SAMPLE CHAPTER

Level Up

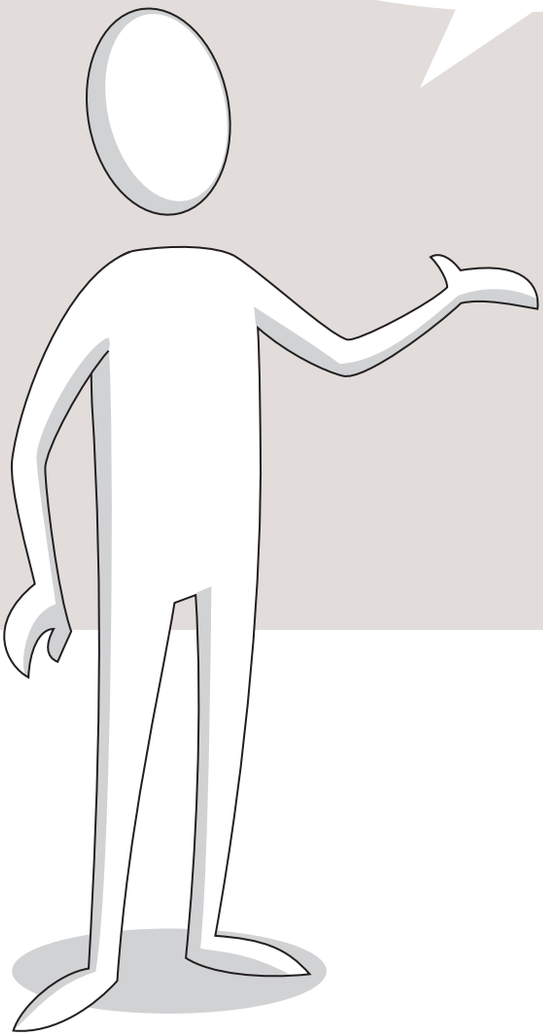
Through **Digital Discoveries** 11

Mc Graw Hill

binarylogic

Contents

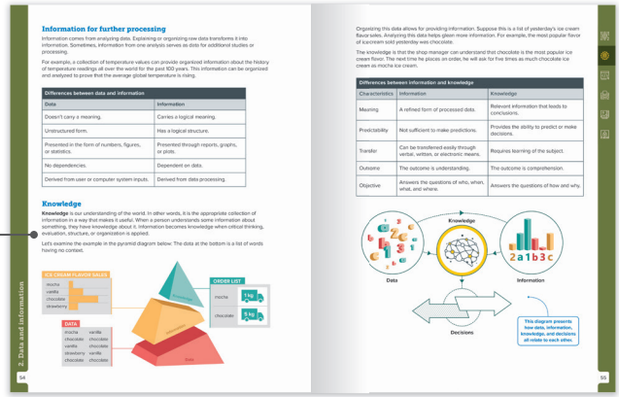
Welcome!
You're about to embark on
a journey that goes beyond just using
technology—you'll learn how it really works
and how you can shape the future with it.
From coding challenges to real-world
applications, this course will help you sharpen
your skills and spark new ideas.
Let's level up together!



Key Features

An innovative approach to building digital competencies, developed by expert educators.

Each unit offers straightforward explanations and contemporary examples, making technology concepts accessible and relevant.



Curriculum aligns with the latest industry standards, preparing students for certifications and future careers.

Every unit includes a variety of tasks and activities designed to help students build essential digital competencies.

Projects and exercises throughout the course reinforce students' understanding and practical application of digital skills in real-world scenarios.

Well-defined learning goals and hands-on, applicable digital skills.

Students learn about platform diversity, expanding their digital toolkit and adaptability.

Each unit organizes key terms that are crucial for digital literacy, equipping students for today's technology-driven workplace.

1. Systems architecture basics

Understanding how computer systems and security work is essential in today's digital world, where protecting data and ensuring system efficiency are critical. This unit covers how components like the CPU and memory work together, how data is stored and accessed across various storage types, and how operating systems manage tasks and resources. You will also learn how file systems organize and protect files and explore access control methods that secure data through permissions, authentication, and security policies. By the end of this unit, you will have a solid understanding of how computers function and how they remain secure.

Learning Objectives

In this unit, you will:

> identify the parts of a computer's architecture and explain their functions.
> understand how the CPU processes instructions through the fetch-execute cycle.
> understand how storage devices work.
> understand the advantages and disadvantages of different storage types.
> learn how the operating system manages memory and processes in a computer.
> understand how the CPU handles multiple programs through multiprogramming.
> identify the main components and structure of a file system.
> understand how files and directories are organized within a hierarchy.
> outline the methods for controlling physical and digital access to systems.
> learn what user roles and permissions are and how they help manage access in a secure way.

Tools

> Microsoft Windows

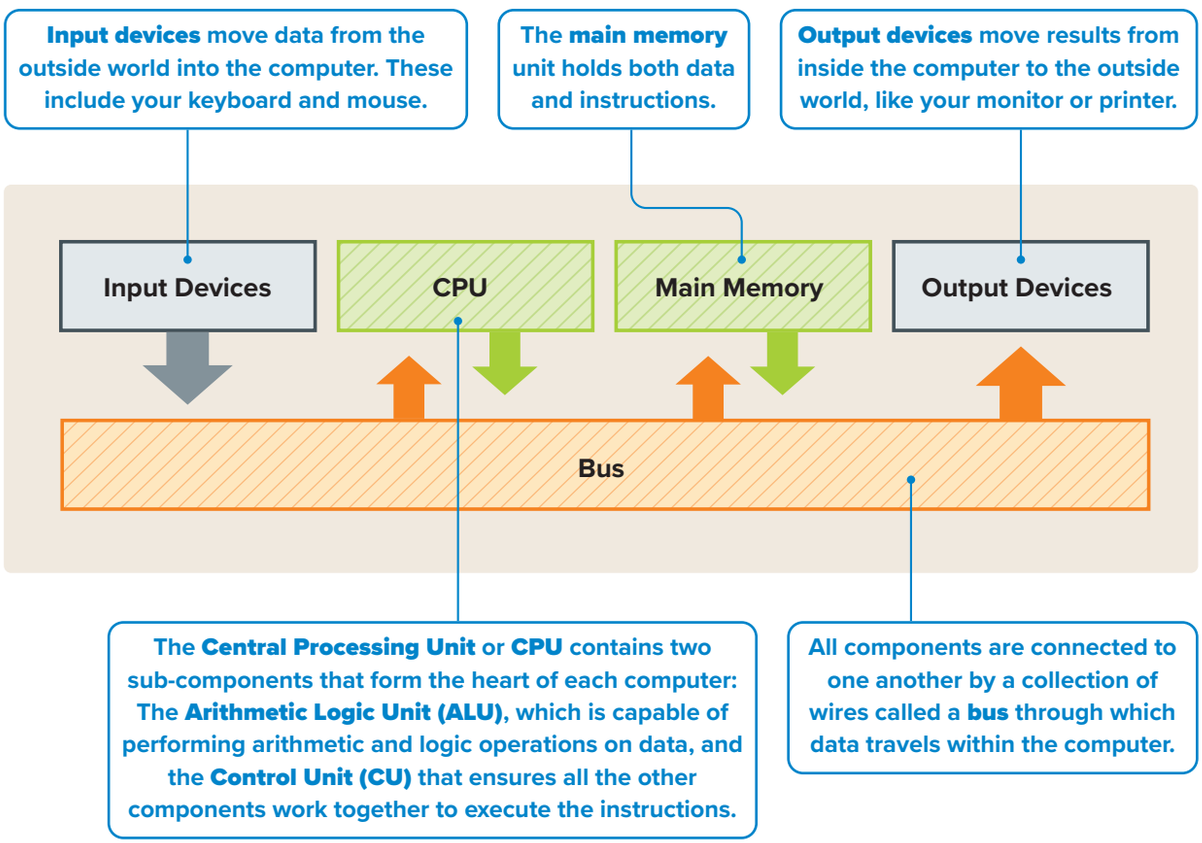
LESSON 1

Computer architecture

What happens inside a computer when you run a program?
Why is it important to store data and instructions in a computer's memory?

All computers can perform three fundamental actions: storing, retrieving, and processing data. Of course, for computers to be useful, we must provide instructions on what to do with the data. These instructions must be stored where the computer can access and retrieve them. Since computers are binary machines, instructions are stored as binary sequences. Data and processing instructions are logically the same and can be stored together. Another major characteristic of computers is the separation between information-processing units and storage units.

These basic traits form the **von Neumann architecture** on which all modern computers are based. In the von Neumann architecture, we can identify distinct components that work together in order for the computer to function.



The importance of the von Neumann architecture

The von Neumann architecture is significant because it provides a clear, organized way for computers to store and process information, laying the foundation for how modern computers operate. A key aspect of the von Neumann model is the concept of the stored program, where both data and instructions are stored together in the same memory space. This design allows instructions to be readily modified, enabling programs to update themselves and perform complex tasks.

Architecture adaptability

One benefit of this architecture is its adaptability. The same hardware can execute a variety of software programs, so different tasks (like calculations, graphics processing, or data management) can all be performed on the same system. This flexibility is a fundamental principle of modern computing, where the same computer can run everything from operating systems to web browsers and games.

Limitations of the architecture

However, this architecture also introduces a limitation known as the von Neumann bottleneck. Since both data and instructions use the same memory and bus system, the CPU can only access one at a time, slowing down performance when there is high data demand. This bottleneck is one reason why modern computers use cache memory and advanced techniques to enhance performance.

Despite these limitations, the von Neumann architecture remains central to most computers, enabling the data processing behind daily computing tasks. By separating memory, input/output, and processing units, the architecture offers a structured and systematic way for computers to handle instructions and execute tasks, forming the basis of modern computing systems.

The fetch-execute cycle

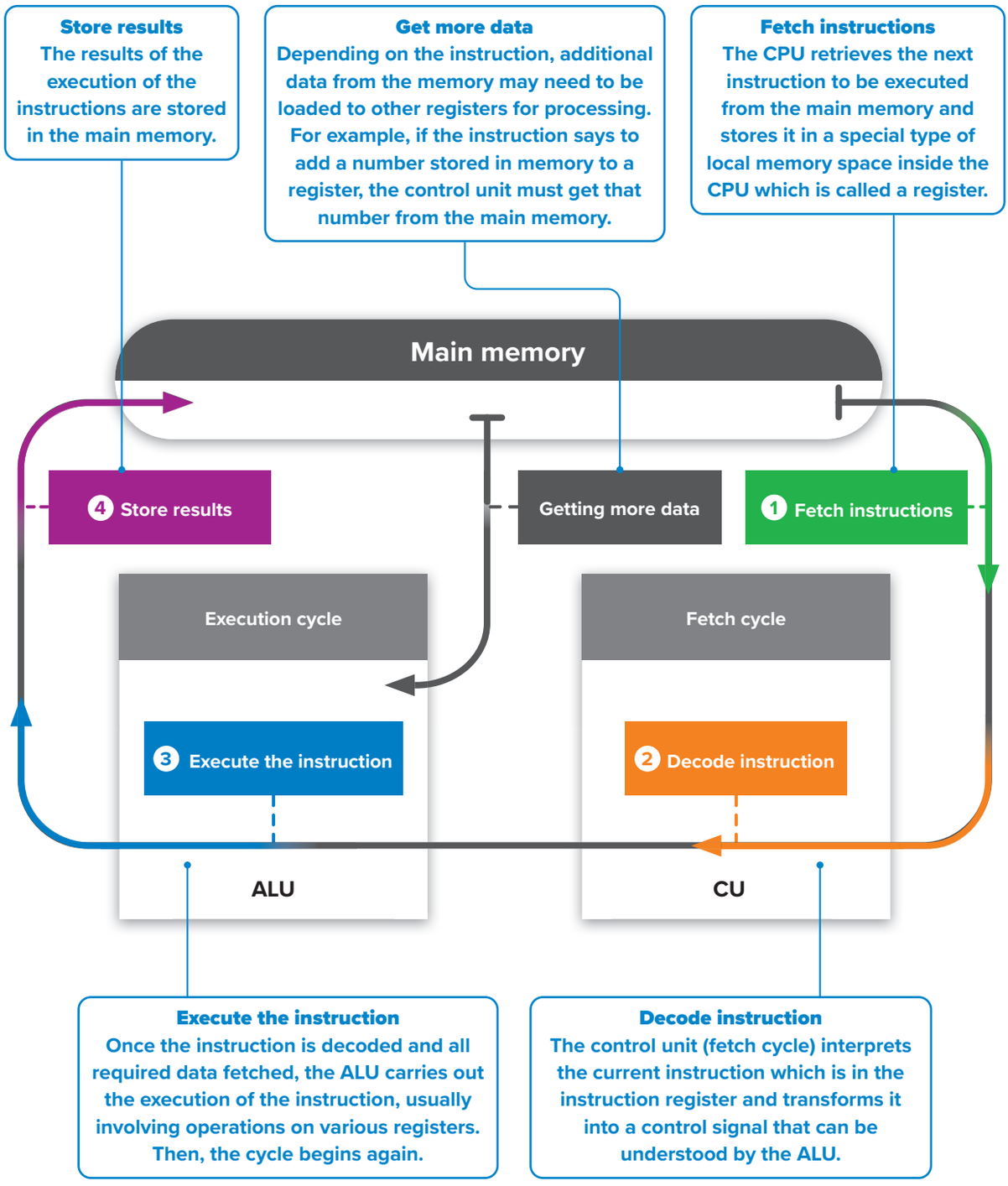
Now that you are familiar with the main architecture of a computer, let's find out how the instructions are executed and process data. This is also known as the **fetch-execute cycle**. Remember that both data and instructions are stored in the main memory.

The steps of the fetch-execute cycle can be diagrammed:

1. Fetch instructions.
2. Decode instruction.
3. Execute the instruction.
4. Store results.

History

John von Neumann described the computer architecture that takes his name together with other engineers during the development of ENIAC, the first programmable, general-purpose digital computer, in 1945. Von Neumann was a brilliant Hungarian mathematician who made many contributions to various fields, including mathematics, physics, and computer science.



Smart Tip

ROM, which stands for Read Only Memory, is another type of memory. Like RAM, it provides random access to data but is non-volatile, meaning it can retain its stored data even when the power is turned off. The instructions that are stored inside it are permanent —they cannot be erased and rewritten, only read. That is why ROM is used to store the instructions that the computer needs to start itself up. These instructions are called Firmware.

An in-depth analysis of the fetch-execute cycle

The fetch-execute cycle, also known as the instruction cycle, is the process through which a CPU retrieves and executes instructions stored in the main memory. This cycle is essential to a computer's operation, as it defines how instructions are processed, allowing the CPU to carry out complex tasks step by step. Understanding each phase of this cycle is key to understanding how a computer functions.

1. **The fetch phase**

   The cycle begins with the fetch phase, where the CPU retrieves the next instruction to be executed from the main memory. The **Program Counter (PC)**, a special register within the CPU, holds the memory address of the next instruction. The address is sent along the address bus to the main memory, and the data bus carries the instruction back to the CPU. The fetched instruction is temporarily stored in the **Instruction Register (IR)**, where it is held until the CPU is ready to decode it. After fetching, the PC is incremented to point to the following instruction, preparing for the next cycle.

2. **The decode phase**

   Once the instruction is in the Instruction Register, the **Control Unit (CU)** decodes it. During the decode phase, the CU interprets the instruction, determining the operation required and identifying any data or addresses needed to execute it. For example, the instruction might specify an arithmetic operation (like addition) or a data transfer (such as moving data between registers). The CU then generates control signals based on the decoded instruction, directing the **Arithmetic Logic Unit (ALU)** or other CPU components on what to do next.

3. **The execute phase**

   In the execute phase, the CPU performs the actual operation specified by the instruction. The ALU is typically involved in this phase, especially if the instruction involves arithmetic calculations or logical comparisons. Data needed for the operation may come from registers or be fetched from the main memory during this phase. The ALU processes the data, generating an output (a result) that is then handled in the store phase.

4. **The store phase**

   After the instruction is executed, the store phase (sometimes called write back) transfers any results generated by the ALU to their destination. This may mean writing the result back to a register for quick access or to main memory for long-term storage. For example, if the executed instruction was an addition, the sum might be stored in a specific register for later use in another operation.

The importance of the fetch-execute cycle

The fetch-execute cycle is fundamental because it allows the CPU to process instructions sequentially, executing complex programs by breaking them down into small, manageable steps. Each cycle builds on the previous one, enabling the CPU to handle everything from simple calculations to advanced processes. By continuously repeating the fetch-execute cycle, the CPU keeps the computer responsive and efficient, processing millions of instructions per second.

The cycle also highlights the CPU's need for coordination between components like the CU, ALU, and registers. Each phase relies on precise timing and interaction among these components, demonstrating the CPU's role in orchestrating data flow and ensuring that tasks are performed correctly and efficiently.

Hands on!

1. **Read the following sentences and put a check mark for True or False.**

	True	False
1. The Arithmetic Logic Unit (ALU) ensures that all components work together to execute instructions.		
2. The von Neumann architecture separates processing units and memory units.		
3. The fetch-execute cycle begins with the CPU decoding the instruction.		
4. The fetch-execute cycle involves four main steps: fetching, decoding, executing, and storing results.		
5. One advantage of the von Neumann architecture is its ability to run different types of programs on the same hardware.		
6. Cache memory and advanced techniques help reduce the impact of the von Neumann bottleneck.		
7. The Instruction Register (IR) temporarily holds the fetched instruction before it is decoded.		
8. The execute phase involves writing the execution results back to memory or registers.		
9. The Program Counter (PC) holds the memory address of the next instruction to be executed.		

2. Read the questions and put a check mark for the correct answer.

1. Which component ensures that all parts of the computer work together to execute instructions?
   - [ ] a. Arithmetic Logic Unit (ALU)
   - [ ] b. Control Unit (CU)
   - [ ] c. Output devices
   - [ ] d. Main memory

2. What causes the von Neumann bottleneck?
   - [ ] a. The CPU has limited processing power.
   - [ ] b. Data and instructions use separate buses.
   - [ ] c. The CPU can only access data or instructions one at a time.
   - [ ] d. The memory is too large for the CPU to handle.

3. Which component is responsible for decoding instructions in the fetch-execute cycle?
   - [ ] a. Arithmetic Logic Unit (ALU)
   - [ ] b. Main memory
   - [ ] c. Control Unit (CU)
   - [ ] d. Input devices

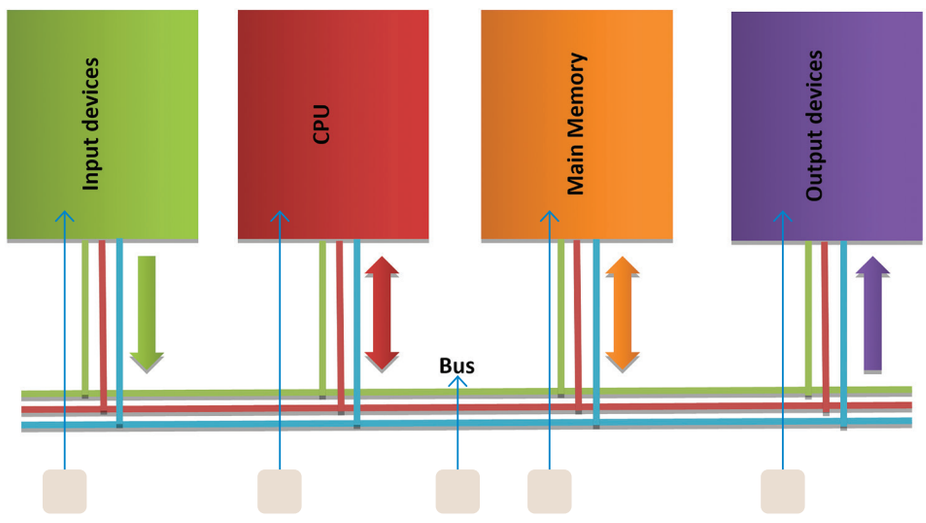
4. Where are the results of executed instructions stored?
   - [ ] a. In the Control Unit (CU)
   - [ ] b. In the Arithmetic Logic Unit (ALU)
   - [ ] c. In the main memory
   - [ ] d. In registers

5. During which phase is the instruction converted into control signals?
   - [ ] a. The fetch phase
   - [ ] b. The decode phase
   - [ ] c. The execute phase
   - [ ] d. The store phase

3. **Read the following sentences and write the number of the correct description.**

1. It stores both data and instructions.
2. Data and instructions are transferred to these devices.
3. Data is transferred through this.
4. Data and instructions are transferred from these devices.
5. It is responsible for the execution of instructions, controls, and system coordination.

Computer Systems Architecture



4. **Describe the steps involved in the fetch-execute cycle, and explain why each step is crucial for a computer to function correctly. What challenges might arise if any of these steps fail, and how could these challenges be addressed?**

5. How might the performance of the fetch-execute cycle be impacted if a program constantly needs to fetch large amounts of data, and what do you think could be done to improve efficiency in this case?

LESSON 2

Storage devices

What types of storage devices are you familiar with?
Why do you think a computer requires different types of storage?

Storage devices are essential components of any computer system, responsible for holding data and programs either temporarily or permanently. They range from the fast, temporary memory used to keep data available for active processes to long-term storage for data that needs to be retained after a computer is turned off. In this lesson, we will explore different types of storage devices, how they work, and the factors that affect their speed and efficiency.

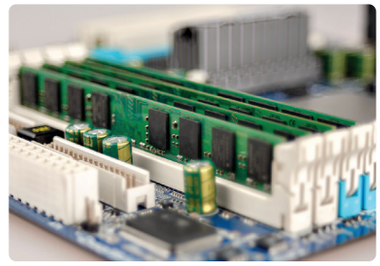
Memory

Random Access Memory (RAM)

The type of main memory that all computers use is called RAM, which stands for Random Access Memory. This type of memory allows for each byte of data to be directly accessed and altered in real time, giving the CPU rapid access to critical data and instructions. RAM is incredibly fast, making it suitable for supporting the smooth operation of processes, but it is volatile. This means its contents are lost the moment the computer is powered off. In contrast, non-volatile memory modules exist that retain their contents even without a constant power supply. A familiar example of non-volatile memory is the flash memory used in USB drives, where data remains intact after the device is unplugged, although data access is much slower.

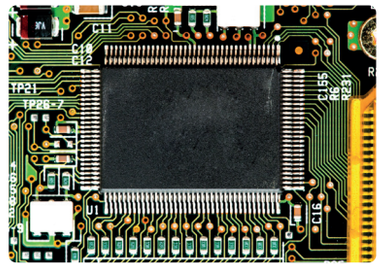
Cache memory

Cache memory acts as a bridge between the ultra-fast CPU and the slower RAM. Cache memory is smaller in capacity but much faster than RAM, storing frequently accessed data and instructions. This reduces the time the processor needs to access the necessary data, improving overall system efficiency. Cache memory ensures that the most critical information is always within quick reach, reducing delays in processing tasks.



Read Only Memory (ROM)

ROM is a type of permanent memory that contrasts with RAM. While RAM is temporary and can be rewritten, ROM stores data permanently, and this data cannot be altered once written. This makes ROM highly suitable for storing crucial system instructions that the computer needs to start up, such as firmware. These instructions remain intact even when the system is powered down, making ROM an essential component in booting up a computer.

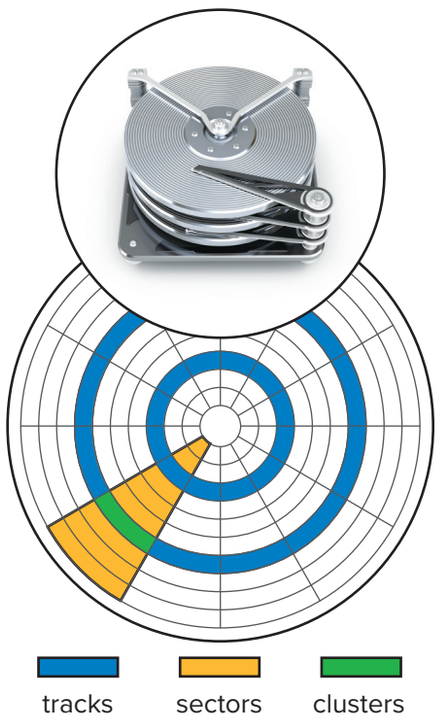


Secondary memory

As we already mentioned, RAM is volatile, so we need another type of storage device where data and instructions can be kept safely when they are no longer being processed or when the computer is turned off. These other types of devices are called secondary storage devices, and the most well-known include the hard disk drive and the CD/DVD drive. Because data must be read from them and written to them, secondary storage devices are considered both input and output devices in the von Neumann architecture.

Hard Disk Drives (HDD)

A desktop computer can be equipped with one or more hard disk drives. The fundamental concept behind HDD operation is the read/write head that moves across spinning magnetic platters to either retrieve (read) or store (write) data. Each platter's surface is organized into concentric **tracks**, which are further divided into **sectors**. A **cluster**, which is a combination of sectors, represents the smallest logical amount of disk space that can be allocated to hold a file on a disk. Data is read from or written to the disk as the read/write head moves to specific tracks and sectors to access the data.



To enhance storage efficiency, HDDs utilize a cylinder-based storage system, where multiple platters are stacked on top of one another. Each platter has its own read/write head, and all the tracks that align vertically across platters are referred to as a cylinder. This configuration allows the HDD to quickly locate and retrieve data from a specific platter, cylinder, and sector combination.

Data access requirements

Different types of software have different requirements for data access. For instance, an HDD used for playing high-definition video must prioritize data transfer speed, as every second of video contains a massive amount of data that must be read quickly. Conversely, a database system requires fast access times to frequently retrieve scattered records stored across the disk.

Seek time is the time it takes for the read/write head to get positioned over the specified track.

Latency is the time it takes for the specified sector to be in position under the read/write head.

Access time is the time it takes for a block to start being read; it is the sum of seek time and latency.

Transfer rate is the rate at which data moves from the disk to the main memory.

Solid State Drives (SSD)

Solid state drives represent a significant advancement in storage technology compared to traditional HDDs. Unlike HDDs, which use spinning magnetic platters and read/write heads, SSDs have no moving parts. Instead, they use flash memory cells to store data electronically. This design allows SSDs to retrieve and write data almost instantly, resulting in much faster access times and significantly improved performance.

Since SSDs use electronic memory, they are inherently more durable than HDDs because they are not affected by physical shocks, making them ideal for laptops and mobile devices.

However, SSDs have a limited lifespan because each flash cell can only handle a finite number of write/erase cycles before it wears out. This limitation is addressed through techniques like wear leveling, which distributes data evenly across cells to extend the lifespan of the drive.

The lack of moving parts not only makes SSDs faster but also quieter and more energy-efficient. This makes them ideal for applications where speed is critical, such as gaming, video editing, and for data-intensive computing tasks.

Advantages of SSDs	
Advantage	Description
Speed	Data access is almost instant, resulting in faster boot times and quicker data retrieval.
Durability	The absence of moving parts makes SSDs resistant to physical shocks and vibrations.
Energy efficiency	SSDs consume less power, making them ideal for battery-powered devices.

Disadvantages of SSDs	
Advantage	Description
Higher cost	SSDs are more expensive per gigabyte compared to traditional HDDs.
Data recovery challenges	Recovering data from a failed SSD is more challenging than from an HDD.

Optical media

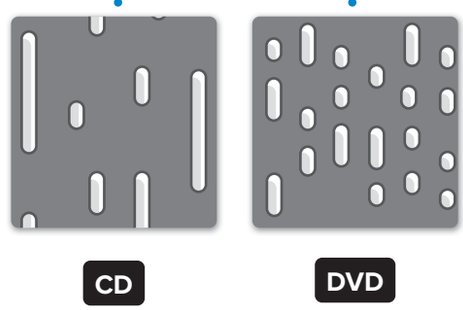
Other secondary storage devices include optical drives like CD/DVD or Blu-Ray drives. Unlike magnetic disks, these devices read and write data optically from a plastic-aluminum disk using a laser beam. In write mode, the laser beam etches a series of **pits** on the aluminum layer as the disk rotates, resulting in a spiral of pits and **lands** which represent binary 0s and 1s. When reading the data, the laser beam bounces off the lands in the aluminum layer but not off the pits, so the binary sequence can be read by a sensor.

These pits are microscopic. On the surface of a Blu-Ray disc, there are about 200 billion pits and lands.

If you happen to come across an old hard disk drive and feel like you want to pop the cover and examine the platters and heads, be warned! All the components inside a hard disk case are carefully aligned and sealed from the outside environment.

A small particle of dust from the air or even a gentle nudge on the heads can make the hard drive unusable! On the other hand, if the drive is already useless and you have permission from the owner, go right ahead!

In these images, we can compare the pits on the surface of a CD (left) to those on a DVD (right), both magnified by a factor of 20,000!



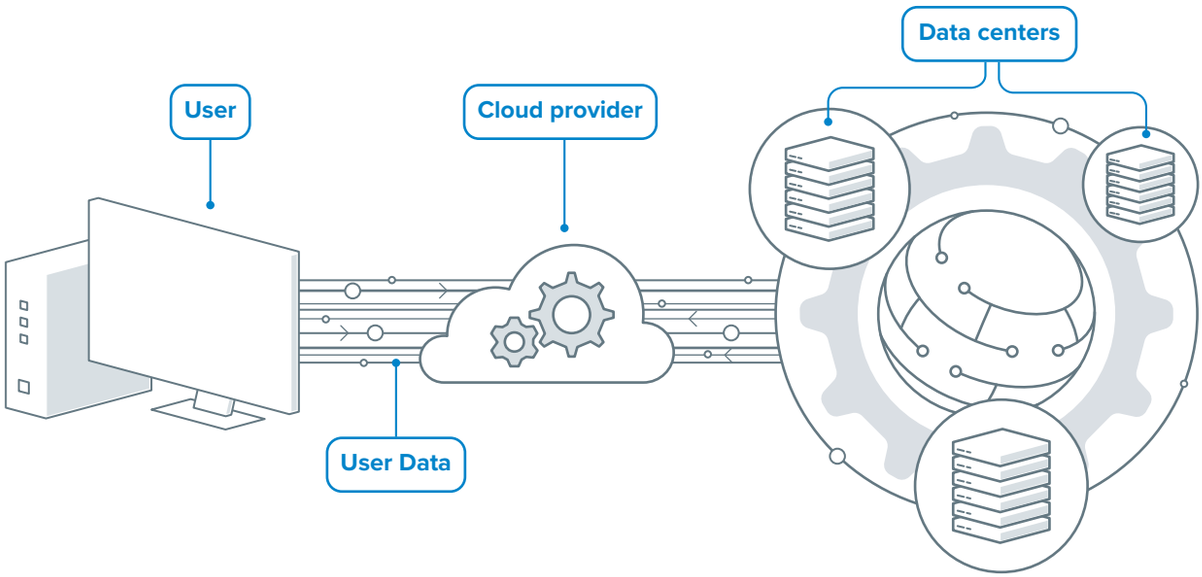
Cloud storage

Cloud storage is a method of storing data on remote servers managed by third-party providers. Unlike local storage (HDDs and SSDs), which stores data directly on a user's device, cloud storage enables access to data over the Internet from any location. When a file is saved to the cloud, it is uploaded to data centers—large facilities equipped with thousands of servers that securely store and manage data for multiple users.

How cloud storage works

Cloud providers, such as Google Drive, Dropbox, or iCloud, store data across multiple servers in data centers. These data centers often use a combination of HDDs and SSDs to provide a balance between speed and capacity. Data is frequently duplicated across multiple servers in different geographic locations to ensure redundancy and data security. This redundancy means that if one server fails, the data is still accessible from another server, making cloud storage highly reliable.

When a user wants to access their data, they connect to the cloud provider's servers through the Internet, and the provider retrieves the data for the user. This process is generally fast, but it depends on the user's Internet speed and the load on the cloud provider's servers. For most applications, this delay is minimal, but large files or frequent data access might be slower than local storage.



Cloud storage is ideal for:

- **Collaborative projects**: Teams can share and edit documents in real time without worrying about local storage limitations.
- **Backups**: Regularly backing up important files to the cloud ensures data can be recovered in case of local hardware failure.
- **Streaming and media**: Media content, such as videos and music, stored in the cloud can be accessed and streamed without occupying local storage, which is particularly useful for mobile devices with limited storage.

Advantages of cloud storage technology	
Advantage	Description
Accessibility	Files stored in the cloud can be accessed from any Internet-connected device, making it convenient for collaboration and remote work.
Scalability	Cloud storage provides virtually unlimited storage capacity, allowing users to expand their storage as needed without investing in additional hardware.
Automatic backup	Cloud storage is often used as a backup solution, as files are stored remotely and redundantly, protecting against data loss from local hardware failure.

Cloud storage considerations

Despite its convenience, cloud storage has limitations. Internet dependency means users need a reliable Internet connection to access their data. Additionally, cloud storage can be costly for large data needs, especially for business users. Data privacy and security are also important considerations, as users must trust that their data is protected on third-party servers. Providers typically offer encryption to safeguard data during transmission and storage, but it's essential to choose reputable services.

Hands on!

1. **Read the following sentences and put a check mark for True or False.**

	True	False
1. RAM is a type of non-volatile memory that retains data even when the computer is turned off.		
2. ROM permanently stores data that cannot be changed after it is written.		
3. Cache memory is slower than RAM but can store more data.		
4. Cache memory improves system efficiency by reducing processing delays.		
5. Hard disk drives (HDDs) store and retrieve data using a read/write head that moves across spinning magnetic platters.		
6. Latency is the time it takes for the read/write head to get positioned over the specified track.		
7. Access time is the sum of seek time and latency.		
8. Transfer rate measures how fast data moves from the hard disk to the main memory.		
9. Recovering data from a failed SSD is simpler than recovering data from an HDD.		
10. Optical media, like CDs and DVDs, store data using a magnetic disk and a read/write head.		

11. The pits and lands on the surface of a Blu-Ray disc represent binary data.

12. Cloud storage provides virtually unlimited storage capacity, allowing users to expand storage without extra hardware.

2. Read the questions and put a check mark for the correct answer.

1. Which type of memory is responsible for storing the critical startup instructions required for a computer to boot up?
   - a. RAM
   - b. Cache memory
   - c. ROM
   - d. Flash memory

2. Which factor affects how quickly the read/write head reaches the correct data track?
   - a. Transfer rate
   - b. Seek time
   - c. Access time
   - d. Latency

3. What is the primary function of the read/write head in a Hard Disk Drive (HDD)?
   - a. To spin the magnetic platters
   - b. To store data permanently
   - c. To move across platters to read or write data
   - d. To cool down the HDD during operation

4. Which of the following is a key advantage of cloud storage?
   - a. It stores data only on one secure server.
   - b. It requires no Internet connection to access data.
   - c. It allows data access from any location with Internet access.
   - d. It is faster than local storage.

5. How do cloud storage providers ensure data security and reliability?

- [ ] a. By storing data only on SSDs
- [ ] b. By encrypting data without backups
- [ ] c. By duplicating data across multiple servers in different locations
- [ ] d. By limiting data access to specific devices

3. How does cache memory improve a computer's performance, and why is it faster than the main memory (RAM)?

4. Why are Solid State Drives (SSDs) generally faster than Hard Disk Drives (HDDs), and what are some advantages and disadvantages of using SSDs over HDDs?

5. Your school is considering moving all student files and documents to cloud storage. Do you think this is a good idea? Explain your opinion by describing both the benefits and the risks, and support your answer with clear reasons and examples.

LESSON 3

Memory and process management

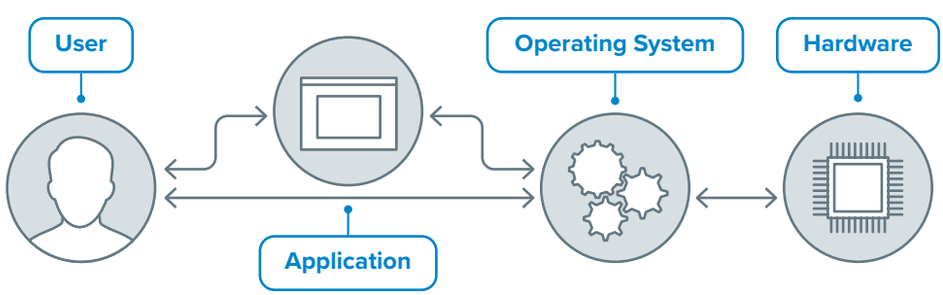
Why do you think memory management is important for a computer to function smoothly?
What problems could occur if multiple programs use the same memory space?

Multiprogramming

You may remember the fetch-execute cycle, which is fundamental to how a CPU processes instructions. When a program is running, it is loaded into the main memory, where the CPU fetches and executes its instructions in sequence. However, in modern computing, this process is more complicated because of the concept of multiprogramming.

Multiprogramming is a technique that allows multiple programs to reside in the main memory simultaneously. This is critical for ensuring that the CPU is utilized efficiently, as it prevents the CPU from being idle while waiting for one program to complete an input/output operation. Instead, the CPU can switch between different programs, giving each one a share of processing time. This makes it possible for users to, browse the Web, listen to music, and run a virus scan simultaneously.

The **operating system (OS)** is responsible for managing multiprogramming. It ensures that each program gets a fair chance to execute and that resources are allocated effectively. This requires careful management of memory and processes, which are fundamental to the OS's role in providing a stable and efficient computing environment.



A multiprogramming example

Let's dive deeper into how multiprogramming works in a real-world scenario. Imagine you are working on a school project using a word processor while also listening to music and downloading a file from the Internet. At first glance, it seems like all these tasks are running simultaneously. However, the CPU is actually switching between these tasks very quickly, giving each one a small amount of processing time in turn. This rapid switching creates the illusion of parallel execution.

Here's a breakdown of how the CPU handles these tasks:

- The word processor might be in the running state as you type. The CPU fetches and executes instructions to display text on the screen.
- The music player might be in the waiting state most of the time, only requiring occasional CPU time to maintain audio playback.
- The download manager may also be in the waiting state, waiting for data packets from the Internet. Once data is received, the CPU processes it and saves it to the hard disk.

This process management is orchestrated by the operating system, which ensures that no single task monopolizes the CPU. By efficiently switching between tasks, the OS maximizes CPU utilization and keeps all active programs responsive.

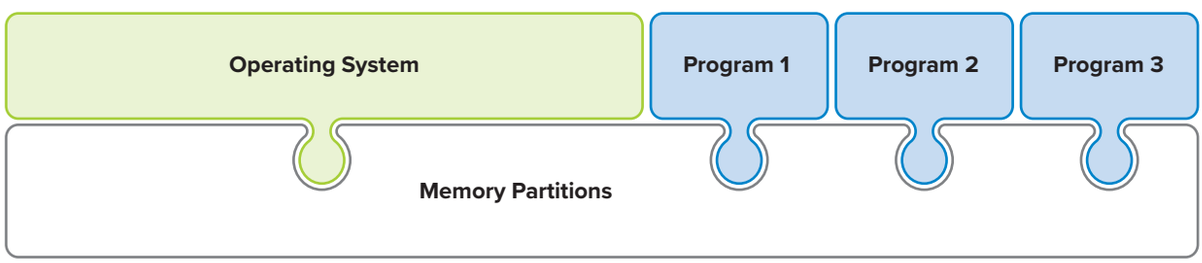
Memory management

Memory management is one of the most crucial tasks performed by the operating system. It involves several key responsibilities to ensure that all programs have the necessary resources to run effectively.

First, the operating system must keep track of where and how programs are located in memory. This is essential because the memory space is shared by multiple programs. The operating system must know which areas of memory are occupied and which are free to ensure efficient memory allocation. Additionally, it must prevent any program from accessing or modifying the memory space allocated to another program, as this could cause data corruption or security issues.

Next, the operating system is responsible for address translation. The main memory is regarded as a continuous sequence of storage locations, each identified by a unique address. However, computer programs use **logical addresses** rather than physical ones. Logical addresses are relative and do not correspond directly to the physical locations in memory. Once the program is loaded into memory, the operating system converts these logical addresses into physical addresses through a process known as **address binding**. This translation is crucial for the program to function correctly, as the CPU accesses instructions and data based on physical addresses.

To illustrate this concept, imagine a library with books arranged in a specific order on shelves. When you search for a book, you may use a reference code (logical address) that the library system translates into the physical location of the book on the shelf. Similarly, the operating system maps logical addresses to physical memory addresses so the CPU can find the data efficiently.



Address binding

Address binding is a crucial aspect of memory management that ensures programs can execute correctly once loaded into memory. As mentioned earlier, programs use logical addresses while they are being developed. However, these logical addresses need to be converted into physical addresses when the program is loaded into the main memory.

This process involves several steps. Initially, the operating system determines where in physical memory the program will be loaded. Once the program's physical location is set, the OS uses address binding to map each logical address to a corresponding physical address. This way, the CPU can execute the program efficiently, knowing exactly where to find instructions and data.

For example, imagine a video editing program being loaded into memory. The software's logical addresses are mapped to physical addresses by the OS. When the user edits a video, the CPU accesses the physical addresses to retrieve video data, apply effects, and render the final video output. Address binding ensures that the program runs smoothly and that the CPU can fetch the necessary data without confusion or errors.

Memory allocation techniques

Memory allocation is an essential task of the operating system, especially in a multiprogramming environment where multiple processes share the main memory. The OS uses different techniques to allocate memory efficiently and minimize issues like fragmentation.

Memory allocation techniques	
Technique	**Description**
Contiguous Memory Allocation	This technique assigns each process a single, contiguous block of memory (using memory addresses that are next to each other on the disk). Although simple, it can lead to external fragmentation, where free memory is scattered in small, unusable blocks between allocated memory partitions.
Non-Contiguous Memory Allocation	To use memory more efficiently, the operating system employs methods like paging and segmentation. In paging, memory is divided into fixed-size divisions (pages) for processes and frames for physical memory, allowing pages to be placed into any available frames and reducing external fragmentation. Segmentation, on the other hand, divides programs into logical segments of varying sizes (such as functions or arrays), offering greater flexibility but still facing challenges with fragmentation.
Virtual Memory	This technique extends physical memory by using disk space as additional memory. Virtual memory allows larger programs to run and increases multitasking capabilities. However, excessive swapping between RAM and disk, called thrashing, can significantly reduce performance.

Fragmentation

A common issue in memory management that occurs when memory space is used inefficiently is fragmentation. There are two main types, external and internal fragmentation. External fragmentation occurs when free memory is scattered across the system in small, non-contiguous blocks, making it challenging to allocate large processes despite having enough total free memory. External fragmentation is a problem in contiguous memory allocation.

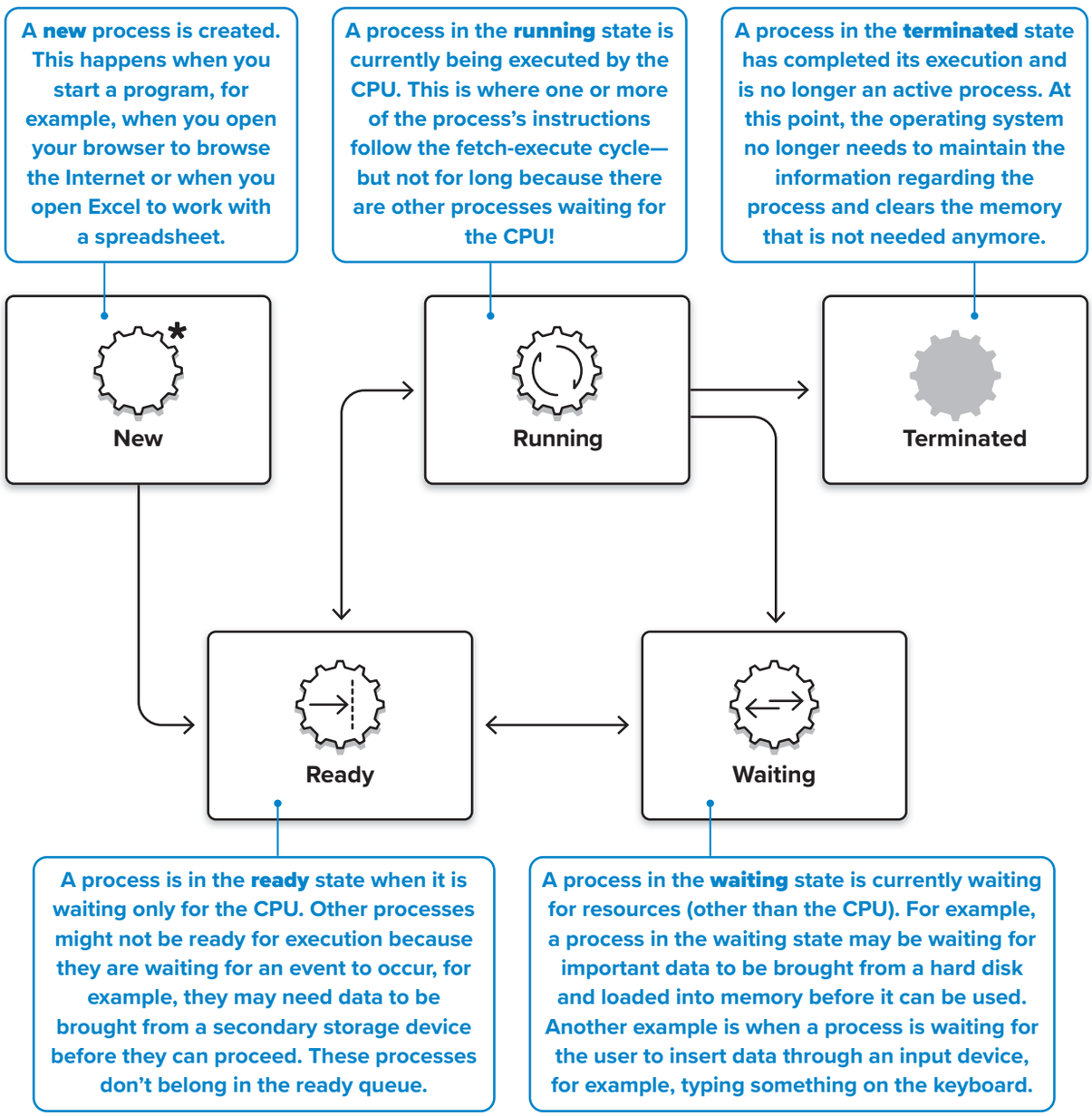
Internal fragmentation happens when allocated memory blocks are larger than the actual memory needed by the process, leaving unused space within the blocks. Internal fragmentation can occur in paging systems where the last page of a process may not be fully utilized.

To reduce fragmentation, operating systems use techniques like compaction (rearranging memory to consolidate free space) or adopt non-contiguous allocation strategies like paging and segmentation.

Process management

The operating system must also manage the use of the CPU by the individual processes. Since only one process can execute part of its instructions at any given time on the CPU, each process goes through a lifecycle of various states as it gains and loses control of the CPU. More specifically, a process enters the system, is ready to be executed, is executing, is waiting for a resource, or is finished.

Let's find out what happens to a process as it goes through each stage.



Smart Tip

Note that many processes may be in the ready state or the waiting state at the same time, but only one process can be in the running state. That's why there is the ready queue and the waiting queue where processes line up to wait in each of these states.

Hands on!

1. **Read the following sentences and put a check mark for True or False.**

	True	False
1. Multiprogramming allows multiple programs to run at the exact same time on the CPU.		
2. The operating system manages multiprogramming by allocating resources fairly to each program.		
3. Logical addresses are the same as physical addresses in memory.		
4. Contiguous memory allocation can lead to external fragmentation.		
5. Non-contiguous memory allocation uses paging and segmentation to reduce fragmentation.		
6. Virtual memory improves performance by using disk space as additional memory.		
7. Segmentation divides programs into fixed-size blocks for memory allocation.		
8. Paging divides memory into fixed-size pages to reduce external fragmentation.		

2. Read the questions and put a check mark for the correct answer.

1. Why is it important for the operating system to track which memory areas are occupied?
   - a. To allow programs to overwrite each other's data
   - b. To prevent the CPU from switching between tasks
   - c. To efficiently allocate memory and prevent data corruption
   - d. To reduce the size of the main memory

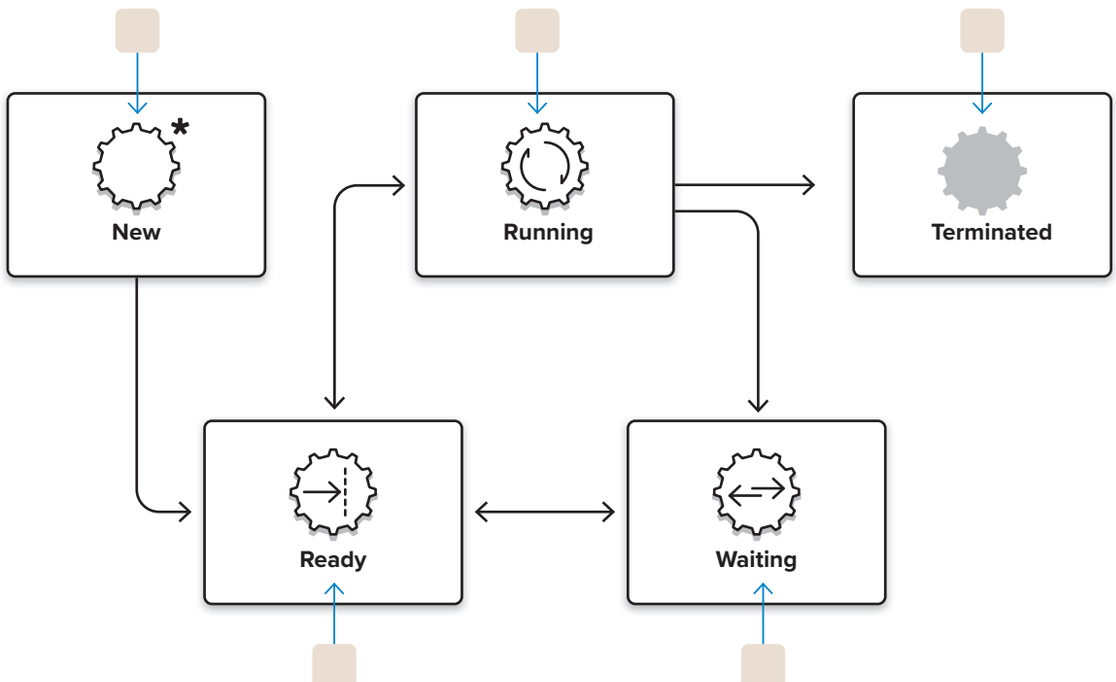
2. What is the process called when the operating system converts logical addresses to physical addresses?
   - a. Memory allocation
   - b. Address binding
   - c. Data encryption
   - d. Paging

3. Which of the following best describes external fragmentation?
   - a. Large memory blocks are split into smaller segments.
   - b. Free memory is scattered in small, unusable blocks throughout the system.
   - c. Programs are stored only in physical memory.
   - d. Processes are divided into equal-sized pages.

4. What is a potential drawback of using virtual memory?
   - a. It requires continuous memory blocks.
   - b. It can cause thrashing due to excessive swapping between RAM and disk.
   - c. It does not support multitasking.
   - d. It cannot be used for large programs.

3. **Read the following sentences and write the number of the correct stage.**

1. A process in this state is currently being executed by the CPU.
2. A process in this state is currently waiting for resources (other than the CPU).
3. A process in this state has completed its execution and is no longer an active process.
4. A process in this state is created when you start a program.
5. A process is in this state when it is simply waiting for the CPU.



4. A program with binary instructions and data is transferred to the main memory. The program uses logical addresses (e.g., 0–9), but physical memory addresses are unordered. If there are enough available physical addresses, the operating system maps each logical address to a physical address. Based on this memory management system, complete the following table.

Logical address
LA–0
LA–1
LA–2
LA–3
LA–4
LA–5
LA–6
LA–7
LA–8
LA–9

Corresponding table	
Logical address	Physical address
0	124

Physical address	
↓	Not available
PA–123	Not available
PA–124	
PA–125	
PA–126	Not available
PA–127	
↓	Not available
PA–534	
PA–535	Not available
PA–536	Not available
PA–537	
PA–538	
PA–539	
↓	Not available
PA–876	
PA–877	
PA–878	Not available
PA–879	
PA–880	Not available
↓	Not available

If PA–537 becomes unavailable due to a hardware failure, how can the operating system use memory allocation techniques to handle the logical address LA–4 and prevent the program from crashing?

5. **If a program is using a lot of memory and CPU, how might this affect other programs? What could the operating system do to manage resources more effectively?**

LESSON 4

File systems

How do you keep your digital files organized? Why is it important to have different permission levels for files and folders, especially on shared computers?

The organization of secondary storage, such as hard disks, is one of the most essential functions of an operating system. Since secondary storage is non-volatile, it retains data even when the computer is turned off, making it the primary location for storing programs and data. Information on a hard disk is organized and stored in **files**, which are named collections of related data and the main organizational units of a hard disk. Files can contain a wide variety of content, including programs, documents, and images. For example, both your web browser application and a digital photo are stored as files on your hard disk.

Directories

To manage these files, the operating system provides a **file system**—a logical structure that enables users to store, organize, and access their data efficiently. A file system organizes files into directories, also known as folders, which act as containers for files and can contain other directories. This structure is often hierarchical, allowing directories to contain multiple subdirectories. A directory that contains another directory is called a **parent directory**, while the directory inside the other is known as a subdirectory. This hierarchical model allows users to organize files in a way that reflects their relationships and purposes. At the very top of this structure is the root directory, from which all other directories branch, creating what is commonly known as a **directory tree**.

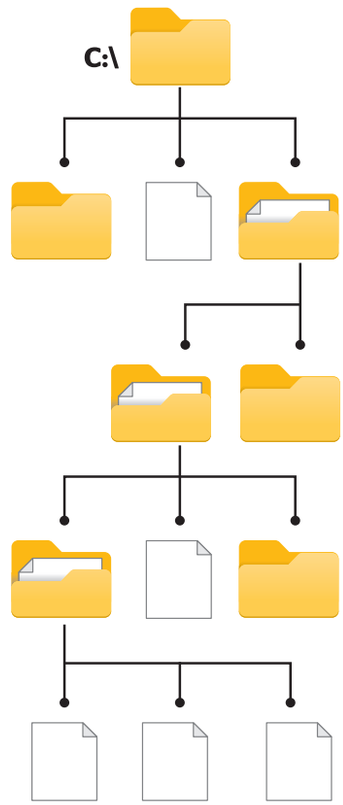
Within this hierarchy, each file and directory has a path—a unique address within the file system that indicates its exact location. There are two types of paths: **absolute paths**, which start from the root directory (e.g., C:\Users\Documents on Windows), and **relative paths**, which are based on the current directory. Paths are crucial for both users and applications to locate files quickly and efficiently, especially in complex directory structures.

Smart Tip

Think of a file system as a digital version of how you organize paper documents in a filing cabinet. Just like you use labeled folders to group related papers and arrange them in drawers, a file system uses directories (or folders) to organize files, making it easier to find what you need quickly.

Hierarchical structure

File systems are commonly visualized as a hierarchical structure, or directory tree, starting from the root directory at the top. The root directory is the highest level of organization, from which all other directories and files stem. Each directory within this structure can contain multiple files or further subdirectories, creating a **branching hierarchy**. This structure is practical for managing large amounts of data, as users can organize files logically and locate data more quickly. To help with navigation, file systems in Windows support shortcuts, which allow users to create pointers to files or directories located elsewhere in the directory tree. This feature is particularly useful for organizing frequently accessed files without duplicating them.



Paths in the directory structure

Paths are also essential within the directory structure, guiding users and applications to the exact location of files. An absolute path starts from the root directory, providing a direct route to the target file or directory, while a relative path starts from the current directory, allowing flexibility in navigation. Through this hierarchical model, file systems provide a logical, organized way to store, retrieve, and secure data, supporting efficient data management in both personal and shared environments.

Types of file systems

Different types of file systems are optimized for specific tasks and environments. Each file system has unique features that make it suitable for particular use cases, and these differences affect how files are stored, accessed, and protected. Some commonly used file systems include:

File system	Description
FAT32 (File Allocation Table)	A simple file system developed by Microsoft, it is widely compatible with many devices and operating systems. It's still commonly used for USB drives and external storage devices due to its cross-platform compatibility, but it has limitations in terms of file and volume size.
NTFS (New Technology File System)	The primary file system for Windows. It offers advanced features like file permissions, encryption, and journaling (a method for improving data integrity). NTFS is designed for large volumes and provides strong security features, making it ideal for modern Windows operating systems.
ext4 (Fourth Extended File System)	The most common file system for Linux operating systems, it is known for its stability and efficiency. It supports large volumes, fast access, and journaling, making it a reliable choice for Linux users.
APFS (Apple File System)	APFS is by macOS operating systems and is optimized for solid-state drives (SSDs). It offers features like fast file cloning and strong encryption, enhancing both performance and data security.

Data integrity

Data integrity, which refers to the protection of data from corruption or loss, is a primary goal of any file system. Several methods are commonly used to achieve this.

Journaling

Journaling is one such technique, employed by file systems like NTFS and ext4. With journaling, the file system records changes in a log or journal before they are written to the disk. If the system crashes mid-operation, the journal allows it to recover to a stable state, reducing the risk of data corruption.

Data redundancy

Another technique for maintaining integrity is data redundancy. Some file systems store multiple copies of crucial metadata, such as the superblock, across the disk. This redundancy is useful in case of metadata corruption, allowing the file system to retrieve backup copies for recovery. **Checksums** are also used to verify data integrity. When data is stored, a checksum (a calculated value) is generated. When the data is accessed, the file system recalculates the checksum to check for any changes, which helps detect and correct errors.

Setting permissions

Permissions and user roles are crucial for data protection. By setting permissions for who can view, modify, or execute a file, the file system helps prevent unauthorized access or accidental deletion. This level of control is especially important in multi-user environments, where different users require varying levels of access to shared files.

Permission types

On storage devices formatted with the NTFS file system, such as the hard drive of a computer running the Windows OS, NTFS permissions are applied to each file and folder. There are several types of NTFS permissions for files and folders.

Permission	Role
Full control	This allows users full control over folders and files, such as reading, writing, modifying, changing permissions, viewing folder/file contents, and deleting them.
Modify	This allows users to read, write, modify, and delete files and subfolders, but not to change permissions.
Read and execute	This allows the user to view files and subfolders and execute executable files (programs). This permission applies the same permissions to files within folders.
List folder contents	This allows users to only view files and subfolders and run executable files in the subfolders. This permission is only applied to subfolders.
Read	This allows users to view files, folders, and subfolders and access their contents.
Write	This allows users to write to files and add files and subfolders within folders.

Modifying permissions

Now it's time to review an example of how to restrict access to a specific folder for a particular user or group by modifying some permissions.

To modify permissions for a specific user:

> Right-click on a file or folder, click **Show more options** 1 and then select **Properties**. 2

> Go to the **Security** tab. 3

> To edit a specific user's permissions, select that user 4, then click on **Edit**. 5

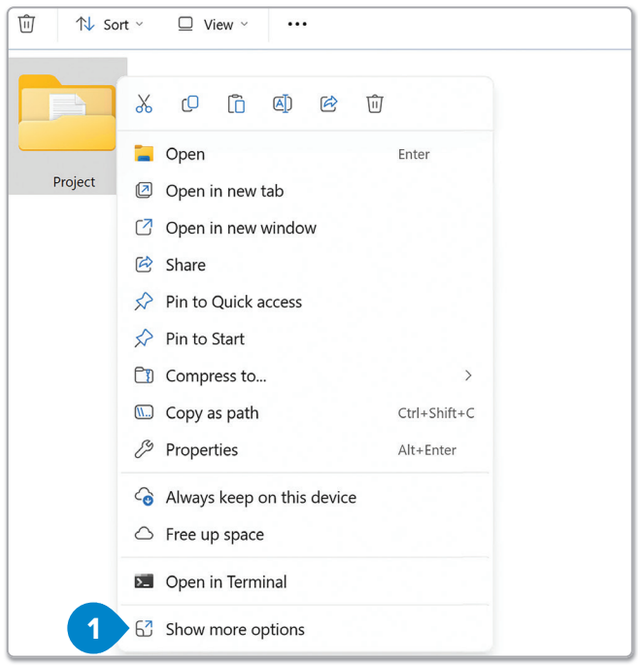
> Click on **Add** to add a user. 6

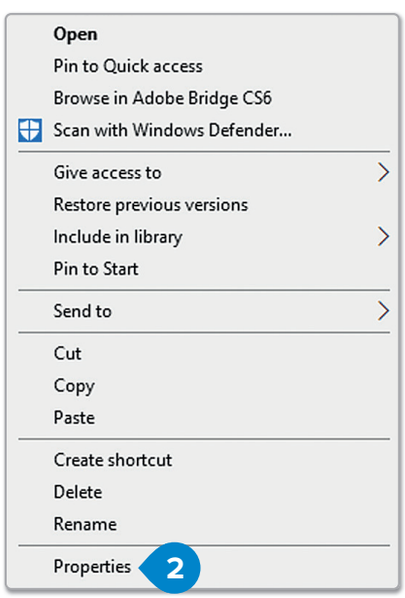
> In the **Select Users or Groups** window, enter the user or group name. 7

> Click on **Check Names** to verify the name is correct. 8

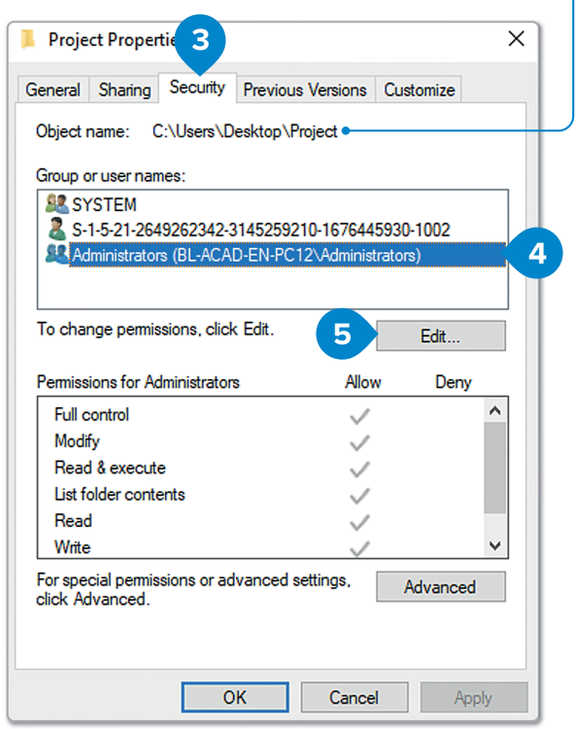
> Click **OK** to add the user or group to the **Access Control** list. 9

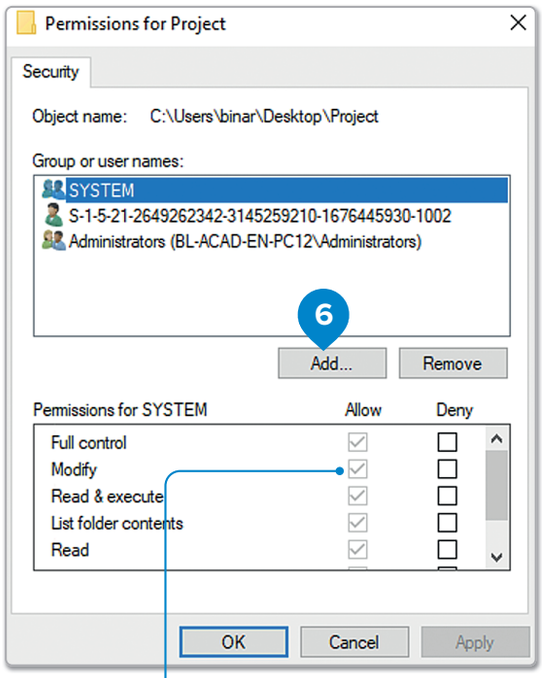
> Now you can choose to **Allow** or **Deny** access. 10



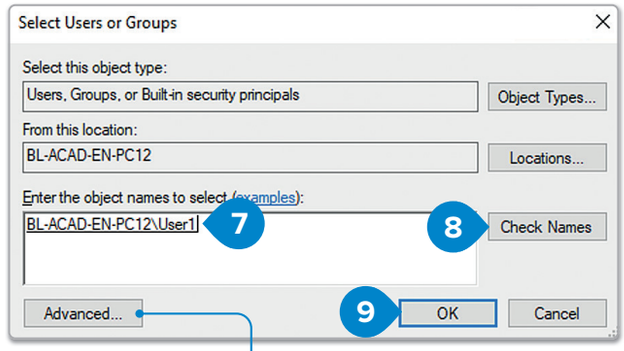


Permissions are inherited in Windows. This means each file or folder inherits permissions from its parent folder, continuing this chain all the way up to the root folder.

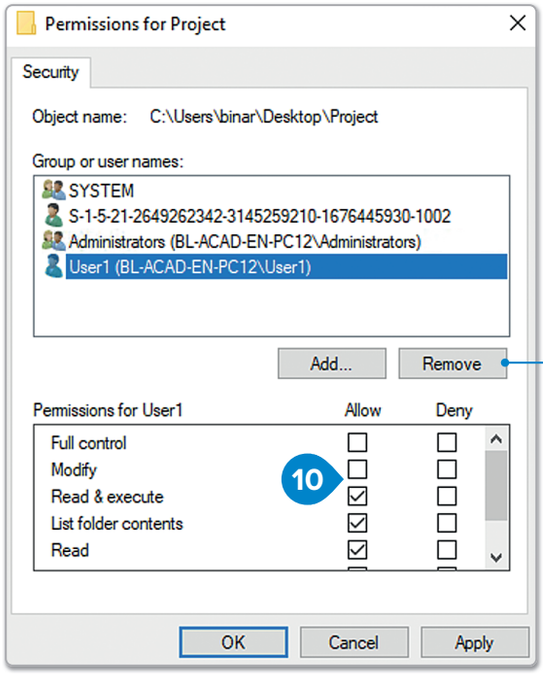




When the Allow column is inactive, it cannot be edited due to inheritance from the root permissions.



If you don't remember the exact username or group name, click on Advanced. Then click on Find Now, which will display all users and groups.



You can remove the added user by selecting them and clicking Remove. However, if you attempt to remove a user who already has inherited permissions, an error message will be displayed.

Smart Tip

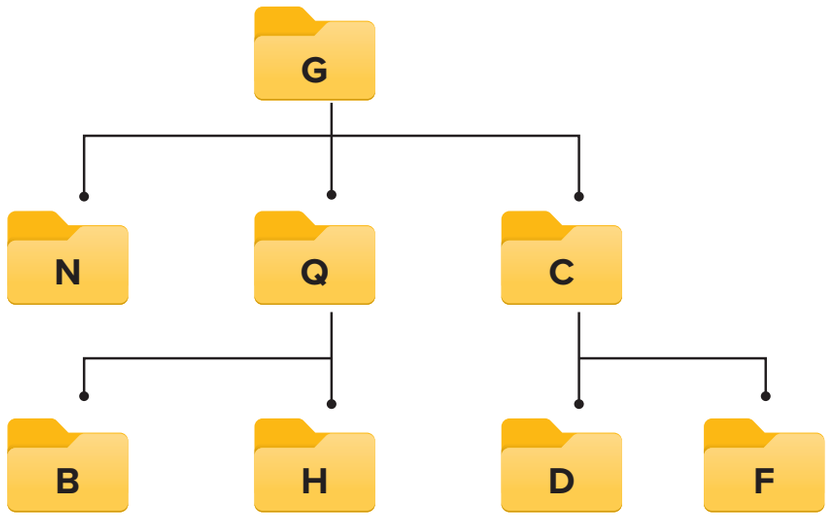
To edit permissions, you must have ownership of the file or folder. If the owner is a different user or a local system account, you won't be able to modify permissions.

Hands on!

1. **Read the following sentences and put a check mark for True or False.**

	True	False
1. A directory that contains another directory is called a parent directory.		
2. The absolute path is based on the current directory.		
3. A file system organizes files into directories to help users store, organize, and access data efficiently.		
4. In a hierarchical file system, the root directory is at the bottom of the directory tree.		
5. Shortcuts in file systems allow users to access files without duplicating them.		
6. The NTFS file system is primarily used in Mac operating systems.		
7. APFS is optimized for solid-state drives (SSDs) and is used by macOS systems.		
8. Journaling helps prevent data loss by recording changes in a log before they are written to the disk.		
9. The "Read" permission allows users to modify and delete files.		

2. **The following diagram depicts the structure of a file system. Put a check mark for the correct answer.**



1. Q is not a
   - a. subdirectory of G
   - b. parent directory of H
   - c. root directory

2. N is a
   - a. root directory
   - b. parent directory of G
   - c. subdirectory of G

3. D has
   - a. two subdirectories
   - b. two parent directories
   - c. no subdirectory

4. B can have the same name
   - a. as H, but not as Q
   - b. as Q, but not as H
   - c. as both Q and H

3. Explain the difference between an absolute path and a relative path in a file system.

4. What does it mean for a file system to have a hierarchical structure, and how do directories (or folders) help manage files?

5. You are working on a group assignment and need to secure the folder called "Class Project" on your computer. Your goal is to set the correct permissions so you have full control over the folder and your classmates have read-only access. Which steps do you need to follow to accomplish this task?

LESSON 5

Access control

What are some ways you protect your devices or accounts so others can't access them?
Do you think adding extra steps to the log in process is useful?

Access control is crucial for maintaining the security of computer systems. It ensures that only authorized users can access certain data or resources, protecting sensitive information, such as personal data, business secrets, or financial records. Without effective access control, systems become vulnerable to unauthorized access, data breaches, and misuse. For example, imagine a hospital's patient database. If access is not restricted, anyone could access or alter medical records, potentially endangering patient safety.

Moreover, access control ensures data integrity by preventing unauthorized modifications, ensures confidentiality by safeguarding sensitive information, and maintains system availability by preventing disruptions or attacks. By controlling who can access what data, organizations can reduce risks and secure their digital infrastructure.

Physical and logical access control

Access control can be divided into two categories, each addressing different aspects of security. **Physical access control** restricts access to physical spaces, like server rooms or data centers, by using locks, badges, or biometric scanners. **Logical access control** manages access to digital resources through software-based methods, like passwords, permissions, and encryption.

In addition to these, administrative measures play a crucial role. Policies and procedures are implemented to ensure that access control mechanisms are effectively managed and maintained. Together, these types of access control form a comprehensive strategy for protecting both physical and digital assets.

Security policies

Security policies are essential rules and guidelines established by organizations to protect their information and resources from unauthorized access and potential threats.

These policies outline how data should be managed, who can access it, and under what conditions access is granted or denied. They serve as a foundation for maintaining a secure environment and ensuring consistency in security practices across the organization.

Access levels

One of the critical functions of security policies is to define access levels. For example, sensitive financial records in a company might only be accessible to the finance department, while other employees are restricted from viewing or modifying this data. By clearly outlining who can access specific resources, security policies help protect data from misuse or accidental alterations.

Password management

Security policies also establish rules for password management. They may require employees to use complex passwords that include a combination of uppercase and lowercase letters, numbers, and symbols. In addition, these policies often enforce regular password changes to reduce the risk of compromised accounts.

Managing data privacy

Managing data privacy is another crucial aspect of security policies, which dictate how sensitive information, like customer addresses or health records, should be encrypted and securely stored to prevent unauthorized access.

Compliance with regulations is another area where security policies are vital. Organizations often need to adhere to laws and standards such as GDPR or HIPAA, which dictate how personal data should be handled. Security policies ensure that these regulations are followed, thereby avoiding legal and financial consequences. Furthermore, these policies provide a framework for incident response, detailing the steps to take in case of a data breach or security incident, such as identifying the breach, reducing the damage, and notifying affected parties if necessary.

Security policies are a cornerstone of access control, defining the rules and procedures that keep systems and data secure. They also provide a basis for auditing and monitoring, ensuring that any suspicious activity is detected and addressed promptly.

Smart Tip

Regularly review and update your security policies to adapt to evolving threats. Outdated policies can leave your system vulnerable to new types of attacks.

User roles and permissions

By assigning specific roles to users, organizations can control access efficiently and reduce the complexity of managing permissions for each individual.

In many systems, users are grouped into predefined roles based on their job functions. For instance, an "administrator" role typically has full access to all resources, including the ability to modify system settings, install software, and manage other users. On the other hand, a "standard user" role might be limited to accessing files, running applications, and modifying personal settings but not making system-wide changes. This separation ensures that only a limited number of trusted personnel have high-level privileges, reducing the risk of accidental or malicious damage to the system.

Permissions specify the exact actions a user can take within a role. Common permissions include Read (viewing data without making changes), Write (modifying or adding data), Execute (running programs or scripts), and Delete (removing data from the system). Combining roles and permissions creates a secure environment where users have only the access necessary to perform their work. This approach helps prevent unauthorized access and minimizes potential damage from insider threats or user errors.

User roles and permissions are also crucial for audit trails and accountability. By logging who accessed or modified data, organizations can track suspicious activities and ensure compliance with security policies. As systems grow in complexity, using well-defined roles and permissions becomes even more important for maintaining a secure and organized environment.

Access control models

There are three primary access control models, each with unique characteristics and use cases.

Model	Characteristics
Discretionary Access Control (DAC)	This allows resource owners, such as users, to decide who can access their files and how. While DAC offers flexibility, it relies heavily on the discretion of individuals, making it potentially less secure if permissions are poorly managed.
Mandatory Access Control (MAC)	MAC is more rigid than DAC. The system enforces strict security policies set by administrators, and users cannot change access permissions. MAC is used in environments requiring high security, like government or military systems, because it ensures consistent, controlled access to resources.
Role-Based Access Control (RBAC)	This simplifies permission management by assigning access rights based on roles within an organization. For instance, an IT administrator may have more privileges than a regular employee. This model is efficient and scalable, especially in large organizations, since permissions are tied to roles rather than individual users.

Authentication

Authentication is the process of verifying that someone is who they claim to be before granting them access to a system or resource. It is a crucial part of access control, as it helps ensure that only authorized individuals can access sensitive information. There are several methods used for authentication, each with its strengths and weaknesses.

Password-based authentication is the simplest and most widely used form. Users must enter a secret password, which is then compared to a stored version to confirm their identity. However, this method has its vulnerabilities, as weak or reused passwords can be smoothly compromised through methods like phishing or brute force attacks. To strengthen password-based systems, organizations often enforce rules such as requiring complex passwords and mandating regular password changes.

Two-factor authentication (2FA) adds an additional layer of security by requiring users to provide two different types of credentials. For instance, after entering a password, users may need to input a code sent to their phone or provide a fingerprint scan. This makes it much harder for attackers to gain access, even if they have the user's password. Biometric authentication, which uses physical characteristics like fingerprints, facial recognition, or iris scans, is becoming more common due to its high level of security. Although effective, it can be costly to implement and raises concerns about privacy and data protection.

Token-based authentication involves the use of a physical or digital token that generates a unique code for each login attempt. This could be a hardware device like a key fob or a smartphone app that provides temporary codes. This method is more secure than relying on passwords alone, as an attacker would need physical possession of the token to gain access. Similarly, smart card authentication requires users to insert a card with embedded information into a reader and enter a PIN. This method is often used in secure environments like corporate offices or government buildings.

Single sign-on (SSO) systems are designed to improve convenience by allowing users to log in once and gain access to multiple applications or systems without needing to re-authenticate. While convenient, SSO systems must be carefully secured, as a compromise of the primary account could give attackers access to all linked resources. Behavioral authentication is a more advanced method that analyzes a user's behavior, such as typing speed or mouse movement patterns, to verify identity. If unusual behavior is detected, the system may require additional verification.

Multi-factor authentication (MFA) combines two or more of these methods to create a robust authentication process. It typically involves something the user knows (like a password), something the user has (such as a smartphone or security token), and something the user is (biometrics). By requiring multiple forms of verification, MFA greatly reduces the risk of unauthorized access, even if one factor is compromised. It is increasingly being adopted in various sectors to secure sensitive information and enhance overall security.

Smart Tip

Enable 2FA wherever possible, especially for sensitive accounts. Even if someone guesses your password, they won't be able to access your data without the second factor.

Hands on!

1. **Read the following sentences and put a check mark for True or False.**

	True	False
1. Access control ensures that only authorized users can access sensitive data and resources.		
2. Logical access control restricts access to physical spaces, like server rooms and data centers.		
3. Passwords and encryption are examples of physical access control.		
4. Mandatory Access Control (MAC) allows users to change their own access permissions.		
5. Two-factor authentication (2FA) requires users to provide two different types of credentials for access.		
6. Single sign-on (SSO) systems eliminate the need for any further security checks after the first login.		
7. Multi-factor authentication (MFA) combines two or more authentication methods to improve security.		
8. Security policies define how data should be managed and who can access it within an organization.		
9. Outdated security policies have no impact on system security.		
10. A "standard user" typically has full access to all system settings and can manage other users.		
11. Role-Based Access Control (RBAC) assigns permissions based on user roles rather than individual users.		

2. Read the questions and put a check mark for the correct answer.

1. What is the primary purpose of access control in computer systems?
   - a. To improve system performance
   - b. To allow public access to sensitive data
   - c. To ensure only authorized users can access specific data or resources
   - d. To automatically back up all data

2. Which access control method protects digital resources using software-based security?
   - a. Physical access control
   - b. Logical access control
   - c. Manual security checks
   - d. A network firewall

3. Which of the following is a key purpose of security policies in an organization?
   - a. To increase employee productivity by limiting Internet use
   - b. To outline rules for managing and protecting sensitive information
   - c. To create public access to company data for transparency
   - d. To eliminate the need for password protection

4. How do security policies help organizations comply with laws like GDPR and HIPAA?
   - a. By allowing open access to all company data
   - b. By defining how sensitive data must be handled and stored securely
   - c. By encouraging employees to share data across departments
   - d. By disabling encryption on sensitive information

5. Which of the following is an example of token-based authentication?
   - a. Entering a password to log in to an account
   - b. Scanning a fingerprint to unlock a device
   - c. Using a smartphone app to generate a temporary login code
   - d. Logging into multiple accounts through Single sign-on (SSO)

3. Describe the difference between authentication and authorization in access control.

4. What are some common authentication methods, and how does multi-factor authentication (MFA) enhance security?

5. Is relying solely on passwords for access control risky? What additional measures could improve security?

Project

Understanding access control

Access control is essential for securing computer systems by regulating who can access data and perform specific actions. Strong access control mechanisms help prevent unauthorized access and protect sensitive information.

1. Create a presentation focused on access control.

In this project, you will work together to create a presentation focused on access control and its role in securing computer systems.

2. The goal

Your goal is to explore how access control protects data and resources within a computer system. Explore information on:

- Authentication methods (like passwords, two-factor authentication, and biometrics).
- Authorization and permissions to control what users can do.
- Different access control models, such as Role-Based Access Control (RBAC) and Mandatory Access Control (MAC).

3. Try to answer the following questions:

- What is access control, and why is it critical for computer security?
- How do authentication and authorization work together to secure a system?
- What are some common threats to access control, and how can they be reduced?

4. Strong access control

Research real-world cases where access control played a key role in preventing or failing to prevent data breaches. Examples will help demonstrate why strong access control is essential.

5. Access control models

Include a section in your presentation to explain access control models, such as Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role-Based Access Control (RBAC), and where each is typically used.

6. Security policies

Devote a section of your presentation to security policies that support access control. Explain how policies define who can access what data and how organizations manage these rules.

Reflect

1. What new information about access control did you find most interesting or important, and why?

2. How do you think different access control methods, like passwords or biometrics, affect security in everyday situations?

3. Which access control model do you think is the most effective, and why?

Wrap up

Take a moment to reflect on your progress.

How confident are you in your ability to apply the following skills?

> I can define and label the components of the von Neumann architecture.

> I can describe the function of the control unit and arithmetic logic unit in processing data.

> I can differentiate between volatile and non-volatile memory and describe how each is used.

> I can explain the structure and operation of HDDs and SSDs, including terminology like tracks, sectors, and clusters.

> I can define multiprogramming and explain how it allows multiple programs to run seemingly simultaneously.

> I can explain how memory management works, including address binding and different memory allocation techniques.

> I can differentiate between various types of file systems.

> I can use permissions to control access to files and folders, and I understand the roles of different permission levels.

> I can explain how access control protects data integrity and prevents unauthorized access.

> I can communicate the importance of security policies in managing access and protecting sensitive information.

Key Terms

access control
address binding
ALU (Arithmetic Logic Unit)
authentication
authorization
binary code
bus
cache memory
cloud storage
contiguous memory allocation
CPU (Central Processing Unit)
CU (Control Unit)
DAC (Discretionary Access Control)
data integrity
directory
FAT32 (File Allocation Table)
fetch-execute cycle
file system
fragmentation
HDD (Hard Disk Drive)
hierarchical structure
input devices
MAC (Mandatory Access Control)
main memory
memory allocation
memory management
multi-factor authentication (MFA)
multiprogramming
NTFS (New Technology File System)
operating system
optical media
output devices
path
permission
physical access control
process management
RAM (Random Access Memory)
RBAC (Role-Based Access Control)
ROM (Read Only Memory)
root directory
secondary memory
security policies
single sign-on (SSO)
solid state drive (SSD)
stored program
transfer rate
virtual memory
von Neumann architecture