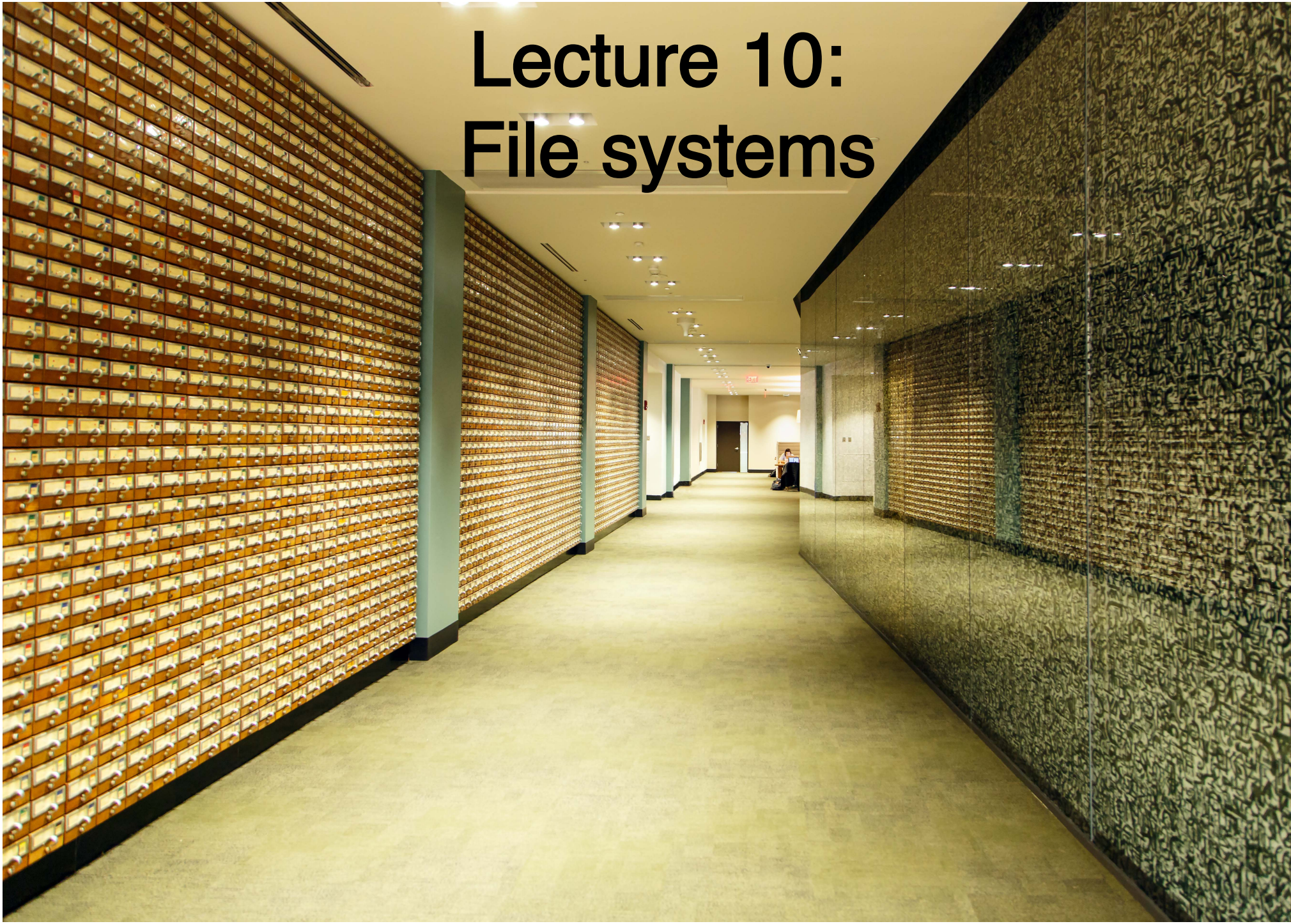
Lecture 10:
File systems

File systems, databases, cloud storage

- file: a sequence of bytes stored on a computer
  - content is arbitrary (just bytes); any structure is imposed by the creator of the file, not by the operating system
- file system: software that provides hierarchical storage and organization of files, usually on a single computer (or nearby)
  - a significant part of the operating system

- database: an integrated collection of logically related records
  - data is organized and structured for efficient systematic access
  - may be distributed across lots of machines & geographically dispersed
- database system: software that provides efficient access to information in a database
  - not usually part of the operating system

- cloud storage: the same thing, but on someone else's computer(s)
  - accessed via the Internet

File systems: managing stored information

- logical structure: users and programs see a hierarchy of folders (directories) and files
  - a file is just a sequence of bytes
    contents determined and interpreted by programs, not the operating system
  - a folder is a special file that contains names of other folders and files
    plus other information like size, time of change, permissions, etc.
    contents are completely controlled by the operating system
  - "root" folder ultimately leads to all others

- physical structure: many options with different properties

- the file system is the part of the operating system that converts between these two views
  - does whatever is necessary to maintain the file/folder illusion
  - hides physical details so that programs don't depend on them
  - presents a uniform interface to disparate physical media

How the file system converts logical to physical

- disk is physically organized into blocks of bytes
  - each block is a fixed number of bytes, like 512 or 1024 or …)
  - reading and writing always happens in blocks
- each file occupies an integral number of blocks
  - files never share a block
  - some space is wasted: a 1-byte file wastes all but 1 byte of the block

- if a file is bigger than one block, it occupies several blocks
  - the blocks are not necessarily "adjacent"

- need a way to keep track of the blocks that make up the file
- usually done by a separate "file allocation table" that lists the blocks that make up each file
  - this table is stored on disk too so it persists when computer is turned off

Converting logical to physical, continued

- every block is part of some file, or reserved by the operating system, or unused

- "file allocation table" keeps track of blocks
  - by some kind of table or array that keeps track of related blocks

- also keeps track of unused blocks
  - disk starts out with most blocks unused ("free")

    some are reserved for file allocation table, etc.
  - as a file grows, blocks are removed from the unused list and attached to the list for the file

    to grow a file, remove a block from the list of unused blocks

    and add it to the blocks for the file

Converting logical to physical: folders / directories

- a folder / directory is a file
  - stored in the same file system
  - uses the same mechanisms
- but it contains information about other files and directories

- the directory entry for a file tells where to find the blocks

  IT DOES NOT CONTAIN THE DATA ITSELF

- the directory entry also contains other info about the file
  - name (e.g., midterm.doc)
  - size in bytes, date/time of changes, access permissions
  - whether it's an ordinary file or a directory

- the file system maintains the directory information
  - very important to keep directory info consistent
  - application programs can change it only indirectly / implicitly

Network file systems

- the file system doesn't have to be local
  - the data could be on some other computer

- network file systems access remote files via network connections
  - user programs access files and folders as if they are on the local computer
  - network file system converts these into requests to ship information to or from another computer across a network

- there has to be a program on the other end to respond to requests
  - "mapping a network drive" or "mounting your H: drive" sets up the connections

- subsequent reads and writes go through the network instead of the local disk

Cloud storage

- the file system doesn't have to be local
  - the data could be on some other computer

- cloud storage systems access remote files via network connections
  - user programs access files and folders as if they are on the local computer
  - file system converts these into requests to ship information to or from another computer across a network

- there has to be a program on the other end to respond to requests
  - connecting to Google Drive or Dropbox or iCloud or ... sets up the connections

- subsequent reads and writes go through the network instead of the local disk

What happens when you say "Open"?

- search for file in sequence of directories as given by the components of its name
  - report and error if any component can't be found
- read blocks of file as needed
  - using the location information in the file allocation table to find the blocks
  - store some of them in RAM

What happens when you say "Save"?

- make sure there's enough space (enough unused blocks)
  - don't want to run out while copying from RAM to disk
- create a temporary file with no bytes in it
- copy the bytes from RAM and/or existing file to temporary file:

```
while (there are still bytes to be copied) {
    get a free block from the unused list
    copy bytes to it until it's full or there are no more bytes to copy
    link it in to the temporary file
}
```

- update the directory entry to point to the new file
- move the previous blocks (of old version) to the unused list
  - or to recycle bin / trash

What happens when you remove a file?

- move the blocks of the file to the unused list
- set the directory entry so it doesn't refer to any block
  - set it to zero, maybe

- recycle bin / trash
  - recycle bin or trash is just another directory
  - removing a file just puts the name, location info, etc., in that directory instead

- "emptying the trash" moves blocks into unused list
  - removes entry from Recycle / Trash directory

- why "removing" a file isn't enough
  - usually only changes a directory entry
  - often recoverable by simple guesses about directory entry contents
  - file contents are often still there even if directory entry is cleared

Forgotten, but not gone…

Q. *Are files really deleted when I empty my Recycle Bin, or are they lurking somewhere in the depths of the computer?*

A. When you delete the items in your Recycle Bin, you are not actually erasing the data from your computer but rather telling Windows (which tracks such matters in its file allocation table) that it is all right to write over those files with new ones.

Until you save new files and data to your computer that can take the space previously allotted to the old ones, the files deleted from your Recycle Bin are still present and could possibly be recovered with a recovery or "unerase" utility program.

New York Times, 10/25/02