Unit 1: Introduction to Operating System

0. What is an Operating System?

Definition:
An **Operating System (OS)** is system software that acts as a **bridge between the user and computer hardware**.

Without an OS, using a computer would be very difficult because users would need to directly control CPU, memory, disk, keyboard, mouse, printer, etc.

Simple Example

When you open **Google Chrome**:

You click the icon → OS loads Chrome from storage → gives memory → gives CPU time → connects internet → displays on screen.

You do not directly talk to CPU or RAM. The OS does it for you.

1.1 What Operating Systems Do

Why do we need an OS?

A computer has hardware:

- CPU
- RAM
- Hard disk / SSD
- Keyboard
- Mouse
- Monitor
- Printer
- Network card

But hardware cannot manage itself. The OS manages all hardware and provides services to users and programs.

Main Functions of an OS

1. Process Management

A **process** is a program in execution.

Example:

- Chrome running
- MS Word running
- YouTube playing
- Antivirus scanning

The OS decides:

- Which process gets CPU
- When to stop a process
- When to start another process

Real-life analogy

CPU is like a **teacher**.
Processes are like **students asking questions**.
The OS is like the **class monitor** deciding whose turn comes next.

2. Memory Management

RAM is limited. The OS manages RAM.

It decides:

- Which program gets memory
- How much memory is given
- When memory is released
- How to protect one program's memory from another

Example:

If Chrome, Word, and Spotify are open, the OS divides RAM among them.

3. File Management

The OS manages files and folders.

It handles:

- Creating files
- Deleting files
- Renaming files
- Saving files

- Organizing folders
- Access permissions

Example:

When you save `assignment.docx`, the OS stores it on disk.

4. Device Management

The OS controls input/output devices.

Examples:

- Keyboard
- Mouse
- Printer
- Scanner
- Speaker
- Camera
- USB drive

The OS uses **device drivers** to communicate with devices.

Driver: A special program that helps the OS control hardware.

5. Security and Protection

The OS protects data and system resources.

Examples:

- Password login
- File permissions
- App permissions
- User accounts
- Antivirus support
- Preventing one app from damaging another app

Example:

On mobile, when WhatsApp asks permission for camera, the OS controls that permission.

6. User Interface

The OS provides a way for users to interact with the computer.

Types:

Interface	Meaning	Example
CLI	Command Line Interface	Linux terminal, Command Prompt
GUI	Graphical User Interface	Windows, macOS, Android

Simple Diagram

```
User
 ↓
Application Programs
 ↓
Operating System
 ↓
Hardware
```

Example:

```
You
 ↓
MS Word / Chrome
 ↓
Windows / Linux / Android
 ↓
CPU, RAM, Disk, Keyboard
```

Short Summary

An OS manages hardware and software resources and makes the computer easy to use.

Exam Points

- OS acts as an interface between user and hardware.
- OS manages CPU, memory, files, devices, and security.
- OS provides a user-friendly environment.

Common Mistakes

- Saying OS is only Windows.

- Forgetting mobile OS examples like Android and iOS.
- Confusing application software with system software.

1.2 Operating System Operations

Meaning

Operating system operations are the internal activities performed by the OS to run the computer safely and efficiently.

1. Booting

Booting means starting the computer and loading the operating system into memory.

Steps of Booting

```
Power ON
 ↓
BIOS/UEFI starts
 ↓
Hardware check
 ↓
Bootloader loads OS
 ↓
OS starts
 ↓
Login screen appears
```

Example:

When you press the power button on your laptop, Windows starts. That process is booting.

2. Program Execution

The OS loads programs into memory and runs them.

Example:

When you double-click VLC:

```
VLC file stored in disk
 ↓
```

```
OS loads VLC into RAM
 ↓
CPU executes VLC instructions
 ↓
VLC opens
```

3. Input/Output Operation

The OS controls input and output.

Examples:

- Reading from keyboard
- Printing document
- Playing sound
- Reading USB drive

The OS hides complex hardware details from the user.

4. Interrupt Handling

An **interrupt** is a signal that tells the CPU that an event needs attention.

Example:

- Keyboard key pressed
- Mouse clicked
- Printer finished task
- Timer signal
- Error occurred

Real-life analogy

A teacher is writing on the board. A student raises hand. The teacher stops and responds. That hand raise is like an interrupt.

5. Error Detection

The OS detects errors such as:

- Memory error
- Disk error

- Printer error
- Network failure
- Illegal instruction

Example:

"Application not responding" is detected by OS.

6. Resource Allocation

The OS distributes resources among programs.

Resources include:

- CPU time
- RAM
- Disk space
- Printer
- Network

Example:

If Chrome and Zoom both need internet, OS manages network access.

7. Protection and Security

The OS prevents unauthorized access.

Example:

One user cannot open another user's private files without permission.

User Mode and Kernel Mode

Modern OS uses two modes:

Mode	Meaning	Example
User Mode	Limited access	Apps like Chrome, Word
Kernel Mode	Full hardware access	OS core/kernel

Why needed?

To protect the system.

If normal apps had full control, they could damage memory, delete files, or crash the computer.

Diagram

```
User Applications
   ↓ User Mode
--------------------
Operating System Kernel
   ↓ Kernel Mode
--------------------
Hardware
```

Short Summary

OS operations include booting, program execution, input/output handling, interrupt handling, error detection, resource allocation, and security.

Exam Points

- Booting loads OS into memory.
- Interrupts help CPU respond to events.
- User mode protects the system from unsafe apps.
- Kernel mode has full hardware access.

Common Mistakes

- Thinking interrupt means error only. It can be normal, like keyboard input.
- Confusing booting with restarting only.
- Forgetting user mode and kernel mode.

1.3 Operating System Services

Meaning

OS services are facilities provided by the OS to users and programs.

Main OS Services

1. User Interface

The OS provides CLI or GUI.

Examples:

- Windows desktop
- Linux terminal
- Android touch interface

2. Program Execution

The OS loads and runs programs.

Example:

Opening calculator, browser, game, or MS Word.

3. I/O Operations

Programs cannot directly control hardware. They request the OS.

Example:

A music app asks OS to play sound through speaker.

4. File System Manipulation

The OS allows:

- Create file
- Delete file
- Read file
- Write file
- Rename file
- Search file

Example:

Saving a photo in gallery.

5. Communication

Processes may communicate with each other.

Types:

Type	Example
Same computer communication	Copy-paste between apps
Network communication	WhatsApp message, email

6. Error Detection

OS detects and handles errors.

Examples:

- Disk failure
- Printer offline
- App crash
- Network error

7. Resource Allocation

OS allocates CPU, RAM, files, and devices.

Example:

During multitasking, OS shares CPU among apps.

8. Accounting

OS keeps records of resource usage.

Example:

- How much CPU used
- How much memory used
- How much disk used

In universities or companies, servers track user activity and usage.

9. Protection and Security

Protection means controlling access to resources.
Security means protecting the system from unauthorized users and attacks.

Examples:

- Password
- Fingerprint
- File permission
- Firewall
- App permission

OS Services Diagram

```
Operating System Services
 ├── User Interface
 ├── Program Execution
 ├── I/O Operations
 ├── File Management
 ├── Communication
 ├── Error Detection
 ├── Resource Allocation
 ├── Accounting
 └── Protection and Security
```

Short Summary

OS services make computers easy, safe, and efficient to use.

Exam Points

Frequently asked question:

“Explain different services provided by operating system.”

Write these:

1. User interface
2. Program execution
3. I/O operation
4. File management
5. Communication

6. Error detection
7. Resource allocation
8. Accounting
9. Protection and security

Common Mistakes

- Mixing OS functions and OS services. They are related but not exactly same.
- Forgetting accounting.
- Writing only Windows examples.

1.4 Operating System Structures

Meaning

Operating system structure means how the OS is designed internally.

Why needed?

Because OS is very large and complex. A good structure makes it:

- Easier to build
- Easier to maintain
- More secure
- More reliable
- Faster

1. Simple Structure

In simple structure, OS components are not clearly separated.

Example

Early MS-DOS.

```
Application Programs
 ↓
System Programs
 ↓
Device Drivers
 ↓
Hardware
```

Advantages

- Simple design
- Fast
- Easy for small systems

Disadvantages

- Poor security
- Difficult to maintain
- One error can crash whole system

2. Monolithic Structure

In monolithic OS, most OS services run in kernel mode.

Examples:

- Traditional UNIX
- Linux kernel style

```
User Programs
 ↓
System Call Interface
 ↓
Kernel:
   File System
   CPU Scheduling
   Memory Management
   Device Drivers
 ↓
Hardware
```

Advantages

- Fast performance
- Direct communication between components

Disadvantages

- Large kernel
- Difficult debugging
- One bug may crash whole OS

3. Layered Structure

The OS is divided into layers. Each layer uses services of the lower layer.

```
Layer 5: User Interface
Layer 4: File Management
Layer 3: I/O Management
Layer 2: Memory Management
Layer 1: CPU Scheduling
Layer 0: Hardware
```

Real-life analogy

A building has floors. Each floor has a purpose.

Advantages

- Easy to debug
- Easy to maintain
- Clear design

Disadvantages

- Slower than monolithic
- Designing layers is difficult

4. Microkernel Structure

In microkernel, only essential services stay in kernel.

Kernel contains:

- Basic memory management
- Basic process communication
- CPU scheduling

Other services run in user space.

```
User Space:
 File System
 Device Drivers
 Network Services

Kernel Space:
 Microkernel
```

```
Hardware
```

Advantages

- More secure
- More reliable
- Easier to extend
- If one service fails, whole OS may not crash

Disadvantages

- Slower communication
- More complex message passing

5. Modular Structure

The OS has a core kernel and loadable modules.

Example:

Linux supports loadable kernel modules.

```
Core Kernel
 ├── File System Module
 ├── Device Driver Module
 ├── Network Module
 └── Security Module
```

Advantages

- Flexible
- Easier to update
- Good performance

Disadvantages

- Module bugs can affect kernel
- Needs careful design

6. Hybrid Structure

Hybrid OS combines different structures.

Examples:

- Windows
- macOS

It uses ideas from monolithic, microkernel, and modular designs.

Advantages

- Good performance
- Flexible
- Practical for modern OS

Disadvantages

- Complex design
- Hard to understand internally

Comparison Table

OS Structure	Speed	Security	Complexity	Example
Simple	Fast	Low	Low	MS-DOS
Monolithic	Very fast	Medium	High	UNIX, Linux
Layered	Medium	Good	Medium	THE OS concept
Microkernel	Slower	High	High	Minix, QNX
Modular	Fast	Good	Medium	Linux
Hybrid	Fast	Good	High	Windows, macOS

Short Summary

OS structure defines how the OS is organized internally. Common structures are simple, monolithic, layered, microkernel, modular, and hybrid.

Exam Points

Frequently repeated question:

“Explain different operating system structures with advantages and disadvantages.”

Must include diagrams.

Common Mistakes

- Saying Linux is purely microkernel. Linux is mainly monolithic with modular features.

- Forgetting disadvantages.
- Drawing unclear diagrams.

1.5 System Call

What is a System Call?

Definition:
A **system call** is a way by which a user program requests a service from the operating system kernel.

Simple meaning:

Apps cannot directly access hardware. They ask the OS using system calls.

Why are System Calls Needed?

Suppose a program wants to read a file.

It cannot directly access the hard disk because that is unsafe.

So it requests the OS:

```
Program: Please open this file.
OS: I will check permission and open it.
```

Real-life Analogy

You are in a restaurant.

You cannot enter the kitchen directly.
You ask the waiter.
The waiter brings food from the kitchen.

```
Customer = User Program
Waiter = System Call
Kitchen = Kernel
Food = OS Service
```

System Call Working

```
User Program
 ↓
System Call
 ↓
Kernel Mode
 ↓
OS performs service
 ↓
Result returned to program
```

Example

When a C program uses:

```
printf("Hello");
```

Internally, it may use a system call to write output to screen.

When a program opens a file:

```
open("data.txt")
read()
write()
close()
```

These are examples of system calls.

Types of System Calls

1. Process Control

Used to create, terminate, and manage processes.

Examples:

- create process
- terminate process
- load program
- wait
- exit

Example:

Opening Chrome creates a process.

2. File Management

Used to manage files.

Examples:

- create file
- open file
- read file
- write file
- close file
- delete file

3. Device Management

Used to access devices.

Examples:

- request device
- release device
- read from device
- write to device

Example:

Printing document.

4. Information Maintenance

Used to get or set system information.

Examples:

- get time
- get date
- get process ID
- get system information

5. Communication

Used for communication between processes.

Examples:

- send message
- receive message
- shared memory
- sockets

Example:

Browser communicating with web server.

6. Protection

Used to control access permissions.

Examples:

- set file permission
- get file permission
- allow or deny access

System Call Types Table

Type	Purpose	Example
Process control	Manage processes	fork, exit
File management	Manage files	open, read, write
Device management	Manage devices	read device, write device
Information maintenance	Get system info	get time, get PID
Communication	Process communication	send, receive
Protection	Security	chmod, permission

Short Summary

System calls are the bridge between user programs and OS kernel services.

Exam Points

- System call allows user program to request OS service.
- It changes execution from user mode to kernel mode.

- Common types: process, file, device, information, communication, protection.

Common Mistakes

- Saying system call is same as function call.
- Forgetting kernel mode transition.
- Not giving examples.

System Call vs Function Call

Basis	Function Call	System Call
Meaning	Calls a function inside program	Requests OS service
Mode	User mode	Switches to kernel mode
Speed	Faster	Slower
Access	No hardware access	Can access hardware through OS
Example	add(), sum()	open(), read(), write()

Important Advantages of Operating System

- Makes computer easy to use
- Manages hardware efficiently
- Supports multitasking
- Provides security
- Manages files and folders
- Controls devices
- Provides user interface
- Detects errors

Disadvantages of Operating System

- Complex software
- Can crash due to bugs
- Requires memory and storage
- Some OS are costly
- Security attacks are possible
- Needs regular updates

Frequently Asked Viva Questions

1. What is an operating system?

An operating system is system software that acts as an interface between user and hardware.

2. Give examples of operating systems.

Windows, Linux, macOS, Android, iOS.

3. What is a process?

A process is a program in execution.

4. What is booting?

Booting is the process of starting a computer and loading the OS into memory.

5. What is a system call?

A system call is a method used by programs to request services from the OS kernel.

6. What is kernel?

Kernel is the core part of the operating system that controls hardware and system resources.

7. What is user mode?

User mode is a restricted mode where normal applications run.

8. What is kernel mode?

Kernel mode is a privileged mode where OS core functions execute.

9. What is the difference between GUI and CLI?

GUI uses graphics and icons, while CLI uses text commands.

10. Why is OS called resource manager?

Because it manages CPU, memory, files, devices, and other resources.

One-Page Quick Revision Sheet

Operating System

OS = Interface between user and hardware

```
User → Application → OS → Hardware
```

Main Functions

- Process management
- Memory management
- File management
- Device management
- Security
- User interface
- Resource allocation

OS Operations

- Booting
- Program execution
- I/O operation
- Interrupt handling
- Error detection
- Resource allocation
- Protection

OS Services

- User interface
- Program execution
- I/O operation
- File manipulation
- Communication
- Error detection
- Resource allocation
- Accounting
- Protection and security

OS Structures

Structure	Example
Simple	MS-DOS
Monolithic	UNIX, Linux
Layered	THE OS concept
Microkernel	Minix, QNX
Modular	Linux modules
Hybrid	Windows, macOS

System Calls

System call = request from program to OS kernel.

Types:

- Process control
- File management
- Device management
- Information maintenance
- Communication
- Protection

OS Functions

- P = Process management
- M = Memory management
- F = File management
- D = Device management
- S = Security
- U = User interface

OS Services

- U = User interface
- P = Program execution
- I = I/O operation
- F = File management
- C = Communication
- A = Accounting
- R = Resource allocation
- E = Error detection

System Call Types

- P = Process control
- F = File management
- D = Device management
- I = Information maintenance
- C = Communication
- P = Protection

2-Mark Questions

1. Define operating system.
2. What is booting?
3. What is a system call?
4. Define kernel.
5. What is process management?
6. What is memory management?
7. What is device driver?
8. What is GUI?
9. What is CLI?
10. What is interrupt?

5-Mark Questions

1. Explain the functions of an operating system.
2. Explain operating system services.
3. Explain system calls with examples.
4. Differentiate between user mode and kernel mode.
5. Explain monolithic OS structure.
6. Explain layered OS structure.
7. Explain microkernel structure.
8. Write advantages and disadvantages of OS.

10-Mark Long Questions

1. Explain what an operating system does with a neat diagram.
2. Explain operating system operations in detail.
3. Explain operating system services with examples.
4. Explain different operating system structures with diagrams, advantages, and disadvantages.
5. Explain system calls, their types, and working mechanism.
6. Compare monolithic, layered, microkernel, modular, and hybrid OS structures.

MCQs with Answers

1. An operating system is:

A. Application software
B. System software
C. Hardware
D. Compiler

Answer: B. System software

2. The main function of an OS is:

A. Playing games
B. Managing resources
C. Designing websites
D. Writing documents

Answer: B. Managing resources

3. A program in execution is called:

A. File
B. Process
C. Folder
D. Command

Answer: B. Process

4. Booting means:

A. Shutting down computer
B. Restarting printer
C. Loading OS into memory
D. Deleting files

Answer: C. Loading OS into memory

5. The core part of OS is called:

A. Shell
B. Kernel
C. Browser
D. Compiler

Answer: B. Kernel

6. Which one is an OS service?

A. File management
B. Cooking
C. Painting
D. Driving

Answer: A. File management

7. System call is used to:

A. Request OS service
B. Open classroom door
C. Format textbook
D. Draw picture

Answer: A. Request OS service

8. Which mode has full hardware access?

A. User mode
B. Kernel mode
C. Safe mode
D. Silent mode

Answer: B. Kernel mode

9. Linux is mainly:

A. Monolithic
B. Only layered

C. Only simple
D. No structure

Answer: A. Monolithic

10. Android is an example of:

A. Operating system
B. Compiler
C. Hardware
D. Spreadsheet

Answer: A. Operating system

Diagram

```
User Applications
   ↓ User Mode
--------------------
Operating System Kernel
   ↓ Kernel Mode
--------------------
Hardware
```

Short Summary

OS operations include booting, program execution, input/output handling, interrupt handling, error detection, resource allocation, and security.

Exam Points

- Booting loads OS into memory.
- Interrupts help CPU respond to events.
- User mode protects the system from unsafe apps.
- Kernel mode has full hardware access.

Common Mistakes

- Thinking interrupt means error only. It can be normal, like keyboard input.
- Confusing booting with restarting only.
- Forgetting user mode and kernel mode.

1.3 Operating System Services

Meaning

OS services are facilities provided by the OS to users and programs.

Main OS Services

1. User Interface

The OS provides CLI or GUI.

Examples:

- Windows desktop
- Linux terminal
- Android touch interface

2. Program Execution

The OS loads and runs programs.

Example:

Opening calculator, browser, game, or MS Word.

3. I/O Operations

Programs cannot directly control hardware. They request the OS.

Example:

A music app asks OS to play sound through speaker.

4. File System Manipulation

The OS allows:

- Create file
- Delete file
- Read file
- Write file
- Rename file

- Search file

Example:

Saving a photo in gallery.

5. Communication

Processes may communicate with each other.

Types:

Type	Example
Same computer communication	Copy-paste between apps
Network communication	WhatsApp message, email

6. Error Detection

OS detects and handles errors.

Examples:

- Disk failure
- Printer offline
- App crash
- Network error

7. Resource Allocation

OS allocates CPU, RAM, files, and devices.

Example:

During multitasking, OS shares CPU among apps.

8. Accounting

OS keeps records of resource usage.

Example:

- How much CPU used
- How much memory used
- How much disk used

In universities or companies, servers track user activity and usage.

9. Protection and Security

Protection means controlling access to resources.
Security means protecting the system from unauthorized users and attacks.

Examples:

- Password
- Fingerprint
- File permission
- Firewall
- App permission

OS Services Diagram

```
Operating System Services
 ├── User Interface
 ├── Program Execution
 ├── I/O Operations
 ├── File Management
 ├── Communication
 ├── Error Detection
 ├── Resource Allocation
 ├── Accounting
 └── Protection and Security
```

Short Summary

OS services make computers easy, safe, and efficient to use.

Exam Points

Frequently asked question:

“Explain different services provided by operating system.”

Write these:

1. User interface
2. Program execution

3. I/O operation
4. File management
5. Communication
6. Error detection
7. Resource allocation
8. Accounting
9. Protection and security

Common Mistakes

- Mixing OS functions and OS services. They are related but not exactly same.
- Forgetting accounting.
- Writing only Windows examples.

1.4 Operating System Structures

Meaning

Operating system structure means how the OS is designed internally.

Why needed?

Because OS is very large and complex. A good structure makes it:

- Easier to build
- Easier to maintain
- More secure
- More reliable
- Faster

1. Simple Structure

In simple structure, OS components are not clearly separated.

Example

Early MS-DOS.

```
Application Programs
 ↓
System Programs
 ↓
```

```
Device Drivers
 ↓
Hardware
```

Advantages

- Simple design
- Fast
- Easy for small systems

Disadvantages

- Poor security
- Difficult to maintain
- One error can crash whole system

2. Monolithic Structure

In monolithic OS, most OS services run in kernel mode.

Examples:

- Traditional UNIX
- Linux kernel style

```
User Programs
 ↓
System Call Interface
 ↓
Kernel:
   File System
   CPU Scheduling
   Memory Management
   Device Drivers
 ↓
Hardware
```

Advantages

- Fast performance
- Direct communication between components

Disadvantages

- Large kernel

- Difficult debugging
- One bug may crash whole OS

3. Layered Structure

The OS is divided into layers. Each layer uses services of the lower layer.

```
Layer 5: User Interface
Layer 4: File Management
Layer 3: I/O Management
Layer 2: Memory Management
Layer 1: CPU Scheduling
Layer 0: Hardware
```

Real-life analogy

A building has floors. Each floor has a purpose.

Advantages

- Easy to debug
- Easy to maintain
- Clear design

Disadvantages

- Slower than monolithic
- Designing layers is difficult

4. Microkernel Structure

In microkernel, only essential services stay in kernel.

Kernel contains:

- Basic memory management
- Basic process communication
- CPU scheduling

Other services run in user space.

```
User Space:
 File System
 Device Drivers
```

```
 Network Services

Kernel Space:
 Microkernel

Hardware
```

Advantages

- More secure
- More reliable
- Easier to extend
- If one service fails, whole OS may not crash

Disadvantages

- Slower communication
- More complex message passing

5. Modular Structure

The OS has a core kernel and loadable modules.

Example:

Linux supports loadable kernel modules.

```
Core Kernel
 ├── File System Module
 ├── Device Driver Module
 ├── Network Module
 └── Security Module
```

Advantages

- Flexible
- Easier to update
- Good performance

Disadvantages

- Module bugs can affect kernel
- Needs careful design

6. Hybrid Structure

Hybrid OS combines different structures.

Examples:

- Windows
- macOS

It uses ideas from monolithic, microkernel, and modular designs.

Advantages

- Good performance
- Flexible
- Practical for modern OS

Disadvantages

- Complex design
- Hard to understand internally

Comparison Table

OS Structure	Speed	Security	Complexity	Example
Simple	Fast	Low	Low	MS-DOS
Monolithic	Very fast	Medium	High	UNIX, Linux
Layered	Medium	Good	Medium	THE OS concept
Microkernel	Slower	High	High	Minix, QNX
Modular	Fast	Good	Medium	Linux
Hybrid	Fast	Good	High	Windows, macOS

Short Summary

OS structure defines how the OS is organized internally. Common structures are simple, monolithic, layered, microkernel, modular, and hybrid.

Exam Points

Frequently repeated question:

“Explain different operating system structures with advantages and disadvantages.”

Must include diagrams.

Common Mistakes

- Saying Linux is purely microkernel. Linux is mainly monolithic with modular features.
- Forgetting disadvantages.
- Drawing unclear diagrams.

1.5 System Call

What is a System Call?

Definition:
A **system call** is a way by which a user program requests a service from the operating system kernel.

Simple meaning:

Apps cannot directly access hardware. They ask the OS using system calls.

Why are System Calls Needed?

Suppose a program wants to read a file.

It cannot directly access the hard disk because that is unsafe.

So it requests the OS:

```
Program: Please open this file.
OS: I will check permission and open it.
```

Real-life Analogy

You are in a restaurant.

You cannot enter the kitchen directly.
You ask the waiter.
The waiter brings food from the kitchen.

```
Customer = User Program
Waiter = System Call
Kitchen = Kernel
Food = OS Service
```

System Call Working

```
User Program
 ↓
System Call
 ↓
Kernel Mode
 ↓
OS performs service
 ↓
Result returned to program
```

Example

When a C program uses:

```
printf("Hello");
```

Internally, it may use a system call to write output to screen.

When a program opens a file:

```
open("data.txt")
read()
write()
close()
```

These are examples of system calls.

Types of System Calls

1. Process Control

Used to create, terminate, and manage processes.

Examples:

- create process
- terminate process
- load program
- wait

- exit

Example:

Opening Chrome creates a process.

2. File Management

Used to manage files.

Examples:

- create file
- open file
- read file
- write file
- close file
- delete file

3. Device Management

Used to access devices.

Examples:

- request device
- release device
- read from device
- write to device

Example:

Printing document.

4. Information Maintenance

Used to get or set system information.

Examples:

- get time
- get date
- get process ID

- get system information

5. Communication

Used for communication between processes.

Examples:

- send message
- receive message
- shared memory
- sockets

Example:

Browser communicating with web server.

6. Protection

Used to control access permissions.

Examples:

- set file permission
- get file permission
- allow or deny access

System Call Types Table

Type	Purpose	Example
Process control	Manage processes	fork, exit
File management	Manage files	open, read, write
Device management	Manage devices	read device, write device
Information maintenance	Get system info	get time, get PID
Communication	Process communication	send, receive
Protection	Security	chmod, permission

Short Summary

System calls are the bridge between user programs and OS kernel services.

Exam Points

- System call allows user program to request OS service.

- It changes execution from user mode to kernel mode.
- Common types: process, file, device, information, communication, protection.

Common Mistakes

- Saying system call is same as function call.
- Forgetting kernel mode transition.
- Not giving examples.

System Call vs Function Call

Basis	Function Call	System Call
Meaning	Calls a function inside program	Requests OS service
Mode	User mode	Switches to kernel mode
Speed	Faster	Slower
Access	No hardware access	Can access hardware through OS
Example	add(), sum()	open(), read(), write()

Important Advantages of Operating System

- Makes computer easy to use
- Manages hardware efficiently
- Supports multitasking
- Provides security
- Manages files and folders
- Controls devices
- Provides user interface
- Detects errors

Disadvantages of Operating System

- Complex software
- Can crash due to bugs
- Requires memory and storage
- Some OS are costly
- Security attacks are possible
- Needs regular updates

Frequently Asked Viva Questions

1. What is an operating system?

An operating system is system software that acts as an interface between user and hardware.

2. Give examples of operating systems.

Windows, Linux, macOS, Android, iOS.

3. What is a process?

A process is a program in execution.

4. What is booting?

Booting is the process of starting a computer and loading the OS into memory.

5. What is a system call?

A system call is a method used by programs to request services from the OS kernel.

6. What is kernel?

Kernel is the core part of the operating system that controls hardware and system resources.

7. What is user mode?

User mode is a restricted mode where normal applications run.

8. What is kernel mode?

Kernel mode is a privileged mode where OS core functions execute.

9. What is the difference between GUI and CLI?

GUI uses graphics and icons, while CLI uses text commands.

10. Why is OS called resource manager?

Because it manages CPU, memory, files, devices, and other resources.

One-Page Quick Revision Sheet

Operating System

OS = Interface between user and hardware

```
User → Application → OS → Hardware
```

Main Functions

- Process management
- Memory management
- File management
- Device management
- Security
- User interface
- Resource allocation

OS Operations

- Booting
- Program execution
- I/O operation
- Interrupt handling
- Error detection
- Resource allocation
- Protection

OS Services

- User interface
- Program execution
- I/O operation
- File manipulation
- Communication
- Error detection
- Resource allocation
- Accounting
- Protection and security

OS Structures

Structure	Example
Simple	MS-DOS

Monolithic	UNIX, Linux
Layered	THE OS concept
Microkernel	Minix, QNX
Modular	Linux modules
Hybrid	Windows, macOS

System Calls

System call = request from program to OS kernel.

Types:

- Process control
- File management
- Device management
- Information maintenance
- Communication
- Protection

Memory Tricks / Mnemonics

OS Functions

PMFDSU

- P = Process management
- M = Memory management
- F = File management
- D = Device management
- S = Security
- U = User interface

OS Services

UP IF CARE

- U = User interface
- P = Program execution
- I = I/O operation
- F = File management
- C = Communication
- A = Accounting

- R = Resource allocation
- E = Error detection

System Call Types

PFDICP

- P = Process control
- F = File management
- D = Device management
- I = Information maintenance
- C = Communication
- P = Protection

2-Mark Questions

1. Define operating system.
2. What is booting?
3. What is a system call?
4. Define kernel.
5. What is process management?
6. What is memory management?
7. What is device driver?
8. What is GUI?
9. What is CLI?
10. What is interrupt?

5-Mark Questions

1. Explain the functions of an operating system.
2. Explain operating system services.
3. Explain system calls with examples.
4. Differentiate between user mode and kernel mode.
5. Explain monolithic OS structure.
6. Explain layered OS structure.
7. Explain microkernel structure.
8. Write advantages and disadvantages of OS.

10-Mark Long Questions

1. Explain what an operating system does with a neat diagram.
2. Explain operating system operations in detail.
3. Explain operating system services with examples.
4. Explain different operating system structures with diagrams, advantages, and disadvantages.
5. Explain system calls, their types, and working mechanism.
6. Compare monolithic, layered, microkernel, modular, and hybrid OS structures.

MCQs with Answers

1. An operating system is:

A. Application software
B. System software
C. Hardware
D. Compiler

Answer: B. System software

2. The main function of an OS is:

A. Playing games
B. Managing resources
C. Designing websites
D. Writing documents

Answer: B. Managing resources

3. A program in execution is called:

A. File
B. Process
C. Folder
D. Command

Answer: B. Process

4. Booting means:

A. Shutting down computer
B. Restarting printer
C. Loading OS into memory
D. Deleting files

Answer: C. Loading OS into memory

5. The core part of OS is called:

A. Shell
B. Kernel
C. Browser
D. Compiler

Answer: B. Kernel

6. Which one is an OS service?

A. File management
B. Cooking
C. Painting
D. Driving

Answer: A. File management

7. System call is used to:

A. Request OS service
B. Open classroom door
C. Format textbook
D. Draw picture

Answer: A. Request OS service

8. Which mode has full hardware access?

A. User mode
B. Kernel mode

C. Safe mode
D. Silent mode

Answer: B. Kernel mode

9. Linux is mainly:

A. Monolithic
B. Only layered
C. Only simple
D. No structure

Answer: A. Monolithic

10. Android is an example of:

A. Operating system
B. Compiler
C. Hardware
D. Spreadsheet

Answer: A. Operating system