Question 1

Suppose that a disk drive has 4000 cylinders, numbered 0 to 3,999. The drive is currently serving a request at cylinder 1,150, and the previous request was at cylinder 1,805.

The queue of pending requests, in FIFO order, is:

2,000; 1,212; 2,396; 2,800; 544; 1,618; 346; 1,523; 3,965; 3,681

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms?

a. FCFS

First-Come, First-Served (FCFS)

FIFO Order	FCFS distance (cylinders)
2000	|2000 - 1150| = 850
1212	|2000 - 1212| = 788
2396	|2396 - 1212| = 1184
2800	|2800 - 2396| = 404
544	|2800 - 544| = 2256
1618	|1618 - 544| = 1074
346	|1618 - 346| = 1272
1523	|1523 - 346| = 1177
3965	|3965 - 1523| = 2442
3681	|3965 - 3681| = 284
	TOTAL: 11731 cylinders

b. SCAN

SCAN disk scheduling algorithm

moving downwards

FIFO Order	FCFS distance (cylinders)
	|1150 - 544| = 606
	|544 - 346| = 198

lowest request = 346

moving upwards

FIFO Order	FCFS distance (cylinders)
	|346 - 1212| = 866
	|1212 - 1523| = 311
	|1523 - 1618| = 95
	|1618 - 2000| = 382
	|2000 - 2396| = 396
	|2396 - 2800| = 404
	|2800 - 3681| = 881
	|3681 - 3965| = 284
	TOTAL: 4423 cylinders

c. C-SCAN

C-SCAN disk scheduling algorithm

moving downwards

FIFO Order	FCFS distance (cylinders)
	|1150 - 544| = 606
	|544 - 346| = 198

moving downwards

FIFO Order	FCFS distance (cylinders)
	|346 - 0| = 346

moving downwards

FIFO Order	FCFS distance (cylinders)
	|3999 - 3965| = 34
	|3999 - 3681| = 284
	|3681 - 2800| = 881
	|2800 - 2396| = 404
	|2396- 2000| = 396
	|2000 - 1618| = 382
	|1618 - 1523| = 95
	|1523 - 1212| = 311
	TOTAL: 3937 cylinders

Feature	RAID 1	RAID 5
Configuration	Mirroring (Data is duplicated on two or more disks)	Striping with parity (Data and parity spread across all disks)
Fault Tolerance	High (Can tolerate failure of any mirrored disk)	Moderate (Can tolerate failure of one disk)
Read Performance	Excellent (Can read from any mirrored disk)	Good (Data is read from multiple disks)
Write Performance	Moderate (Data must be written to all mirrored disks)	Slower (Requires parity calculation and writing)
Storage Efficiency	50% (Usable capacity is half due to mirroring)	High (Uses (N-1)/N of total disk capacity)
Best Use Cases	Critical files requiring high availability, such as databases, system files, and VMs	Large files, backups, and general-purpose storage
Recovery Time	Fast (Mirrored disk is immediately available)	Longer (Requires parity reconstruction)
Data Availability	High (Data remains available as long as one disk survives)	Good (Data remains available after one disk failure)
Cost	High (Requires double the disk space)	Moderate (Better storage efficiency with more disks)
Write Penalty	Low (Simple duplication of data)	High (Data and parity must both be updated)
Capacity	Low (50% of total capacity)	Higher (More efficient use of total capacity)
Suitability for Small Files	Excellent (Quick access and write)	Good (May be impacted by parity calculations)
Suitability for Large Files	Moderate (Good reads, slower writes)	Excellent (Efficient storage with parity protection)

RAID1
RAID1 is best used for files that are frequently accessed.
Examples of these are files on web servers, files used on virtual machines, Operating system files, boot files, & database files are examples of the files that best stored with RAID 1.

RAID5
RAID5 is best used for large files & in scenarios where storage needs to be cost effective.
Examples of these are Media Files, backup/archive/ data files, email server files, shared network files are examples of the files that best stored with RAID 5.

Question 2

a) what is the advantage of supporting memory-mapped I/o to device-control registers?

Memory-mapped I/O (MMIO) removes the need for specific I/O instructions from the instruction set.
This enables **greater flexibility:** protection rules that prevent user programs from executing these I/O instructions do not need to be enforced.

All devices are treated as memory locations, thus the hardware and software interface becomes **less complex**. This simplifies both the software development of instructions & the operating systems' device driver architecture.

MMIO makes use of the CPU's built-in memory access techniques, maybe paging, caching, and other memory system enhancements,
MMIO minimizes the overhead associated with device-specific I/O operations which increases the speed at which device registers can be accessed leading to **performance enhancement.**

b) what is the disadvantage of supporting memory-mapped I/O to device-control registers?

Memory-mapped I/O (MMIO) while flexible, needs to be properly safeguarded.
Device-control registers typically shouldn't be cached. In MMIO, Device control devices are accessed like memory, so the memory translation units need to ensure that:
- memory addresses associated with device control registers are not accessible by other programs
- device registers must be properly mapped or emulated for guest operating systems.
- programming errors such as writing to incorrect addresses can easily occur
- caching for MMIO regions should are disabled

Normal memory accesses and MMIO share the same memory bus. Excessive numbers of MMIO operations could lead to memory bus conflict and a possible reduction in system performance.

For MMIO, the memory area takes up memory address space

Consider a file system that uses i-nodes to represent files. Disk blocks are 8kB in size, and a pointer to a disk block requires 4bytes.

This file system has 12 direct disk blocks, as well as single, double, and triple indirect disk blocks. What is the maximum size of a file that can be stored in this file system?

Direct Blocks	12pointers × 8KB = 96KB
Single Indirect Block	**Number of pointers** 8192B/4B = 2048pointers **block size** 2048 × 8KB = 16 384KB
Double Indirect Block	**Number of data blocks** 2048pointers × 2048pointers = 4 194 304pointers **block size** 4 194 304 × 8KB = 33 554 432 KB
Triple Indirect Block	**Number of data blocks** 2048pointers × 2048pointers × 2048pointers = 8 589 934 592KB **block size** 8 589 934 592KB × 8KB = 68 719 476 736KB
maximum size of a file that can be stored in this file system	96KB 16 384KB 33 554 432 KB 68 719 476 736KB = 68,753,047,648 KB.

Explain why logging metadata updates ensures recovery of a file system after a file-system crash.

The crash-consistency problem is to make the file system state changes atomic in the face of failures such as power loss of operating system crash.

Journaling (write ahead logging) is the most commonly used solution for the crash-consistency problem. It involves writing the metadata changes to a log, or journal, that is kept in a defined position within the file system. After a crash, the file system can use the log to undo the modifications since they are written there.

Advantages Journaling:

To **maintain consistency** throughout the filesystem, the system will identify & replay the log to redo any metadata updates that are incomplete.

To **minimise data loss** is, data is limited by the system to only the most recent operations.

Metadata updates are logged by the system to **keep track of operations** in progress during a crash.

During an operation, if a crash occurs, the file system can replay the log to redo any metadata updates that are incomplete or roll it back to **prevent partial/inconsistent changes**. This ensures that metadata updates are atomic: either done completely or not at all.

Question 3

Why do operating systems mount the root file system automatically at boot time?

The kernel (primitive program) mounts the file system it boots the system with the most vital parts of the operating system into memory. These vital parts include the core system functions as well the fundamental directory structure for user files, programs, and libraries.

An experimental addition to UNIX allows a user to connect a watchdog program to a file. The watchdog is invoked whenever a program requests access to the file. The watchdog then either grants or denies access to the file.

Discuss two pros and two cons of using watchdogs for security.

The watchdog associated with a file is a characteristic of that file, much like its owner or protection mode.
the information belongs in the file’s i-node, where it can be modified only through secure system calls.

Pros:

Context awareness

Watchdogs can make context aware decisions, allowing the system to grant or deny access in different scenarios such as
-unauthorized access
-malicious access
-user identity
-time of access

Cons:

Watchdogs while providing oversight on file access requests, may introduce a bottleneck in file operations.
The accuracy and integrity of the watchdog program itself are now crucial to the system's security.
If the watchdog program itself is compromised, the security of the system would also be compromised as it is reliant on the integrity of the watchdog program.

Question 4

Discuss the strengths and weaknesses of implementing an access matrix using capabilities that are associated with domains

An access matrix is used to manage and restrict permissions. It's a table that displays the actions that a user or group of users can take on particular system objects (read, write, execute, print).

Capabilities are tokens of access, giving domains access rights

Advantages	Disadvantages
Greater flexibility and quicker object access (compared to object-sepcific ACL).	less control over revoking/restricting flow of capabilities as the system would need to locate & nullify all copies of a capability
transferability of capabilites: access rights can be adjusted easiliy by passing capabilities between domains.	unauthorised access may happen if capabilities are exchanged inappropriately or disclosed between domains.
scalability/flexibility for distributed systems through decentralisation of access control.	access audits can be difficult as information is decentralised accross domains. this can also introduce a storage/memory overhead as decentralised capabilites would take up more resources compared to a central source.
Granularity over control of objects which can greatly improve security	there can be complexity in designing granularity levels for many domains