CS 4310 Operating Systems

Homework 1

Prof. Gilbert Young

Due Date: October 21, 2025

Problem 1 (10 points) Conduct a research and give the top five most popular operating systems of the world. Justify your answer and explain why they are the most popular.

The five most popular operating systems widely used right now (across desktops, laptops, and phones) are:

1. **Microsoft Windows:** On desktops/laptops, Windows remains dominant approximately 70.3% of desktop OS usage in 2025. Windows has been around for decades on PCs and has an enormous installed base, while many businesses, educational institutions, and consumers rely on it. The software contains a massive library of desktop applications (productivity, games, enterprise tools) that support Windows strongly. Nearly all PC-hardware vendors support Windows, making it widely available. In enterprise offices, schools, etc., Windows is typically the default environment with its Active Directory service, which maintains its large share.

2. **macOS:** macOS is the main desktop OS from Apple. On desktops/laptops, it holds a higher share in certain industries and regions (especially where Apple hardware is popular). macOS runs on MacBooks and desktops which have a solid reputation for design, reliability, and integration with the Apple ecosystem. macOS is favored in creative industries (video, music, graphics) and among developers, reinforcing its status. Many users perceive macOS as more secure or stable (rightly or wrongly) than some other options, which attracts a dedicated user base. Because many Mac users pay premium prices, Apple focuses on quality and experiences, which helps maintain loyalty.

3. **Linux:** Even though Linux desktop share (for general consumer desktops/laptops) remains modest compared to Mac and Windows, it is widely important for servers, cloud infrastructure, supercomputing, and embedded systems where stability, security, or customization are key. Linux distributions often cost nothing and allow modification, which appeals to tech-savvy users, developers, and organizations seeking control. While not dominating consumer desktops, Linux powers a large majority of servers and is the backbone of internet infrastructure. Users can choose from many Linux flavors (Ubuntu, Fedora, Debian, etc.), which keeps a strong niche and ecosystem alive.

4. **iOS:** iOS is the main non-Android mobile OS with about $28 - 30\%$ global share across all mobile devices. Apple's ecosystem gives iOS a premium image and high loyalty. iOS offers more integration with Apple controlling both hardware and software, offering optimized performance, user experience, and strong support. iOS tends to get new apps/features earlier and has strong support for software updates across devices for many years. In certain developed markets, iOS has a large penetration, which boosts its global visibility.

5. **Android:** Android is often cited as the world's most popular installed OS base with about 73.9% global share across all mobile devices. Android runs on a vast number of phone and tablet models from many manufacturers, including budget devices in emerging markets. The Google Play ecosystem and Android's flexibility mean it covers many use-cases. Its market reach is particularly popular in Asia, Africa, Latin America and regions with growing smartphone penetration. Most new smartphones (outside of iOS) ship with Android, giving it a large pre-installed base advantage. It spans phones, tablets, and many smart devices, which broadens the usage base beyond just PCs.

Problem 2 (10 points) Which of the following instructions should be allowed only in kernel mode?

(a) Disable all interrupts.

(b) Change the time-of-day clock.

(c) Read the time-of-day clock.

(d) Clear the memory.

- **(a) Disable all interrupts:** Interrupts control how the CPU responds to events. Allowing user programs to disable interrupts could let them control the CPU, effectively freezing the system. Therefore, this is a privileged instruction that must be restricted to the kernel.
- **(b) Change the time-of-day clock:** The system clock affects all system processes and timestamps. Allowing a user process to change it could disrupt scheduling, logging, and security mechanisms. So modifying the system clock should be restricted to the kernel.
- **(d) Clear the memory:** Clearing (or overwriting) memory could affect other processes or the OS itself. This operation must be managed by the OS to maintain isolation and protection.

Problem 3 (10 points) Consider a computer system that has cache memory, main memory (RAM) and disk, and the operating system uses virtual memory. It takes 2 nsec to access a word from the cache, 10 nsec to access a word from the RAM, and 20 ms to access a word from the disk. If the cache hit rate is 90% and main memory hit rate (after a cache miss) is 98%, what is the average time to access a word?

Table 1: Memory Hierarchy Performance Metrics

Component	**Access Time**	**Hit Rate**	**Miss Rate**	**Notes**
Cache	2 ns	0.90	0.10	—
RAM	10 ns	0.98	0.02	*(after a cache miss)*
Disk	$20\,\text{ms} = 2 \times 10^7\,\text{ns}$	—	—	*(after cache & memory miss)*

Expected access time calculation:

$$\begin{aligned} T &= (0.90)(2\,\text{ns}) + (0.10)(0.98)(10\,\text{ns}) + (0.10)(0.02)(2 \times 10^7\,\text{ns}) \\ &= 1.8\,\text{ns} + 0.98\,\text{ns} + 40{,}000\,\text{ns} \\ &= 40{,}002.78\,\text{ns} \end{aligned}$$

The average time to access a word is about $40{,}002.78\,\text{ns} \approx 40\,\mu\text{s}$.

Problem 4 (10 points) In the figure, three process states are shown. In theory, with three states, there could be six transitions, two out of each state. However, only four transitions are shown. Are there any circumstances in which either or both of the missing transitions might occur?

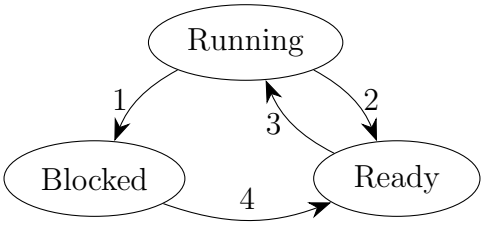

Figure 1: Process State Diagram

Currently, the missing transitions in the process state diagram are Blocked → Running and Ready → Blocked.

- **Blocked → Running:** When the blocking event completes (e.g., I/O finishes) and the just-unblocked process has higher priority than the current one, a preemptive scheduler may immediately dispatch it. That makes the transition effectively Blocked → Running without first sitting in Ready.
- **Ready → Blocked:** Normally only a running process can block, so this transition doesn't happen in the simple model. But it can if the OS performs pre-dispatch resource checks (e.g., verifies required memory pages, locks, or other resources before giving the CPU) and finds something unavailable. The scheduler may then place the process directly into Blocked to wait for the resource/event.

Problem 5 (10 points) What is the biggest advantage of implementing thread in kernel space? What is the biggest disadvantage?

Biggest Advantage (Kernel-Level Threads)

The biggest advantage of kernel-level threads is true concurrency and better CPU utilization. The operating system's scheduler knows about each thread. It can schedule multiple threads from the same process on different CPUs or cores simultaneously, achieving real parallelism. If one thread blocks (e.g., waiting for I/O), the kernel can still run another thread from the same process, which improves responsiveness and throughput.

Biggest Disadvantage (Kernel-Level Threads)

The biggest disadvantage of kernel-level threads is higher overhead and slower context switching. Because the kernel manages each thread, every thread operation (creation, destruction, synchronization, scheduling) involves system calls and mode switches between user and kernel mode. This increases overhead compared to user-level threads, where thread management happens entirely in user space.

Problem 6 (10 points) Five batch jobs A through E, arrive at a computer center at almost the same time. They have estimated running times of 7, 6, 3, 5, and 9 minutes. Their (externally determined) priorities are 3, 2, 1, 5, and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the mean process turn-around time. Ignore process switching overhead.

(a) Round robin (with time slice of 2 minute).

(b) Priority scheduling.

(c) First-come, first served (run in order A, B, C, D, E)

(d) Shortest job first.

Table 2: Batch Job Running Times and Priorities

Batch Job	**Running Time (minutes)**	**Priority**
A	7	3
B	6	2
C	3	1
D	5	5
E	9	4

(a) **Round Robin:** The mean turnaround time is $\frac{(15+23+24+27+30)}{5} = 23.8$ minutes.

Table 3: Round Robin Schedule Table

Batch Job	**Start Time**	**End Time**	**Job Completion**
A	0	2	—
B	2	4	—
C	4	6	—
D	6	8	—
E	8	10	—
A	10	12	—
B	12	14	—
C	14	15	Job C completed @ 15
D	15	17	—
E	17	19	—
A	19	21	—
B	21	23	Job B completed @ 23
D	23	24	Job D completed @ 24
E	24	26	—
A	26	27	Job A completed @ 27
E	27	29	—
E	29	30	Job E completed @ 30

(b) **Priority Scheduling:** The mean turnaround time is $\frac{(5+14+21+27+30)}{5} = 19.4$ minutes.

Table 4: Priority Scheduling Schedule Table

Batch Job	**Start Time**	**End Time**	**Job Completion**
D	0	5	Job D completed @ 5
E	5	14	Job E completed @ 14
A	14	21	Job A completed @ 21
B	21	27	Job B completed @ 27
C	27	30	Job C completed @ 30

(c) **First-Come First-Serve:** The mean turnaround time is $\frac{(7+13+16+21+30)}{5} = 17.4$ minutes.

Table 5: First-Come First-Serve Schedule Table

Batch Job	**Start Time**	**End Time**	**Job Completion**
A	0	7	Job A completed @ 7
B	7	13	Job B completed @ 13
C	13	16	Job C completed @ 16
D	16	21	Job D completed @ 21
E	21	30	Job E completed @ 30

(d) **Shortest Job First:** The mean turnaround time is $\frac{(3+8+14+21+30)}{5} = 15.2$ minutes.

Table 6: Shortest Job First Schedule Table

Batch Job	**Start Time**	**End Time**	**Job Completion**
C	0	3	Job C completed @ 3
D	3	8	Job D completed @ 8
B	8	14	Job B completed @ 14
A	14	21	Job A completed @ 21
E	21	30	Job E completed @ 30

Problem 7 (10 points)

(a) A RAID can fail if two or more of its drives crash within a short time interval. Suppose that the probability of one drive crashing in a given hour is C. What is the probability of a k-drive RAID failing in a given hour?

(b) A RAID can fail if three or more of its drives crash within a short time interval. Suppose that the probability of one drive crashing in a given hour is C. What is the probability of a k-drive RAID failing in a given hour?

(a) **RAID fails if two or more drives crash:** Each drive fails independently with probability C per hour. We want

$$P(\text{RAID fails}) = P(\text{at least 2 drives fail}),$$

which is the complement of 0 or 1 drive failing:

$$P_{\text{fail}} = 1 - [P(0 \text{ fails}) + P(1 \text{ fails})].$$

Using the binomial distribution:

$$\begin{aligned} P(0 \text{ fails}) &= (1-C)^k \\ P(1 \text{ fails}) &= kC(1-C)^{k-1} \end{aligned}$$

Therefore, the probability of a k-drive RAID failing in a given hour is

$$P_{\text{fail}} = 1 - (1-C)^k - kC(1-C)^{k-1}.$$

(b) **RAID fails if three or more drives crash:** We can use the same idea, but excluding the cases of 0, 1, or 2 drive failures.

$$\begin{aligned} P_{\text{fail}} &= 1 - [P(0 \text{ fails}) + P(1 \text{ fails}) + P(2 \text{ fails})] \\ P(0 \text{ fails}) &= (1-C)^k \\ P(1 \text{ fails}) &= kC(1-C)^{k-1} \\ P(2 \text{ fails}) &= \frac{k(k-1)}{2}C^2(1-C)^{k-2} \end{aligned}$$

Therefore, the probability of a k-drive RAID failing in a given hour is

$$P_{\text{fail}} = 1 - (1-C)^k - kC(1-C)^{k-1} - \frac{k(k-1)}{2}C^2(1-C)^{k-2}.$$

Problem 8 (10 points) Disk requests come into the disk driver for cylinders 25, 40, 7, 32, 22, 5, and 15, in that order. A seek takes 20 msec per cylinder moved. How much seek time is needed for

(a) First-come, first served.

(b) Shortest Seek First (SSF).

(c) Elevator algorithm (initially moving upward).

In all cases, the arm is initially at cylinder 17.

(a) **First-Come First-Serve:** Disk Arm Movement Order: $17 \to 25 \to 40 \to 7 \to 32 \to 22 \to 5 \to 15$. The number of cylinders needed is $8 + 15 + 33 + 25 + 10 + 17 + 10 = 118$ cylinders. The total seek time needed is $118 \cdot 20\,\text{ms} = 2{,}360\,\text{ms} = 2.36\,\text{s}$.

(b) **Shortest Seek First (SSF):** Disk Arm Movement Order: $17 \to 15 \to 22 \to 25 \to 32 \to 40 \to 7 \to 5$. The number of cylinders needed is $2 + 7 + 3 + 7 + 8 + 33 + 2 = 62$ cylinders. The total seek time needed is $62 \cdot 20\,\text{ms} = 1{,}240\,\text{ms} = 1.24\,\text{s}$.

(c) **Elevator Algorithm:** Disk Arm Movement Order: $17 \to 22 \to 25 \to 32 \to 40 \to 15 \to 7 \to 5$. The number of cylinders needed is $5 + 3 + 7 + 8 + 25 + 8 + 2 = 58$ cylinders. The total seek time needed is $58 \cdot 20\,\text{ms} = 1{,}160\,\text{ms} = 1.16\,\text{s}$.