**The University of Azad Jammu and Kashmir, Muzaffarabad**



|  |  |
| --- | --- |
| **Submitted to:** | **Engr. Sidra Rafique** |
| **Course Title:** | **Operating System** |
| **Course Code:** | **SE-3205** |
| **Session:** | **2022-26** |
| **Semester:** | **6th** |
| **Roll No:** | **2022-SE-01** |
| **Submitted from:** | **Khurram Farman** |
| **Lab Task No** | **1** |
|  |  |

**Bachelors of Science in Software Engineering (2022-26)**

**Department of Software Engineering**

Contents

[Operating System 3](#_Toc198114416)

[1. Introduction 3](#_Toc198114417)

[Key Functions of an Operating System: 3](#_Toc198114418)

[Types of Operating Systems: 3](#_Toc198114419)

[Examples of Operating Systems: 4](#_Toc198114420)

[Why is the Operating System Important? 4](#_Toc198114421)

[2. CPU Core 4](#_Toc198114422)

[Evolution and History of CPU Cores: 4](#_Toc198114423)

[How Do Multiple Cores Help? 5](#_Toc198114424)

[Logical vs Physical Cores 5](#_Toc198114425)

[3. What is the Difference Between 32-bit and 64-bit OS? 5](#_Toc198114426)

[4. Windows Operating System 6](#_Toc198114427)

[History & Versions Overview 6](#_Toc198114428)

[Key Features of Windows OS 6](#_Toc198114429)

[Editions of Windows 6](#_Toc198114430)

[Security in Windows 7](#_Toc198114431)

[Popular Use Cases 7](#_Toc198114432)

[5. Task Manager 7](#_Toc198114433)

[Key Functions of Task Manager 7](#_Toc198114434)

[How to Open Task Manager? 8](#_Toc198114435)

[Use Cases 8](#_Toc198114436)

[Example Use Scenario 8](#_Toc198114437)

[6. Operating System Services 8](#_Toc198114438)

[Key OS Services 8](#_Toc198114439)

[Why OS Services Matter 9](#_Toc198114440)

[7. Windows vs macOS vs Linux 9](#_Toc198114441)

# Operating System

## Introduction

An **Operating System (OS)** is **system software** that acts as an **interface between the user and the computer hardware**. It manages both the hardware and software resources of a computer system, allowing users and applications to interact with the hardware without needing to know low-level hardware details.

### Key Functions of an Operating System:

|  |  |
| --- | --- |
| Function | Description |
| Process Management | Handles creation, scheduling, and termination of processes. |
| Memory Management | Allocates and deallocates memory space to applications. |
| File System Management | Organizes, stores, retrieves, and manages data on storage devices. |
| Device Management | Controls and communicates with hardware devices via drivers. |
| Security & Access Control | Protects data and system resources from unauthorized access. |
| User Interface (UI) | Provides a command-line or graphical interface for user interaction. |

### Types of Operating Systems:

|  |  |
| --- | --- |
| Type | Description |
| Batch OS | Executes batches of jobs with minimal user interaction (e.g., early IBM systems). |
| Time-Sharing OS | Allows multiple users to use the system at the same time (e.g., UNIX). |
| Distributed OS | Controls a group of computers and makes them appear as a single system. |
| Real-Time OS | Provides quick response times; used in embedded systems like robots, IoT. |
| Network OS | Provides services to computers connected over a network. |
| Mobile OS | Designed specifically for mobile devices (e.g., Android, iOS). |

### Examples of Operating Systems:

* **Windows** – User-friendly GUI-based OS used in PCs.
* **Linux** – Open-source and customizable, widely used in servers.
* **MacOS** – Developed by Apple for Mac computers.
* **Android** – Linux-based mobile OS by Google.
* **IOS** – Apple’s mobile OS for iPhones and iPads.

### Why is the Operating System Important?

* It ensures **efficient utilization** of hardware.
* Enables **user interaction** with the system.
* Manages **applications and system resources**.
* Provides **security, multitasking, and error handling**.

## CPU Core

A **CPU core** is an individual processing unit within a CPU that can **read, execute, and process instructions**. Each core has its own **ALU (Arithmetic Logic Unit)**, **registers**, and **control unit**, enabling it to run its own task independently of other cores.

### Evolution and History of CPU Cores:

|  |  |
| --- | --- |
| ****Time Period**** | ****Development**** |
| **1970s–1980s** | CPUs had a **single core**. Only one instruction was processed at a time. Performance improvements came from increasing clock speed and refining chip design. |
| **Early 2000s** | Physical limitations (like heat and power consumption) made it hard to boost clock speeds further. Chipmakers introduced **dual-core CPUs**, allowing two tasks to run at once. |
| **Mid to Late 2000s** | **Quad-core** and **hex-core** CPUs became common in desktops and laptops. Software began evolving to take advantage of parallel processing. |
| **2010s–Present** | CPUs now feature **8, 12, 16, or more cores** (especially in servers and high-performance systems). Even smartphones today use multi-core processors (e.g., octa-core in Android devices). |

### How Do Multiple Cores Help?

* **Multitasking**: Run several applications at once without slowdowns.
* **Parallel Processing**: Divide complex tasks (like video rendering or data analysis) across cores for faster performance.
* **Better Power Efficiency**: Lower clocked multi-core CPUs can do more with less heat and power than a single high-clocked core.

### Logical vs Physical Cores

* **Physical Core**: A true, separate processor unit on the chip.
* **Logical Core**: Created by technologies like **Hyper-Threading** (Intel) or **SMT** (Simultaneous Multi-Threading) in AMD, where a single core handles two threads, appearing as two "logical" cores to the OS.

## What is the Difference Between 32-bit and 64-bit OS?

|  |  |  |
| --- | --- | --- |
| Feature | **32-bit OS** | **64-bit OS** |
| **Processor Architecture** | Uses 32-bit CPU architecture | Uses 64-bit CPU architecture |
| **Memory (RAM) Support** | Can address up to **4 GB** of RAM only | Can address **more than 18 quintillion bytes** (theoretical limit); practically **up to 2 TB+** |
| **Data Handling** | Processes **32 bits** of data at a time | Processes **64 bits** of data at a time |
| **Performance** | Lower performance, suitable for basic tasks | Higher performance, better for multitasking and heavy applications |
| **Software Compatibility** | Can run only **32-bit applications** | Can run both **32-bit and 64-bit applications** (if supported) |
| **Security Features** | Fewer modern security features | Supports advanced security features (e.g., DEP, ASLR) |
| **Use Cases** | Older PCs, basic office tasks | Modern PCs, gaming, development, virtualization, etc. |

## Windows Operating System

**Windows OS** is a **graphical operating system** developed by **Microsoft**, first released in **1985**. It provides a **user-friendly interface**, allowing users to perform tasks like file management, internet browsing, software installation, gaming, and more.

### History & Versions Overview

|  |  |  |
| --- | --- | --- |
| Version | Release Year | Key Features |
| **Windows 1.0** | 1985 | First GUI-based Microsoft OS |
| **Windows 95** | 1995 | Start menu, taskbar introduced |
| **Windows XP** | 2001 | Stability, wide adoption |
| **Windows 7** | 2009 | Speed, performance, refined UI |
| **Windows 8/8.1** | 2012-13 | Metro UI, touchscreen support |
| **Windows 10** | 2015 | Universal apps, Cortana, security updates |
| **Windows 11** | 2021 | Redesigned UI, Snap layouts, improved multitasking, TPM 2.0 requirement |

### Key Features of Windows OS

* **Graphical User Interface (GUI)**: Easy-to-use desktop environment.
* **File Management**: Explorer, folders, drives system.
* **Multitasking**: Run multiple apps simultaneously.
* **Security**: Windows Defender, BitLocker, regular patches.
* **Compatibility**: Supports most hardware and software globally.
* **Remote Desktop & Networking**: For professional and enterprise users.
* **Microsoft Store**: Install apps directly from a centralized platform.

### Editions of Windows

* **Home** – For regular users.
* **Pro** – Advanced features for professionals.
* **Enterprise** – Corporate-grade features.
* **Education** – For schools and universities.

### Security in Windows

* Built-in antivirus: **Microsoft Defender**
* Firewall & SmartScreen
* Windows Hello (face/fingerprint login)
* Regular **security updates**

### Popular Use Cases

* Personal computing
* Office work
* Gaming (DirectX support)
* Software development
* Enterprise and IT environments

## Task Manager

**Task Manager** is a built-in utility in **Windows Operating Systems** that allows users to monitor and manage system performance, applications, processes, and services.

### Key Functions of Task Manager

|  |  |
| --- | --- |
| Tab | Description |
| **Processes** | Displays all running applications and background processes along with their CPU, memory, disk, network, and GPU usage. |
| **Performance** | Shows real-time graphs and stats for CPU, memory (RAM), disk, Ethernet/Wi-Fi, and GPU usage. |
| **App History** | Tracks CPU time and network usage per app (mainly for UWP apps). |
| **Startup** | Lists apps that start with Windows and allows users to **enable/disable** them to optimize boot time. |
| **Users** | Displays all active users and their resource usage. |
| **Details** | A detailed view of each process with PID, status, and resource consumption. |
| **Services** | Shows Windows services, their status (Running/Stopped), and allows starting/stopping them. |

### How to Open Task Manager?

* **Ctrl + Shift + Esc**
* **Right-click** on the taskbar → Task Manager
* **Ctrl + Alt + Delete** → Select Task Manager

### Use Cases

* Kill **unresponsive programs**
* Check **resource-heavy apps**
* Monitor **performance bottlenecks**
* Disable **startup programs**
* Start/stop **Windows services**
* Check system health in real-time

### Example Use Scenario

If your system is **slow**, you can:

1. Open Task Manager.
2. Go to the **Processes** tab.
3. Sort by **CPU** or **Memory** usage.
4. Identify and end the high-resource-consuming task.\

## Operating System Services

An **Operating System (OS)** provides various **services** that act as a bridge between the **hardware** and **user applications**, ensuring the system functions smoothly and securely.

### Key OS Services

|  |  |
| --- | --- |
| ****Service**** | ****Purpose**** |
| **Program Execution** | Loads programs into memory and executes them. |
| **I/O Operations** | Manages input/output devices like keyboard, mouse, printer, etc. |
| **File System Manipulation** | Allows programs and users to read/write, create/delete, and organize files. |
| **Communication** | Enables communication between processes (Inter-Process Communication – IPC). |
| **Error Detection** | Detects and reports system errors for smoother recovery. |
| **Resource Allocation** | Manages hardware resources like CPU time, memory, and disk space. |
| **Security & Protection** | Prevents unauthorized access and protects data integrity. |
| **User Interface** | Provides ways for users to interact with the system (CLI, GUI, etc.). |
| **Networking** | Allows systems to connect and communicate over networks. |
| **Accounting** | Tracks resource usage for auditing or billing in multi-user environments. |

### Why OS Services Matter

These services:

* Simplify application development by handling hardware-level tasks.
* Ensure fair and efficient resource usage.
* Maintain system stability and performance.
* Provide a secure and user-friendly environment.

## Windows vs macOS vs Linux

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Windows | macOS | Linux |
| Developer | Microsoft | Apple Inc. | Open-source community (e.g., Linux Foundation) |
| First Release | 1985 (Windows 1.0) | 1984 (as Macintosh System) | 1991 (Linux kernel by Linus Torvalds) |
| Kernel Type | Hybrid Kernel | Hybrid (based on Unix, XNU kernel) | Monolithic (also supports modular kernel) |
| Source Code | Closed-source (proprietary) | Closed-source (with some open components) | Open-source |
| System Architecture | Supports 32-bit & 64-bit | Primarily 64-bit | Supports 32-bit, 64-bit, and ARM |
| File System | NTFS, exFAT, FAT32 | APFS, HFS+ | ext4, Btrfs, XFS, etc. |
| Security | Moderate (frequent target of malware) | High (controlled app ecosystem) | Very high (user-controlled, fewer threats) |
| Customization | Limited (GUI & features locked down) | Very limited (Apple controls UX) | Highly customizable (themes, UI, kernel, etc.) |
| User Interface | GUI (Start Menu, taskbar, File Explorer) | GUI (Dock, Finder, Spotlight) | GUI (GNOME, KDE, XFCE, etc.) or CLI |
| Command Line | CMD / PowerShell | Terminal (zsh/bash) | Terminal (bash, zsh, fish, etc.) |
| Software Support | Best (widest compatibility) | Limited to macOS apps | Moderate (depends on distro, supports Wine) |
| Gaming | Excellent (DirectX support, wide range) | Poor (few games supported) | Fair (growing via Steam Proton & Lutris) |
| Hardware Compatibility | Supports wide range of hardware | Only works on Apple hardware | Broad support (desktop/server/embedded) |
| Use Case Focus | General-purpose, business, gaming | Creative professionals (design, media) | Developers, servers, enthusiasts |
| System Updates | Frequent (Windows Update) | Controlled and stable | Depends on distro (user-controlled) |
| Cost | Paid (licensed per device) | Comes with Apple devices (indirectly paid) | Free |
| Support & Community | Microsoft support, large user base | Apple support, limited user community | Massive community, forums, extensive online help |
| Resource Requirements | High (more RAM/CPU for smooth use) | Moderate to high | Low to high (depends on distro) |