Working with Data Storage

- Basic Concepts
  - Number Bases
  - Data on Computers
    - Files
    - Directories
    - Permissions
    - Shortcuts
  - Data Storage
    - Physical
    - Logical
    - Security
- OS-Specific Structures
  - Windows
  - Linux
  - Mac OS X
- File Systems
  - Space
  - Names
  - Directories
  - Metadata
  - Examples

DRAFT

Number Bases

- Numbers are expressed in **bases**, where…
  - The base is the number of possible values a digit can have.
  - The range of values for a digit will be zero through the base minus one.
- ***Examples:***
  - **Decimal: 0 - 9**
  - **Binary: 0 - 1**

Bits and Number Bases

- ***Binary:***
  - Values are `0` and `1`, so any number will be expressed in these digits only.
  - Each digit (`0` or `1`) is called a bit.

Dec	Bin	Dec	Bin	Dec	Bin
0	0	4	100	8	1000
1	1	5	101	9	1001
2	10	6	110	10	1010
3	11	7	111	...	

  - An 8-bit sequence is called a byte, which forms the basic unit of data storage in modern computing.
  - Given most file sizes today, we have larger units -- such as kilobytes (KB), megabytes (MB), gigabytes (GB), and so forth...

Bits and Number Bases

- ***Conversion:*** You calculate the value of the number by multiplying each digit by exponents of the base.
  - Generally, you start where the *right-most* digit
  - ***Binary-to-Decimal:*** 10011

Digit	1	0	0	1	1
Exponent	* 2^4	* 2^3	* 2^2	* 2^1	* 2^0
Product	16	0	0	2	1
SUM	16	16	16	18	19

Bits and Number Bases

- ***Decimal-to-Binary:*** 719
  - Divide the number by two
  - Place the remainder on the end
  - Repeat with the quotient, placing the remainder before the previous digit.
  - Do this until you get a quotient of ***zero***.

Value	Quotient	Remainder
719	359	1
359	179	1
179	89	1
89	44	1
44	22	0
22	11	0
11	5	1
5	2	1
2	1	0
1	0	1

1 0 1 1 0 0 1 1 1 1

Bits and Number Bases

- ***Octal:***
  - Values -- and therefore digits -- are `0` through `7`
  - Notice a pattern here. As the base increases, a digit can have more values. For this reason, the same value can be represented with fewer digits.

  One octal digit is equivalent to three binary digits -- either three binary digits or one octal can express 8 different values (**2^3**)
  - You may see octal used, for example, in Linux permissions...

Dec	Oct	Dec	Oct	Dec	Oct
0	0	4	4	8	10
1	1	5	5	9	11
2	2	6	6	10	12
3	3	7	7	...	

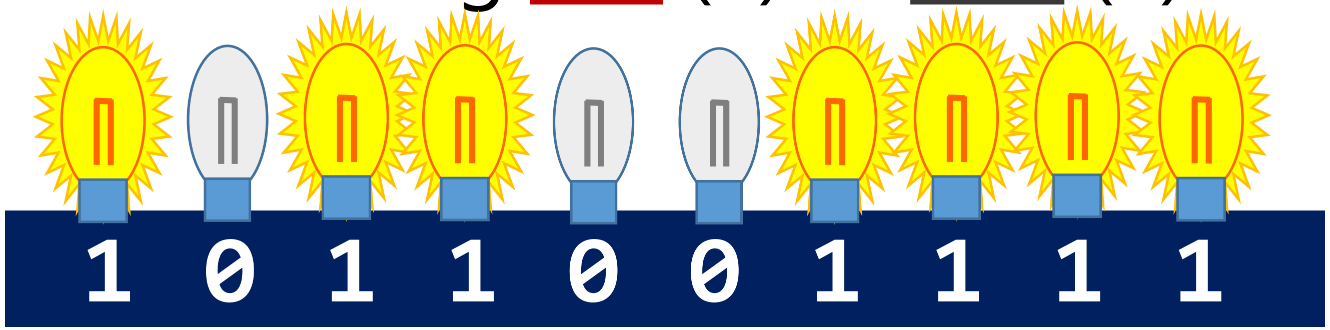
Bits and Number Bases

- ***Hexadecimal:***
  - Values are 0 through 15
  - Digits are 0 – 9, with 10 – 15 represented by A through F
  - A hex digit is equivalent to a *quartet* (4 bits)
  - ***Example:*** 719 → 10 1100 1111 → 2 c f
  - This way, you can easily convert *back and forth* between the two

Hex	Binary	Hex	Binary
0	0000	4	0100
1	0001	5	0101
2	0010	6	0110
3	0011	...	

Bits and Number Bases

- A number expressed in binary digits is a *bit string*, and you can think of them as being **ON** (1) or **OFF** (0)
- ***For example:***



- *Selecting bits*:
  - Sometimes, you will want to "turn" some bits on or off
  - This will be the case in scenarios where individual bits or bit sequences in the string have meaning, *in their own right*.

Bits and Number Bases

- This can be accomplished by using a **bit mask**, along with **bitwise operations**.
  - A *bit mask* is simply a bit string, where the different bits or bit sequences have special meaning
  - A *bitwise operation* acts upon a bit pair to produce **0** or **1**, and we will look at two of them:
    - **OR** is used to turn bits **on**
    - **AND** is used to turn bits **off**

Bits and Number Bases

- OR operation:
  - Any bit ***or*** 1 is turned/left **ON**
  - In contrast, any bit ***or*** 0 is simply left **unchanged**

Bit		Mask	Result	
1	OR	1	1	Turned ON (if zero, would have been)
0	OR	1	1	Turned ON
1	OR	0	1	Unchanged
0	OR	0	0	Unchanged

  - If you use a bit mask with OR , it will turn some bits on while keeping the others as they were.

Bits and Number Bases

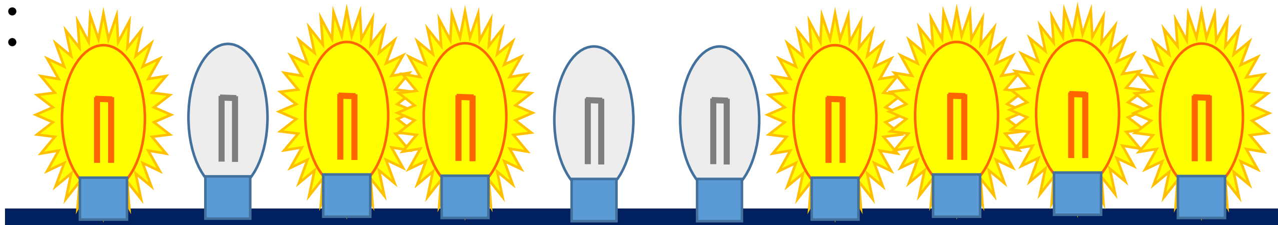
- AND operation:
  - Any bit ***and*** 0 is turned/left **OFF**
  - In contrast, any bit ***and*** 1 is simply left **unchanged**

Bit		Mask	Result	
1	AND	1	1	Unchanged
0	AND	1	0	Unchanged
1	AND	0	0	Turned OFF
0	AND	0	0	Turned OFF (if zero, would have been)

  - If you use a bit mask with AND , it will turn some bits off while keeping the others as they were.

Bits and Number Bases

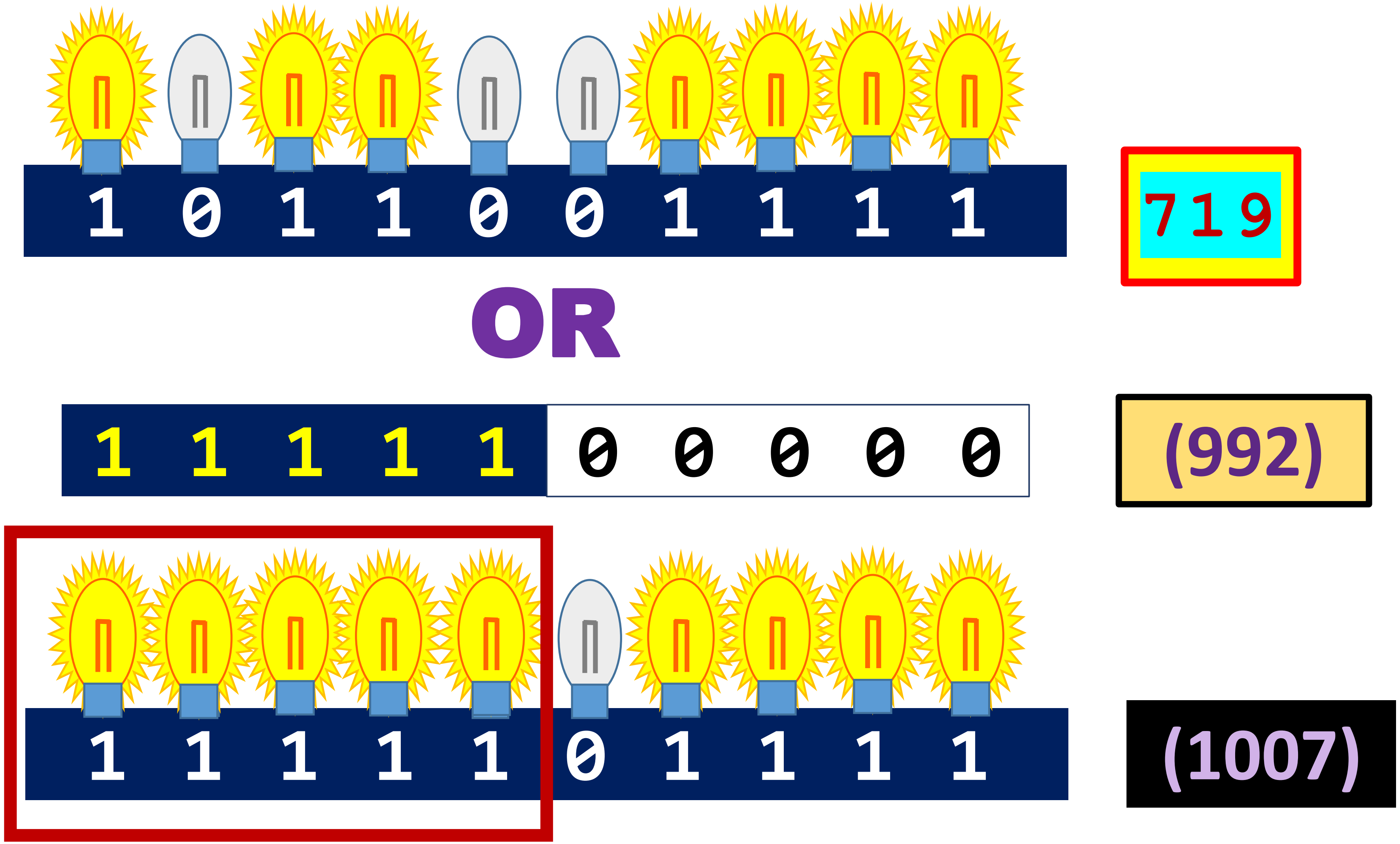
- Let's look at an example:
  - Our original bit string:



`1 0 1 1 0 0 1 1 1 1`

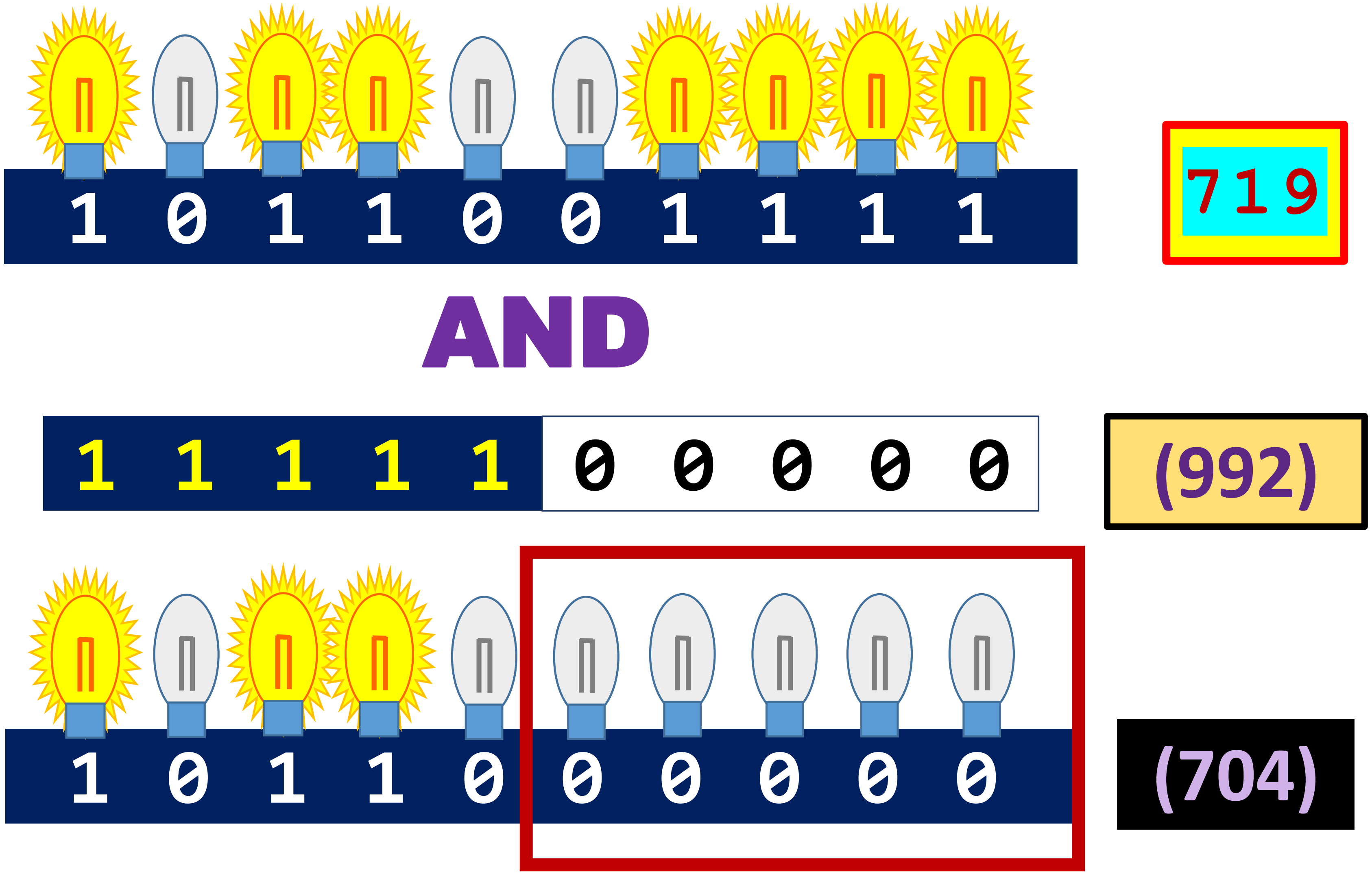
  - Bit string's decimal value: `719`
  - A bit mask: `1 1 1 1 1 0 0 0 0 0` (992)

Mask applied with OR



First 5 bits are turned **ON**

Mask applied with AND



Last 5 bits are turned **OFF**

Data on Computers - Files

- A file is essentially a container for information, and it is one of the basic objects used by a computer's operating system.
- It may consist of one or more of the following:
  - User's personal files
  - Software programs
  - Configuration data for
    - The system
    - The users
    - Specific programs
  - Other code and software
  - And so forth…

Data on Computers - Files

- Files may be visible (usually) or hidden. Hidden files will usually not appear in file browsing interfaces unless an option is set.
- Some can be accessed by any user on the machine, while others can only be accessed by…
  - Specific users
  - Specific user groups
  - Software
  - The system itself

Data on Computers - Files

- You can think of file data in two ways:
  - All files will consist of binary data, in the form of ones and zeroes. Everything boils down to bits and bytes.
  - Aside from the concrete bits/bytes, there is also the question of: What is the information the file is intended to represent and store?
- Files come in different kinds of formats. The format of a file is a way of organizing/representing data so that certain types of software can interpret and work with it.

Data on Computers - Files

- The most basic file format would be plain text.
    - With ASCII text, each byte corresponds to a single text character. This is about as simple as file data will get.
    - Other types of text, one character may require two or more bytes.
    - Examples of plain text format:
        - Text files
        - HTML, CSS, and JavaScript
        - Software code
    - These can usually be read in a basic text editor.

Data on Computers - Files

- Because plain text can be limiting, we have more complex ways of creating, storing, and viewing information:
  - Word-processed documents
  - Presentations
  - Spreadsheets
  - PDF (Portable Document Format) files
  - ZIP files (containing compressed files and folders)

Data on Computers - Files

- In these, data in the file is structured in a very specific way, so that the files can be used effectively by compatible software programs.
  - These would only be viewable or editable -- practically, at any rate -- with particular software programs
  - If you try to view the raw file data, as plain text, it will look relatively incomprehensible -- or completely!
  - It is meant to be program-readable, as human-readable is not a concern here.

Data on Computers - Files

- You are probably to filenames having extensions: .txt, .docx, .pdf, and so forth.
  - These are often used by the user to know the file format.
  - The operating system may use them, likewise. Moreover, the operating system can use file extensions to determine which program should be used to open a file.
  - These can be default programs or user-specified programs.

Data on Computers - Files

- However, a file extension is not equivalent to a file being of one format or another.
  - File extension is part of the filename
  - The file format has to do with how the data is structures within the file.
  - For example, a Word document in .docx format is a bit more than you may think…

Data on Computers - Files

- When naming files, it's important to keep in mind two questions:
  - What will the OS allow you to do?
  - What is most practical to do?
- At the very least, two files in the same folder cannot have the same name.
  - Here, and in general, you should start considering the file's extension as part of its name
  - For example, usually file.txt and file.docx could exist in the same folder

Data on Computers - Files

  - Of course, even two files with identical names could exist -- so long as they are located in different folders
  - Also, consider whether or not the file system is case-sensitive; in other words, are file.txt and File.txt two different names?
- Usually, the file system will have restrictions on the use of some characters in a file name.
- Other than that, you have much freedom. However, you should also make sure your file names are helpful.

Data on Computers - Files

- Some tips:
  - Avoid spaces in file names. They just complicate things too much
  - There are other options for separators: hyphens, underscores, periods, etc.
  - Be descriptive, so that you have a good initial idea of what is in the file, based on the name.
  - Avoid excessively long filenames
  - Find some file naming conventions relevant to the material/domain
- Part of this, of course, is the creation and use of a practical directory (i.e., folder) structure!

Directories

- Files will go into containers, called "directories". You may be used to calling them "folders", as well.
- In addition to files, a directory may also contain other directories.
  - Those (inner) directories would be considered subdirectories.
  - On a computer, these will be organized into a hierarchy, where we say that…
    - A parent directory is the container of…
    - …one or more child directories.

Directories

  - These are relative to a position in the hierarchy. A directory might be a parent to some subdirectories, while being a child to its own parent directory.
- At the apex of the directory hierarchy is a "root" of sorts, which is unique in that it has no parent itself.
  - This will differ by operating system
  - In modern versions of Windows, you may have an area called "This PC", where one or more drives are listed.

Directories

- On Unix-based systems, you have an official "root" directory named /
  - To make it more confusing, there may also be a directory inside there, with the name root
  - However, what makes / special is that it is the highest level in a Unix/Linux directory hierarchy!
- You can have directories for various purposes, including (but not limited to) the following:
  - The operating system
  - Applications: executable files and application data
  - User data: personal files and user-specific app data
  - Temporary files

Directories

- Every file or folder on a computer will have a path that indicates its location within the larger file system
  - This path will be a series of directory locations that ends in the file/folder's location.
  - A path can take one of two different forms, either of which can usually be used, so long as it is a valid path to an existing file:
    - Absolute path: A path from the root of the directory hierarchy to the file in question. This type can always be recognized, regardless of your current directory.
    - Relative path: A path to the file, from your current directory.
      - If the file is in a subdirectory of your current, then it's just a matter of traversing downwards
      - Sometimes, however, you may need to travel up the directory tree and down another branch.
  - Depending on the operating system, there may be differences in file path syntax.

Shortcuts

- Sometimes, you want to have a file or folder in multiple places at once, BUT you do not want to copy or duplicate it.
- Instead, you want the same file/folder, just in different places, and moving it to and fro would be impractical
- Fortunately, most any operating system allows for various types of "shortcutting" operations -- where you create an object that may look like a normal file or folder but in fact is only pointing to one.

Shortcuts

- You may hear terms such as these:
  - "shortcut"
  - "hard link"
  - "symbolic link"
- These will differ according to the file system and operating system.
  - In some cases, this object will point directly to the file or folder in question.
  - In other cases, the object will be more like a "data file" that holds a path (absolute or relative) to the actual file or folder.

Shortcuts

- Whatever operating system you are using, it is important to understand the types of shortcuts you are using, as well as how they work.
  - What happens if you have a shortcut to a file or folder that does not exist or has been deleted or moved? Or renamed?
  - What happens if you delete or move a shortcut itself?

Permissions

- A computer will have users and groups.
  - A user may be a human end-user or an entity representing some aspect of the system and/or its software
  - There may also be default user groups, as well as other groups created by human administrators
  - Depending on the operating system, there will be certain default system users and default user groups.
- These users and groups will be selectively allowed or denied certain permissions, with respect to specific files and directories on the computer.

Permissions

- You can think of file and directory permissions as belonging to three types:
    - Read: Opening a file for reading or viewing the contents of a directory.
    - Write: Changing the data within a file or altering the contents of a directory (by adding, removing, or renaming files and subdirectories within it)
    - Execute: Running a executable file (like a program) or accessing the contents of a directory.

Permissions

- For any particular file or directory, describing its permissions is a matter of enumerating…
  - ***Which*** users and groups have some permissions, and…
  - ***What*** permissions (read, write, and/or execute) they possess
- When attempting to troubleshoot computer behavior, knowing permissions will be invaluable
- Moreover, well-structured permissions are an important aspect of a computer's basic security.

Physical vs. Logical

- In computer technology, we may speak of physical vs logical components:
  - ***Physical*** tends to be concrete; often referring to the actual material objects.
  - ***Logical*** often refers to something more abstract or virtual.
  - Example: A computer may have a physical hard drive, but…
    - It could be partitioned into two or more logical volumes…
    - Which the computer would treat like entirely separate drives

Data Storage

- First, we can think of physical devices used for permanent data storage. These are designed to be:
  - Attached to a computer or other intelligent electronic device
  - Readable and (usually) writeable by that device
- Examples include:
  - Hard-disk drives (HDD) and solid-state drives (SSD)
    - Built into the device itself
    - External/detachable

Data Storage

- Flash drives (e.g., USB sticks)
- Optical discs such as CDs and DVDs, which require special drives for reading and writing

- These devices feature various trade-offs in data speed, storage capacity, purchase price, and longevity.
- Many data storage devices -- like HDDs and SSDs -- can be divided into sections, or partitions.
  - The partitions will be on the same physical device/object
  - The operating system may recognize this fact, but it will also treat them as if they were separate devices.

Data Storage

- When formatting a disk, you can decide…
  - How many partitions to create
  - What to name them
  - What sizes to specify (with some limits, of course!)
  - The file-system for each partition
- At least, an HDD or SSD will have at one partition, plus possible "hidden" partitions (e.g., for booting or OS recovery)

- Partitions can be useful in cases where you want to
  - Manage different bodies of information differently
  - Achieve stricter separation between them, without the requirement of multiple physical drives.

Data Storage

- Once a data storage device is attached to a computer, the latter can mount it (or its partitions) as different storage volumes. Think of Windows and its drive letters…
- Think of "mount" as signifying that the computer
  - Recognizes the storage device
  - Connects to treat the files/folders within as part of its own file system
- Physically, we can think of data storage devices and their partitions

Data Storage

- We can also have logical drives and volumes, where the object does not necessarily correspond to one discrete physical device or partition.
  - Partitions are "sort of" logical in the sense that they cause a single device to appear as multiple
  - Other examples include:
    - Disk images, or "virtual disks"
    - RAID (Redundant Array of Independent Disks) drives, where multiple disks appear as a single drive, for extra protection against data loss
    - Network drives

Data Storage

- Different operating systems handle this differently
- This allows for a certain amount of flexibility between:
  - On one hand, how the data storage facilities appear to the human user
  - On the other hand, how the storage is actually implemented
- As a result, file and folder management can be made more secure and convenient

OS-Specific Structures

- Windows: The root of your directory hierarchy may have a name like `This PC` and it will feature a list of drives, indicated by letters:
  - `C` drive: Usually the main storage drive.
  - If the computer has more internal disks, these may be referred to by subsequent letters: `D`, `E`, …
  - If you attach removable storage, the OS may arbitrarily assign it another letter, such as `R` or `F`
  - You might also map a networked drive to some letter, such as `Z`

OS-Specific Structures

- Linux: The first thing to remember about a Linux operating system is that "everything is a file"
  - This is true whether you are talking about a directory, a link (i.e., a shortcut in Linux), or an actual file
  - Sometimes, this makes command-line processing easier
  - At other times, you may have to determine if something is a directory or a file (or a link) before further interaction with it.

OS-Specific Structures

- Instead of drive letters like Windows has, you have this directory: `/dev`
  - In other words, `dev` is a subdirectory of `/`
  - "Device files" are stored here
  - For a standard hard drive, for example, you may have a file like `/dev/sda`
  - If partitioned, those partitions may be named sequentially:
  - `/dev/sda1`
  - `/dev/sda2`
  - `/dev/sda3` (And so forth...)

File Systems

- When you want to prepare a new disk or partition to store data, you will format it -- which overwrites all previous data, amongst other things.
- Part of this will include defining a file system -- a phrase that can have different meanings, depending on usage.
  - Sometimes, "file system" may simply refer to a disk, disk partitions, or the file/directory hierarchy on one of those
  - Formally, a "file system" is a structured approach to writing data to storage and retrieving it from.
- Space

File Systems

- When you want to prepare a new disk or partition to store data, you will format it -- which overwrites all previous data, amongst other things.
- Part of this will include defining a ***file system*** -- a phrase that can have different meanings, depending on usage.
  - Sometimes, "file system" may simply refer to a disk, disk partitions, or the file/directory hierarchy on one of those
  - Formally, a "file system" is a specific, structured approach to writing data to storage and retrieving it from.

File Systems

- All data is ultimately bits and bytes, so there must be some way to know where some data ends and other data begins
- The data must be structured and indicated, in storage, so that it will be usable
- One aspect of this is the allocation of ***space*** on disk.
  - Usually, a file system will allocate space on a unit by unit basis, where each unit has a set size.
  - A.k.a., "blocks", "clusters", etc.

File Systems

  - This size is some number of bytes, where that number is a power of two: 256-byte, 4 KB (4096 bytes), 64 KB, etc.
  - Even if a file's data only amounts to a particular size, the actual "size on disk" may be higher because the file gets some number of blocks for its data size.
  - Space can be wasted or "fragmented"
- A file system will have rules/restrictions for file ***names*** -- allowed characters, length, case-sensitive vs. insensitive
- There are also methods of file grouping and organization -- i.e., folders/***directories***