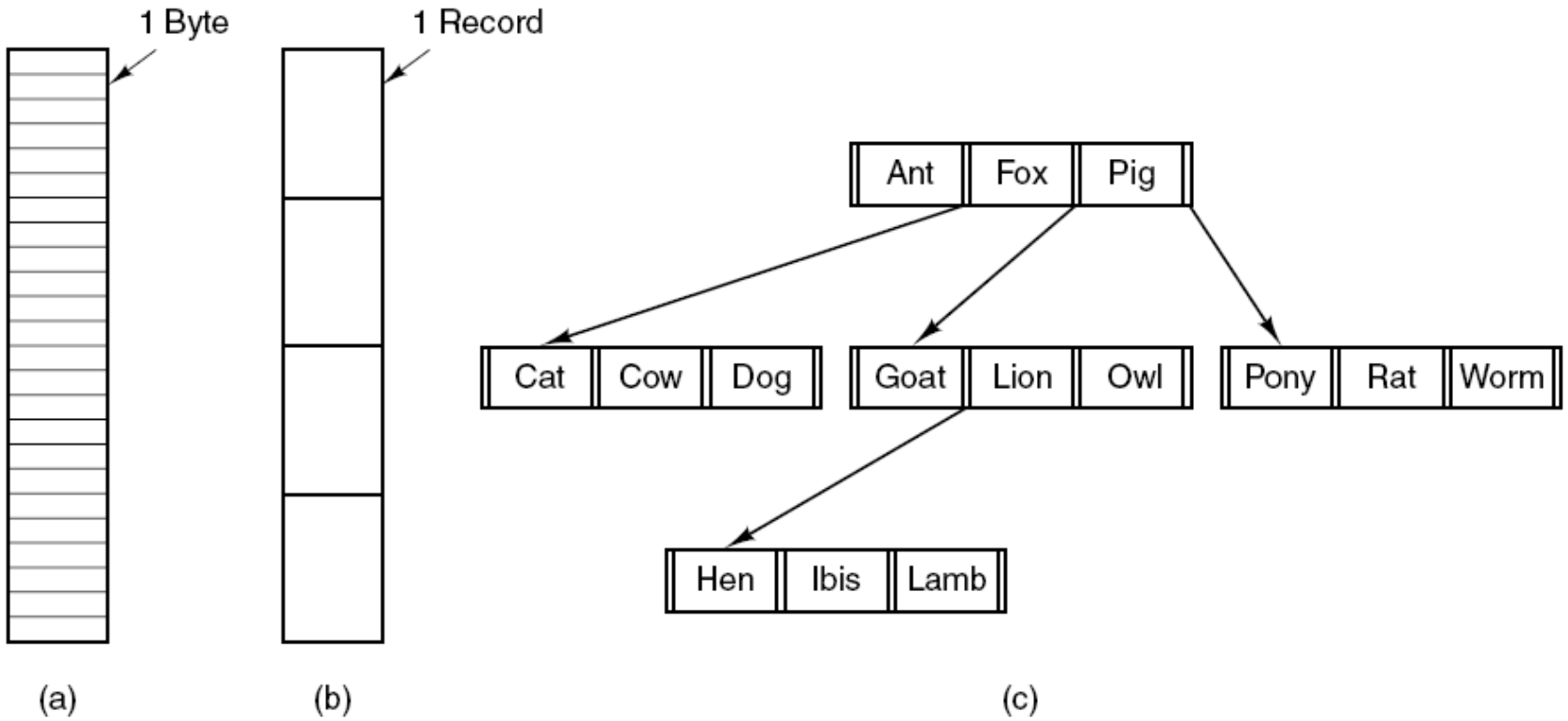
- Датотеките се апстрактен механизам
  - За да се зачува информација на дискот и потоа истата да се прочита
  - Кога еден процес ќе креира датотека, ја именува за да подоцна до истата може да се пристапи преку нејзиното име
- Имиња на датотеки од два дела
  - Екстензија: се наведуваат карактеристиките на датотеката
  - Во Unix, екстензијата е само конвенција; исклучок е C компајлерот
  - Во Windows, со екстензијата се означува која програма ја поседува датотеката од тој тип; со двоен клик се стартува соодветната програма

# Имиња на датотеки

| Extension | Meaning |
|---|---|
| file.bak | Backup file |
| file.c | C source program |
| file.gif | Compuserve Graphical Interchange Format image |
| file.hlp | Help file |
| file.html | World Wide Web HyperText Markup Language document |
| file.jpg | Still picture encoded with the JPEG standard |
| file.mp3 | Music encoded in MPEG layer 3 audio format |
| file.mpg | Movie encoded with the MPEG standard |
| file.o | Object file (compiler output, not yet linked) |
| file.pdf | Portable Document Format file |
| file.ps | PostScript file |
| file.tex | Input for the TEX formatting program |
| file.txt | General text file |
| file.zip | Compressed archive |

Слика 4-1. Некои типични екстензии.

# Структура на датотеки




Слика 4-2. Три типа на датотеки.

(a) Секвенца од бајти. (b) Секвенца од записи. (c) Дрво.

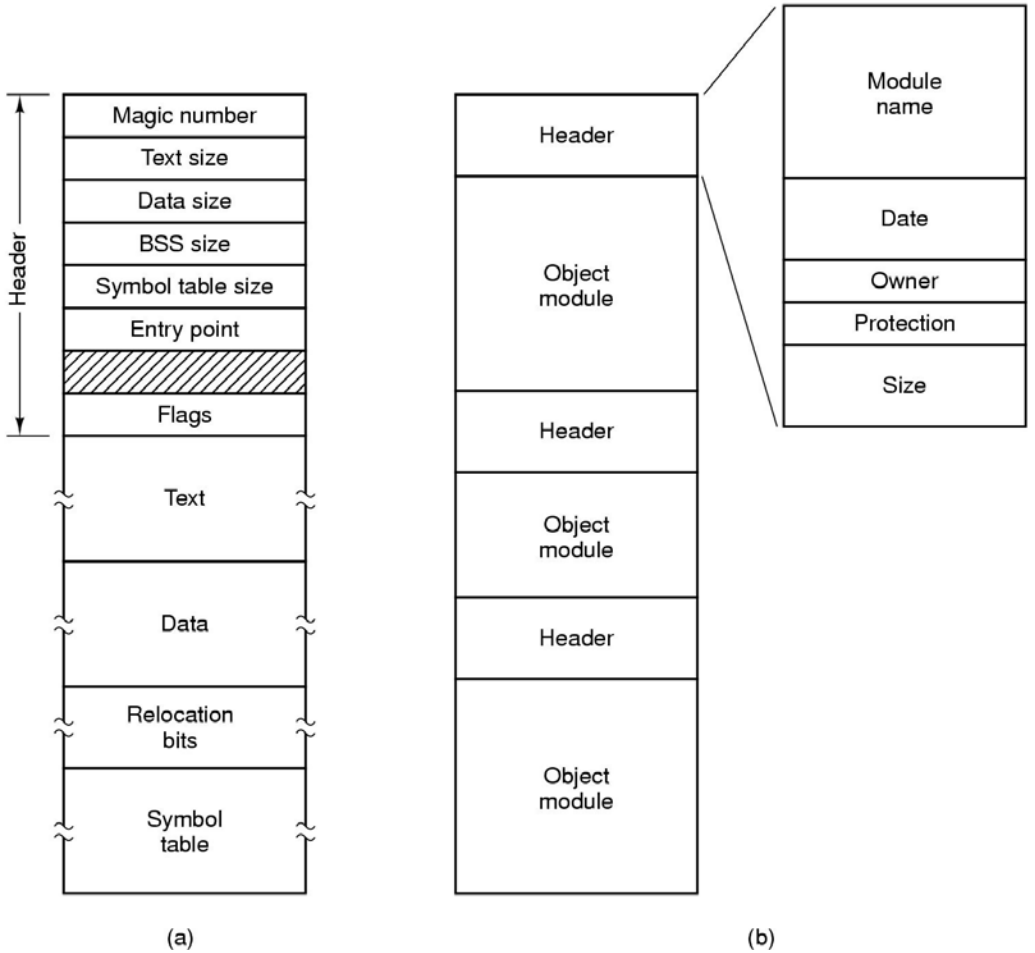
# Типови на датотеки (1)

- Регуларни:
  - ASCII или бинарни датотеки
  - ASCII содржат текст; кој може да се прикаже и печати
  - Бинарни, имаат интерна структура позната за програмите кои ги користат
- Именици
  - Датотеки во кои се организираат останати датотеки

# Типови на датотеки (2)

- Знаковни специјални датотеки (датотека за знаков уред – character device)
  - Поврзани со I/O и моделираат сериски I/O уреди
- Блоковски специјални датотеки (датотека за уред кој работи со блокови податоци – block device)
  - Се моделира диск

# Типови на регуларни датотеки




Слика 4-3. (a) Извршна датотека. (b) Архива.

# Пристап до датотеките (1)

- File descriptor
  - Тоа е small integer со кој се претставува некој објект управуван од јадрото, во/од кој процесот може да запишува/чита
  - Секој процес има приватен простор на file descriptors, почнувајќи од 0
  - Стандардно, 0 е стандарден влез, 1 е стандарден излез, и 2 е стандардна грешка
- Системски повици: read()  и write() читаат од и запишуваат во датотеките означени со file descriptors

# Пристап до датотеките (2)

- Секвенцијален пристап
  - Се читаат сите бајти/записи од почетокот па до крајот на датотеката
  - Не може да се скока на произволна локација
  - Се користи кај магнетни траки
- Случаен пристап
  - Се читаат бајти/записи по било каков редослед
  - Неопходно за базите на податоци
  - Тврди/SSD дискови

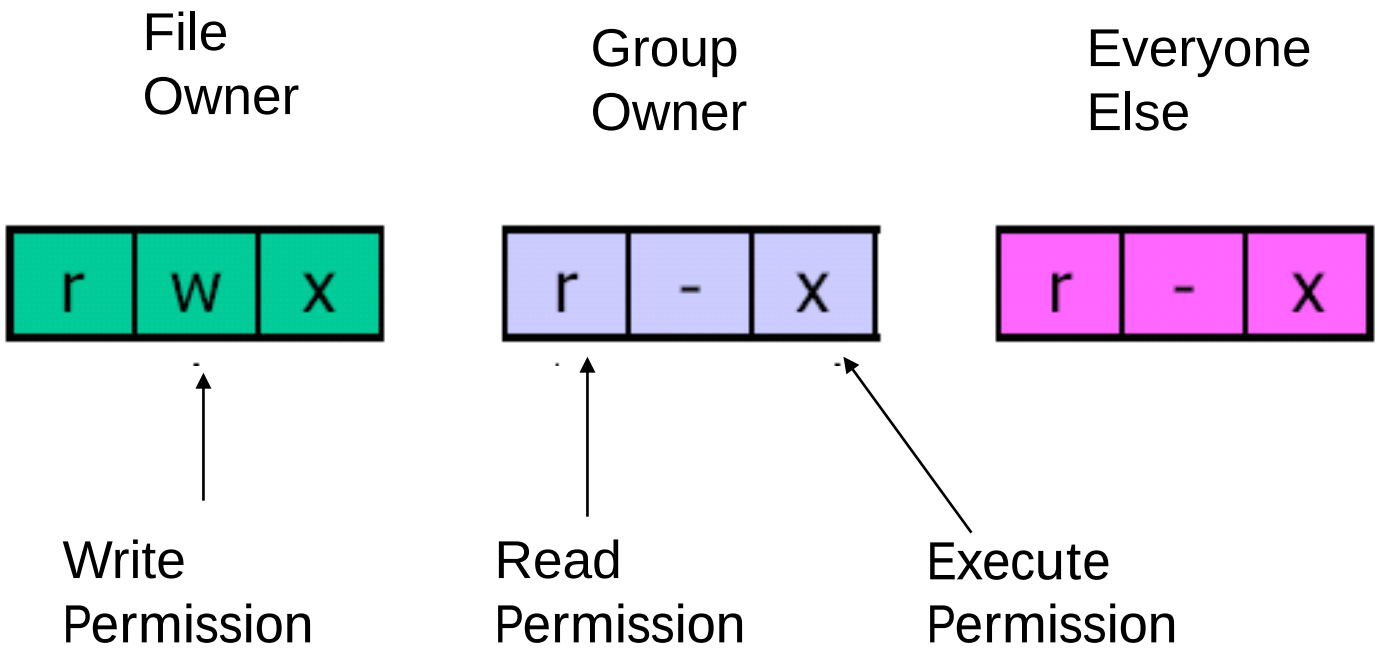
# Атрибути на датотеките (мета-податоци)

| Attribute | Meaning |
|---|---|
| Protection | Who can access the file and in what way |
| Password | Password needed to access the file |
| Creator | ID of the person who created the file |
| Owner | Current owner |
| Read-only flag | 0 for read/write; 1 for read only |
| Hidden flag | 0 for normal; 1 for do not display in listings |
| System flag | 0 for normal files; 1 for system file |
| Archive flag | 0 for has been backed up; 1 for needs to be backed up |
| ASCII/binary flag | 0 for ASCII file; 1 for binary file |
| Random access flag | 0 for sequential access only; 1 for random access |
| Temporary flag | 0 for normal; 1 for delete file on process exit |
| Lock flags | 0 for unlocked; nonzero for locked |
| Record length | Number of bytes in a record |
| Key position | Offset of the key within each record |
| Key length | Number of bytes in the key field |
| Creation time | Date and time the file was created |
| Time of last access | Date and time the file was last accessed |
| Time of last change | Date and time the file was last changed |
| Current size | Number of bytes in the file |
| Maximum size | Number of bytes the file may grow to |

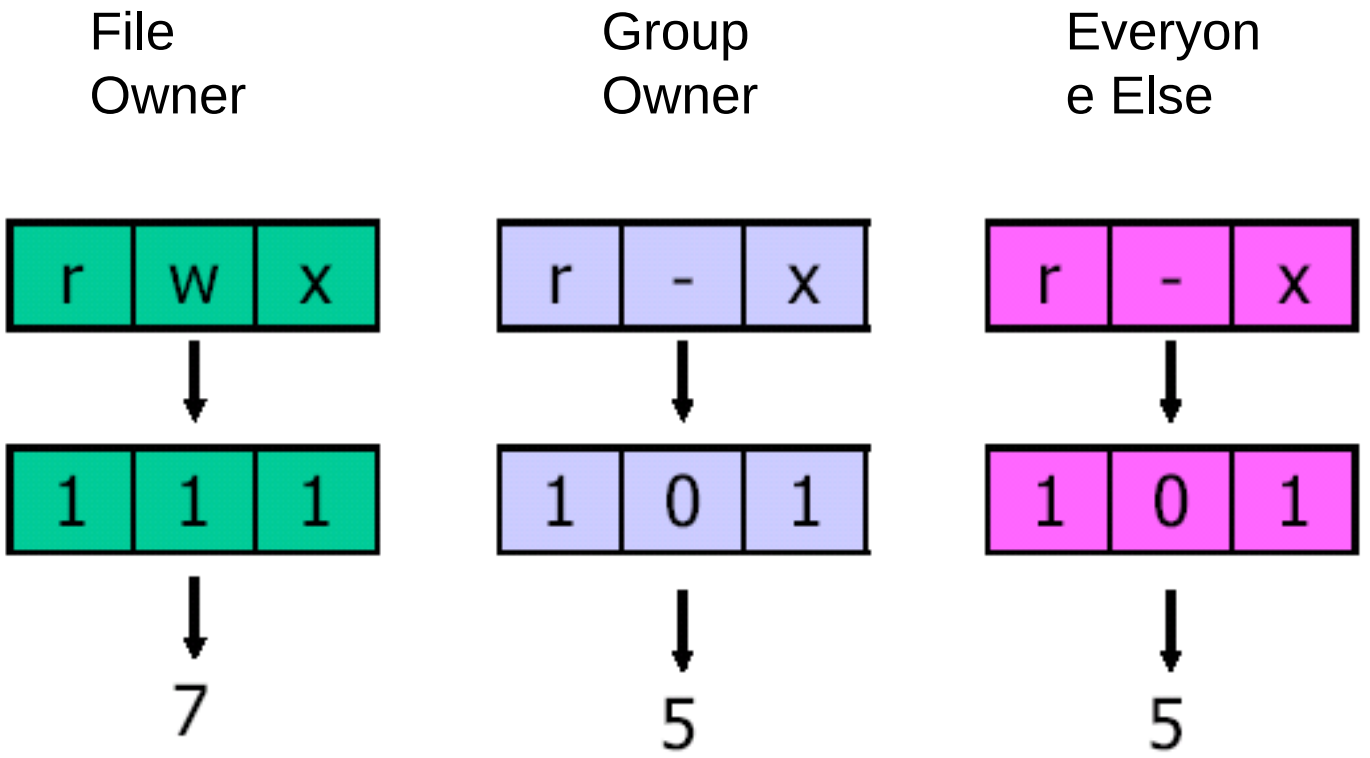
# Атрибути на датотеките (3)

```
drwxr-xr-x 2 root   root   4096 Sep 24  2008 Unit2
drwxr-xr-x 2 root   root   4096 May 26 19:21 a
-rwxr-xr-x 1 root   root  10930 Aug  5 22:49 a.out
-rwxrwx--T 1 root   root     81 Aug  2  2008 a.txt
-rwxr-x--- 1 root   root     81 May 26 19:20 b.txt
```

# Атрибути на датотеките (4)



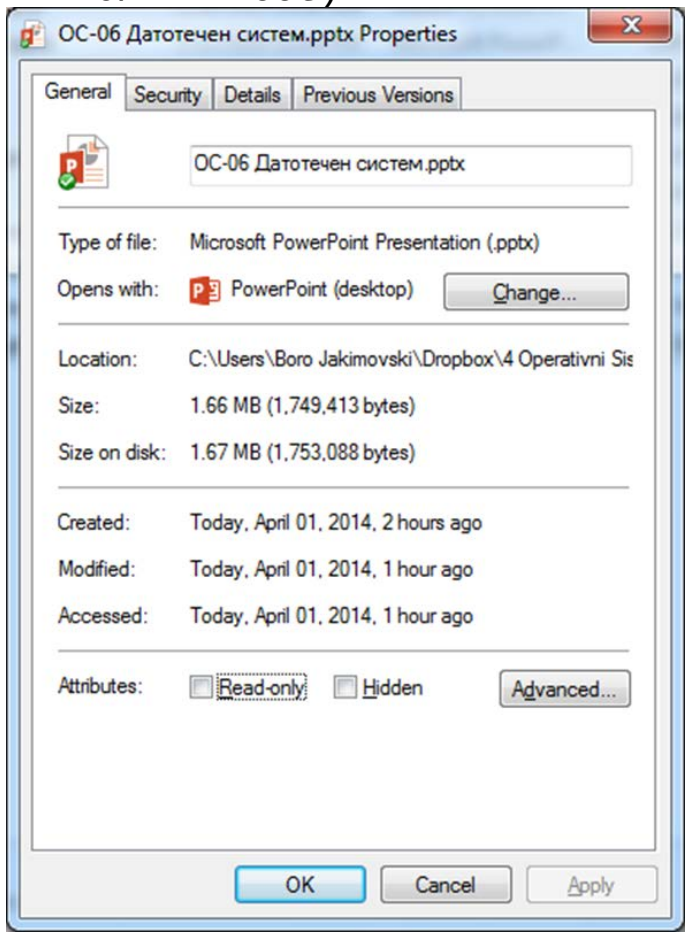
# Атрибути на датотеките (5)



# stat на Linux

```
$stat /etc/passwd
  File: `/etc/passwd'
  Size: 119417          Blocks: 248        IO Block: 4096   regular file
Device: 803h/2051d      Inode: 2696882     Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2014-04-01 11:04:17.000000000 +0200
Modify: 2014-03-30 09:52:09.000000000 +0200
Change: 2014-03-30 09:52:09.000000000 +0200
```

File properties на Windows



# Операции со датотеки (1)

- Create
- Delete
- Open
  - Се вчитуваат атрибутите и листата на адреси на дискот во главната меморија за да се забрза пристапот
- Close
- Read
- Write
- Append
  - Ограничена форма на write

# Операции со датотеки (2)

- Seek
  - За датотеки со случаен пристап
- Get Attributes
  - Се користи за програмата make во UNIX
- Set Attributes
  - Заштита на пристап на датотеките chmod
- Rename

# Пример за користење на системски повици за датотеки (1)

```
/* File copy program. Error checking and reporting is minimal. */

#include <sys/types.h>                  /* include necessary header files */
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]);       /* ANSI prototype */

#define BUF_SIZE 4096                   /* use a buffer size of 4096 bytes */
#define OUTPUT_MODE 0700                /* protection bits for output file */

int main(int argc, char *argv[])
{
    int in_fd, out_fd, rd_count, wt_count;
    char buffer[BUF_SIZE];

    if (argc != 3) exit(1);             /* syntax error if argc is not 3 */

    /* Open the input file and create the output file */
    in_fd = open(argv[1], O_RDONLY);    /* open the source file */
    if (in_fd < 0) exit(2);             /* if it cannot be opened, exit */
    out_fd = creat(argv[2], OUTPUT_MODE); /* create the destination file */
    if (out_fd < 0) exit(3);            /* if it cannot be created, exit */
```

Слика 4-5. Едноставна програма за копирање на датотека.

# Пример за користење на системски повици за датотеки (2)

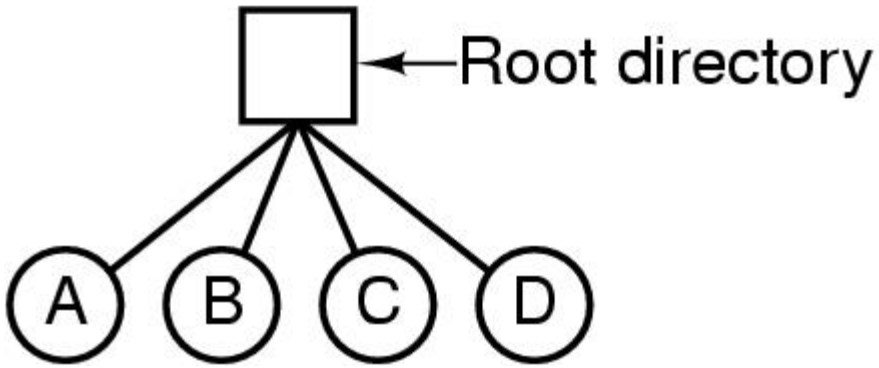
```
/* Copy loop */
while (TRUE) {
        rd_count = read(in_fd, buffer, BUF_SIZE); /* read a block of data */
if (rd_count <= 0) break;                         /* if end of file or error, exit loop */
        wt_count = write(out_fd, buffer, rd_count); /* write data */
        if (wt_count <= 0) exit(4);               /* wt_count <= 0 is an error */
}

/* Close the files */
close(in_fd);
close(out_fd);
if (rd_count == 0)                                /* no error on last read */
        exit(0);
else
        exit(5);                                  /* error on last read */
}
```

Слика 4-5. Едноставна програма за копирање на датотека.

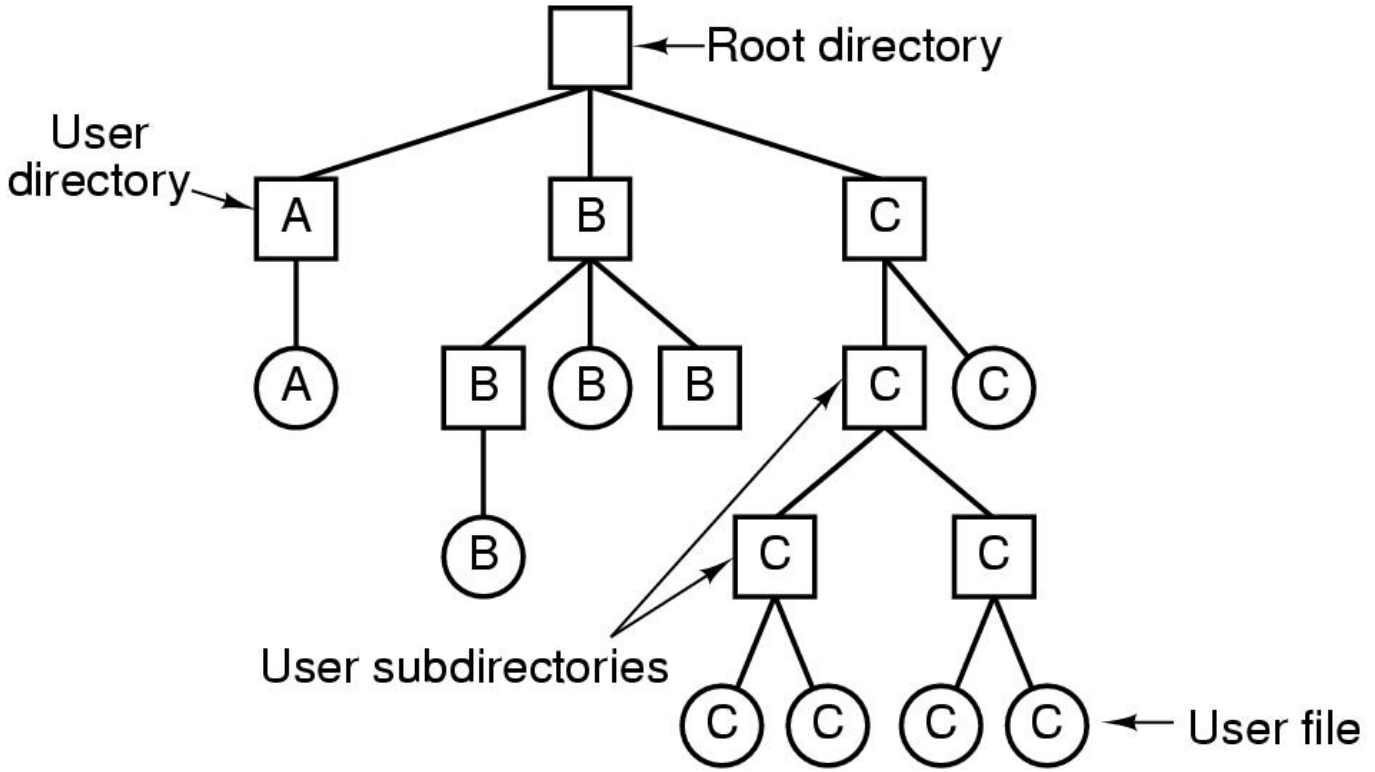
# Хиерархиски системи на именици (1)

- Едно-нивоовски систем на именици:
- Предност:
  - поедноставен софтверски дизајн
  - едноставно и брзо лоцирање на датотеки
- Недостатоци:
  - не смее да се користат датотеки со исти имиња

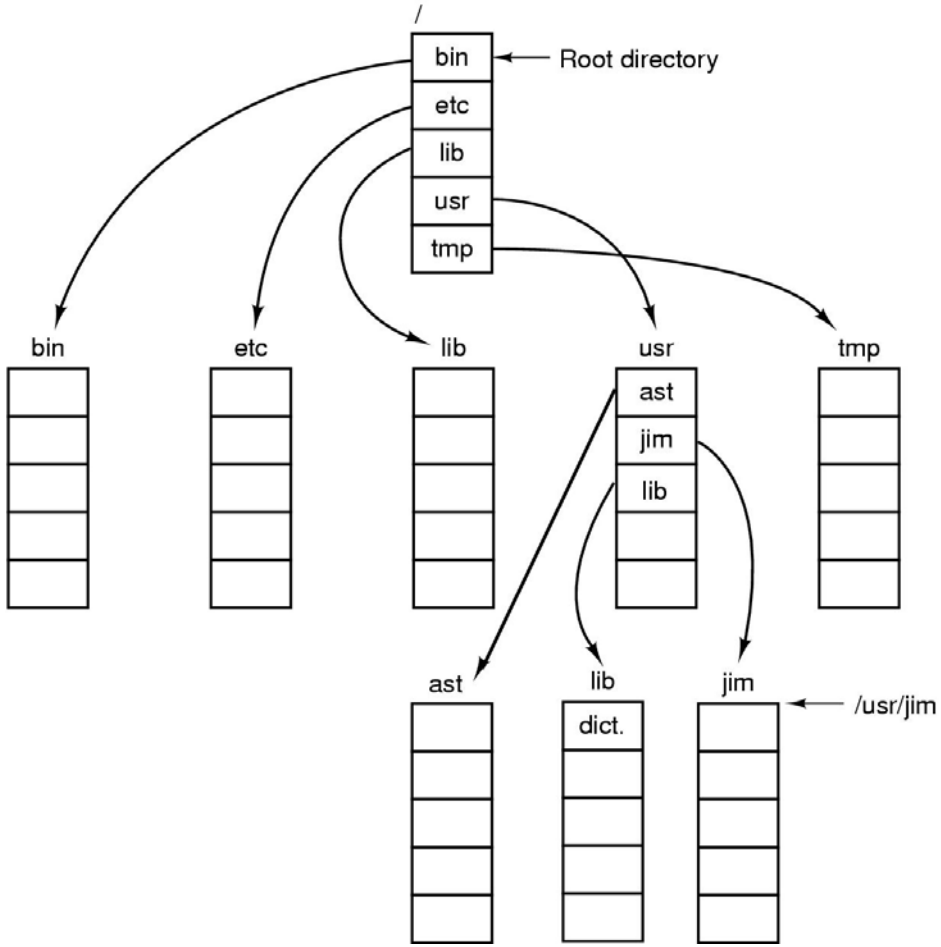


# Хиерархиски системи на именици (1)

- Групирање на датотеките на природен начин
- Секој корисник може да има повеќе директориуми



# Имиња на патеки




Слика 4-8. UNIX дрво на именици

# Имиња на патеки

- Апсолутни патеки
  - Почнува од коренот и е уникатна
    - /home/korisnik
- Релативни патеки
  - Концепт на тековен именик
  - Сите патеки се релативни во однос на тековниот именик
  - Ако работниот именик е /usr/ast:
    - Тогаш наместо

      cp /usr/ast/mailbox /usr/ast/mailbox.bak
    - може:

      cp mailbox mailbox.bak

# Имиња на патеки

- Специјални имиња:
  - “.” и “..”
  - Тековен именик и родител-именик
    - cp /usr/lib/dictionary .
    - cp  /usr/lib/dictionary ../dictionary
    - cp  /usr/lib/dictionary /usr/ast/dictionary

# Системски повици за именици

- Create
- Delete
- Opendir
  - Before listing all the files
- Closedir
- Readdir
- Rename
- Link
  - Linking is a technique that allows a file to appear in more than one directory
- Unlink

# Работа со именици

- Тврда врска (Hard link)
  - Овозможува една датотека да ја има во повеќе од еден именик
  - Со бројач во атрибутите на датотеката се чува информација во колку именици ја има датотеката
- Симболичка врска (Symbolic link)
  - Се креира датотека со патека до вистинската локација на датотеката

# Две страни на имплементацијата на датотечни системи

- Корисници:
  - Како се именуваат датотеките, кои операции се дозволени врз нив,  како изгледа дрвото на именици
- Имплементатори
  - Како се чуваат датотеките и имениците, управување со дисковиот простор и обезбедување ефикасност и надежност

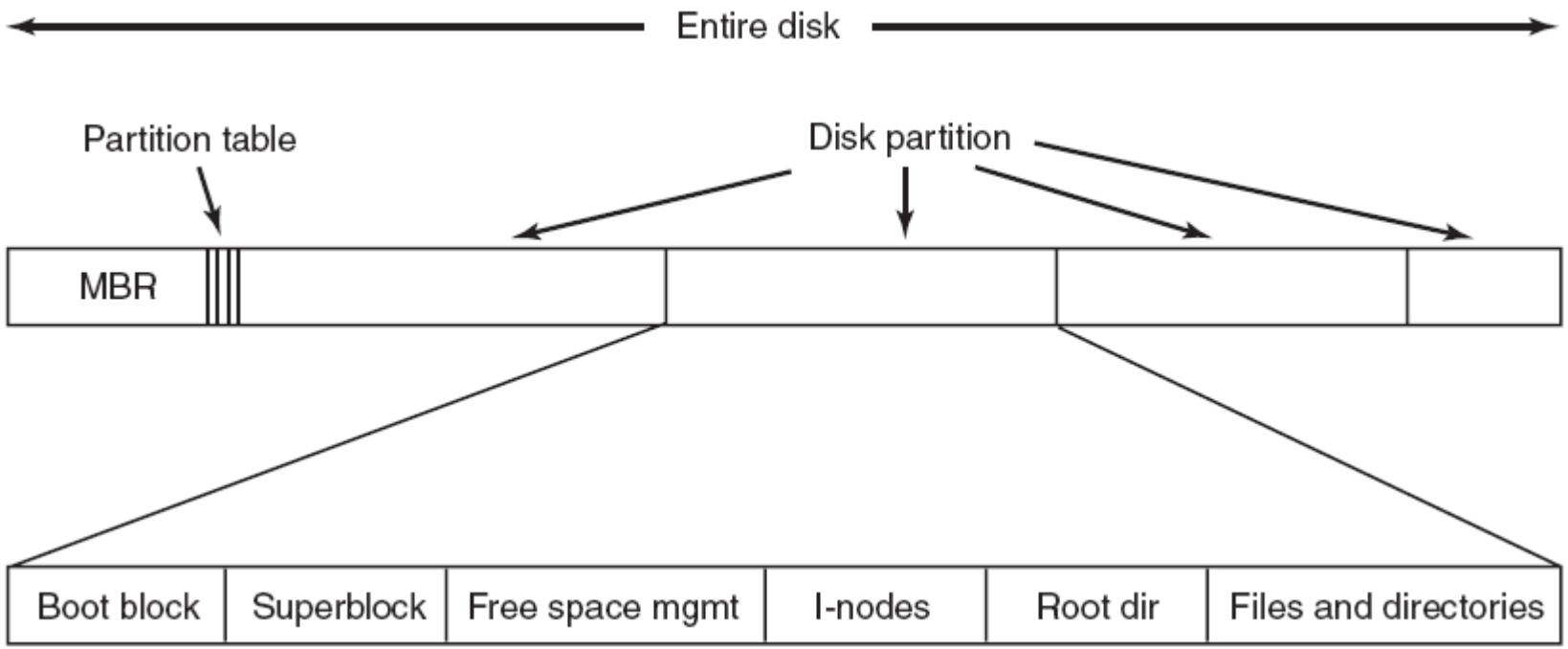
# Организација на просторот на дисковите

- Дисковите најчесто се поделени на повеќе партиции
- Секторот 0 се нарекува MBR (master boot record), се користи за подигање на компјутерот
- На крајот од овој сектор се наоѓа **партициона табела** во која се сместени почетна и крајна адреса на секоја партиција

# Организација на просторот на дисковите

- BIOS го чита и извршува MBR, MBR ја наоѓа активната партиција
- Секоја партиција има boot блок – со кој се вчитува ОС, потоа следи
  - Super block – параметри за датотечниот систем (магичен број, број на блокови, итн.)
  - Информација за слободните блокови (битмапа или листа)
  - Потоа се i-nodes, root именик и сите останати датотеки и именици

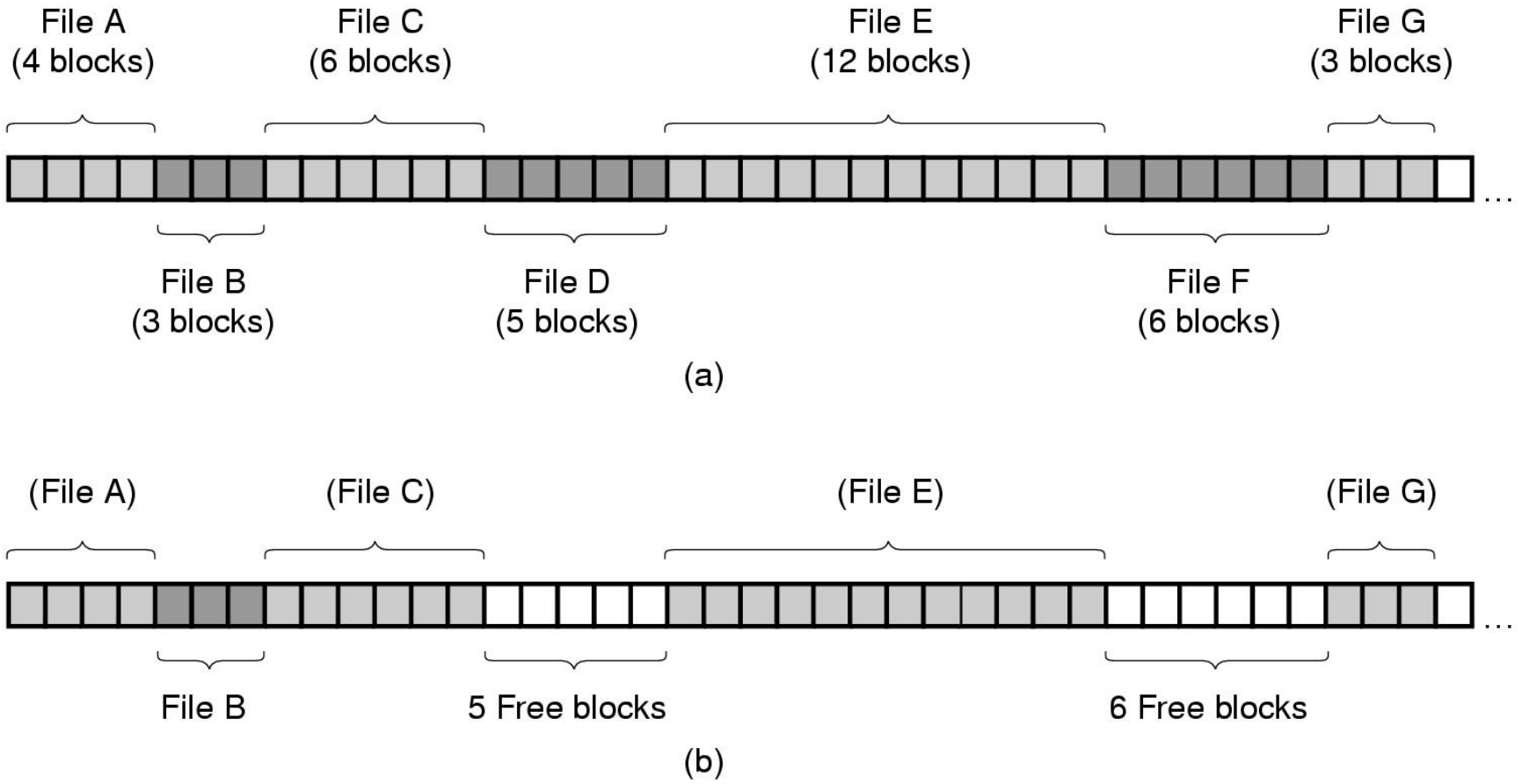
# Изглед на датотечен систем



# Имплементација на датотеки

- Датотеките се составени од повеќе блокови
- На кој начин ќе се алоцираат блоковите во рамки на една датотека?
  - Последователна алокација
  - Алокација со помош на поврзана листа
  - Алокација со помош на поврзана листа со табела во меморијата
  - I-nodes

# Последователна алокација

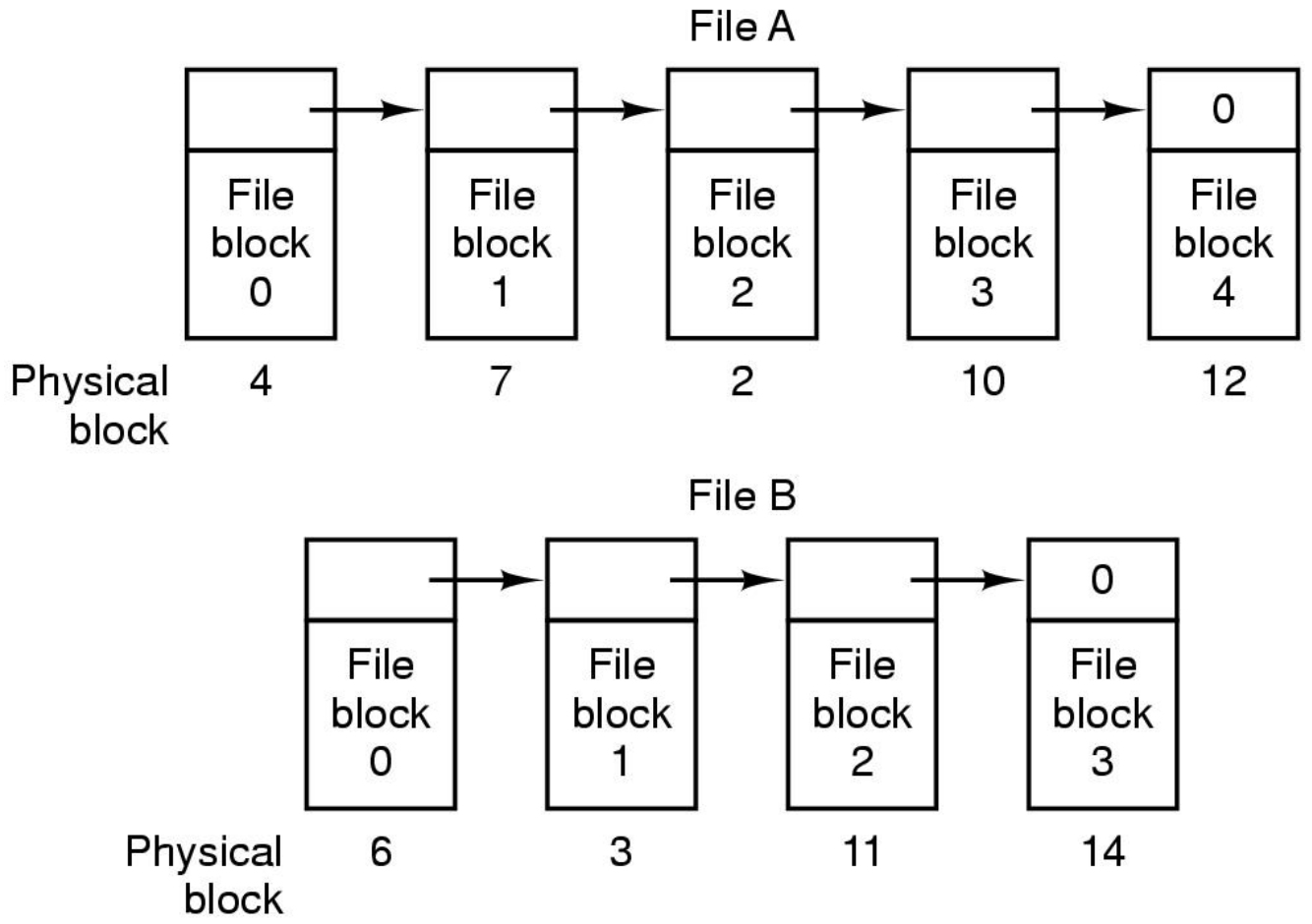



Слика 4-10. (a) Соседна алокација за 7 датотеки (b) Состојба на дискот откако ќе се отстранат датотеките D и E.

# Последователна алокација

- Предности
  - Лесна за имплементација (адресата на првиот блок и бројот на блокови)
  - Читање на датотека во една операција
- Недостатоци:
  - Фрагментирање на дискот
  - Мора однапред да се знае големината на датотеките.
- Имплементација: CD-ROM, extents кај DVD поради Universal Disk Format

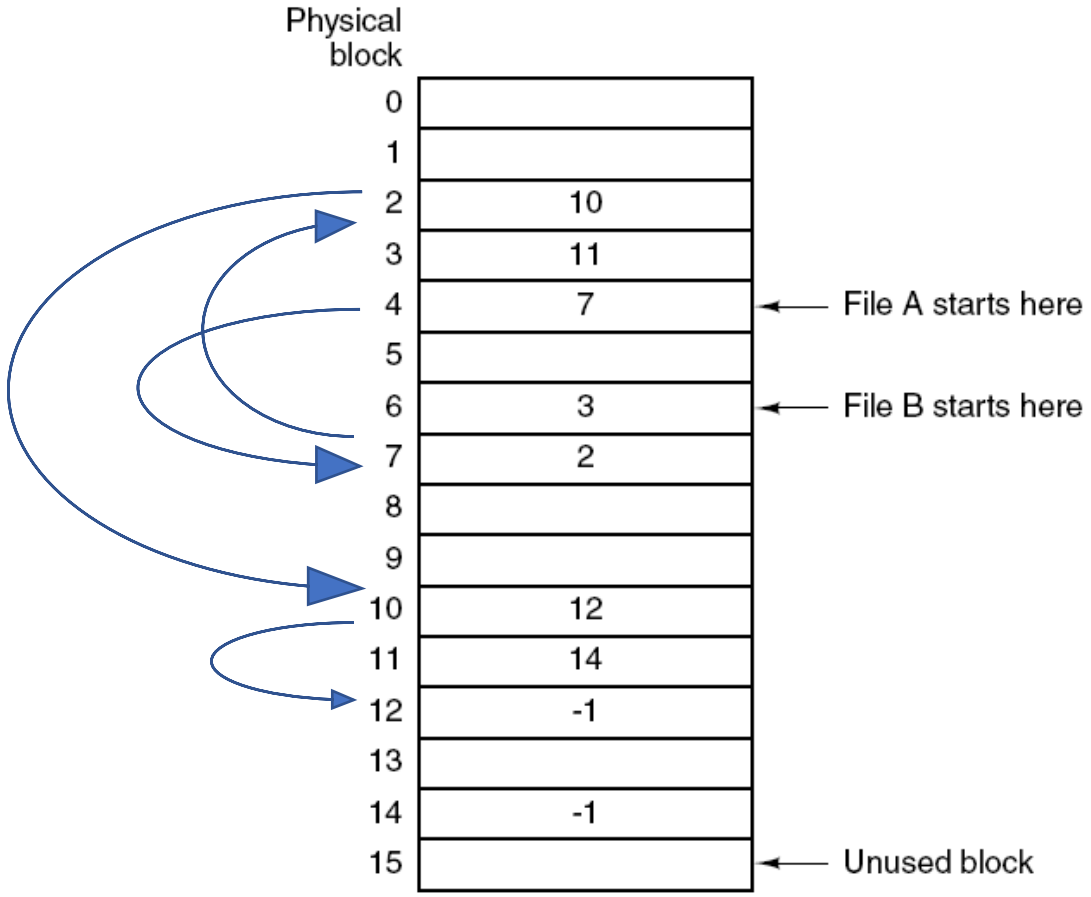
# Алокација со помош на поврзана листа



# Алокација со помош на поврзана листа

- Секој блок покажува на наредниот блок на датотеката
- Предности
  - Само интерна фрагментација кај последниот блок
  - Доволно е да се знае адресата на првиот блок
- Недостатоци
  - Бавно пребарување
  - Се користи дел од блокот како покажувач

# Алокација со помош на поврзана листа со табела во меморијата



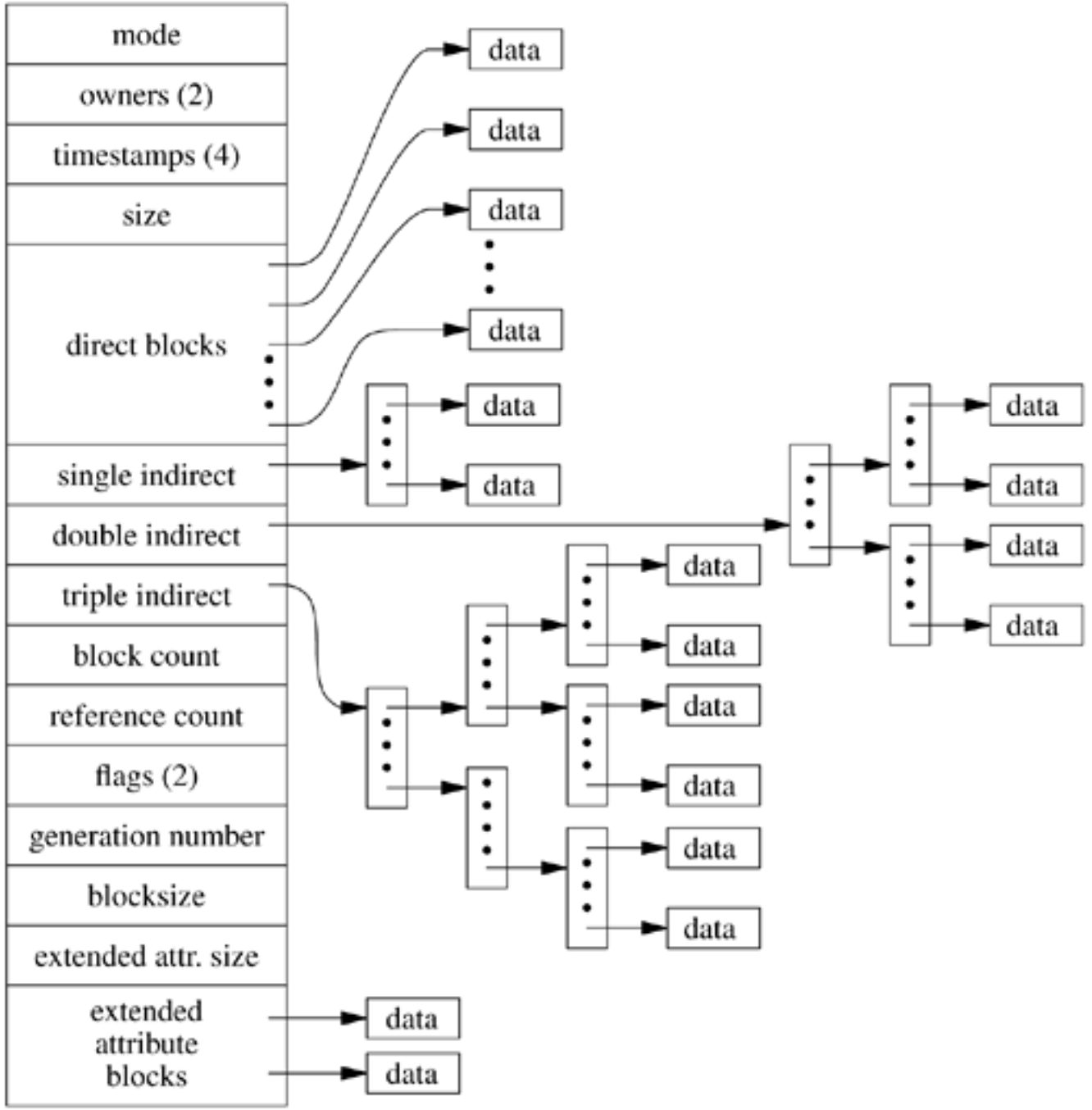
# Алокација со помош на поврзана листа со табела во меморијата

- Покажувачите на блоковите се сместуваат во табела во меморијата
  - FAT-File Allocation Table
  - Првично користен во MS-DOS, но поддржан од сите следни верзии на Windows
- Предности
  - Се користи целиот блок
  - Брз случаен пристап
- Недостатоци
  - Целата табела мора да биде цело време во меморија
  - Лоша скалабилност

# Пример

- Ако имаме и релативно мал диск од 20 GB со големина на блок од 1KB, во табелата се потребни 20 милиони записи со големина од по 3B. Тоа значи дека 60MB од главната меморија се трошат на табелата.
- Диск од 1 TB со големина на блок од 1KB има 1 милијарда записи од по 3B. Тоа значи дека 2.4 GB од главната меморија се трошат на табелата.

# i-nodes


mode
owners (2)
timestamps (4)
size
direct blocks
single indirect
double indirect
triple indirect
block count
reference count
flags (2)
generation number
blocksize
extended attr. size
extended attribute blocks
data

# i-nodes

- На секоја датотека се придружува структура на податоци наречена **индексен јазел** (i-node)
- Јазелот ги адресира атрибутите и адресите на блоковите на датотеката
- Предности
  - Индексниот јазел се наоѓа во меморија само кога е отворена датотеката
- Недостаток
  - Фиксна големина