**Operating Systems Bank of Questions Summary**

### 1. List the layers involved in file system implementation

**Answer:** **Mnemonic: D-B-I-F-U (“D-BIFU”)**

1. **Device Layer**: Handles the actual hardware of disk drives.
2. **Block Layer**: Deals with logical blocks of data.
3. **I/O Control Layer**: Contains device drivers and interrupt handlers.
4. **File System Layer**: Organizes data into files and directories.
5. **User Interface Layer**: Offers system calls for user-level interaction.

**Analogy**: Like a library – storage is the shelves (device), blocks are books, control is the librarian, FS is the catalog, UI is the search screen.

### 2. Describe the basic structure of a file system

**Answer:**

1. **Boot Control Block**: Contains boot loader code and OS bootstrap info.
2. **Volume Control Block**: Stores metadata about volume – size, block count.
3. **Directory Structure**: Organizes and maps file names to blocks.
4. **File Control Blocks (FCBs)**: Holds metadata for each file (permissions, owner, size, etc).

**Diagram**:

Disk  
├── Boot Block  
├── Volume Control Block  
├── Directory Structure  
└── File Control Blocks

### 3. Explain the concept of recovery in operating systems

**Answer:**

1. **Definition**: Recovery is the process of restoring a system to a consistent state after failure.
2. **Goal**: Maintain data integrity and system availability.
3. **Mechanism**: Uses logs, checkpoints, and transaction rollbacks.
4. **Types**: Includes crash recovery, application recovery, and database recovery.

**Quote**: “Every reliable system must assume it will fail. Recovery is not optional—it’s foundational.”

### 4. What are the different types of failures that require recovery?

**Answer:**

1. **Transaction Failure**: Occurs when an operation cannot complete due to logical errors (e.g., divide by zero).
2. **System Crash**: Operating system halts unexpectedly (e.g., due to kernel panic or power loss).
3. **Disk Failure**: Data is lost due to bad sectors or head crashes.
4. **Media Failure**: Entire storage medium becomes unreadable.

**Mnemonic**: T-S-D-M (Transaction, System, Disk, Media).

### 5. Explain the undo and redo operations in recovery

**Answer:**

1. **Undo Operation**: Reverts the effects of uncommitted transactions.
2. **Redo Operation**: Re-applies effects of committed transactions that might not have been saved.
3. **Log-based Recovery**: Uses logs to track actions; each log entry supports undo/redo.
4. **Durability Principle**: Redo ensures committed changes persist post-failure.

**Table**:

| Operation | Purpose | When Used |
| --- | --- | --- |
| Undo | Cancel changes | Incomplete before crash |
| Redo | Reapply changes | Committed before crash |

### 6. What is checkpointing? How does it help in recovery?

**Answer:**

1. **Definition**: A checkpoint is a snapshot of the OS state at a specific time.
2. **Purpose**: Reduces time needed to recover from a failure.
3. **Mechanism**: OS records all necessary data in persistent storage.
4. **Benefit**: Only need to process log entries after the latest checkpoint.

**Example**: Like saving a draft – resume from last save.

### 7. Explain demand paging with a diagram. What are its advantages and disadvantages?

**Answer:**

1. **Definition**: Loads pages into memory only when needed.
2. **Mechanism**: On page fault, OS loads required page from secondary storage.
3. **Advantages**: Lower memory usage, fast startup.
4. **Disadvantages**: High page fault rate can cause delays.

**Diagram**:

CPU → Page Table → Memory  
 ↓  
 Secondary Storage (on fault)

### 8. Explain the concept of Copy-On-Write (COW). How is it used in process creation?

**Answer:**

1. **COW Concept**: Processes initially share memory pages after a fork().
2. **Efficiency**: Saves memory and reduces process creation time.
3. **Modification Trigger**: When a process modifies a shared page, it gets a private copy.
4. **Use in Forking**: Forked processes share until a write occurs.

**Analogy**: Two people reading the same newspaper until one decides to annotate.

### 9. What is page replacement? Explain FIFO, LRU, and Optimal algorithms with examples.

**Answer:**

1. **Definition**: When memory is full, OS must swap out pages.
2. **FIFO**: Remove oldest page (first-in).
3. **LRU**: Remove least recently used page.
4. **Optimal**: Remove page that won’t be used for longest time (ideal).

**Example** (Page ref: 1, 2, 3, 1, 4):

* FIFO: 1,2,3 → remove 1 → fault.
* LRU: Use timestamp to decide.
* Optimal: Requires future knowledge.

### 10. Explain frame allocation strategies. Compare fixed and variable allocation.

**Answer:**

1. **Fixed Allocation**: Each process gets a fixed number of frames.
2. **Variable Allocation**: Number of frames depends on process size or priority.
3. **Comparison**:
   * Fixed: Simpler, may waste memory.
   * Variable: More efficient, complex to implement.
4. **Global vs Local**: Allocation can be done from a global pool or local per-process.

### 11. What is thrashing? How can it be detected and prevented?

**Answer:**

1. **Thrashing**: High paging → low CPU utilization.
2. **Detection**: Spike in page fault rate and disk activity.
3. **Prevention**: Use working set model to allocate enough frames.
4. **Solutions**: Reduce degree of multiprogramming or increase RAM.

**Analogy**: Constantly switching TV channels and never watching a full show.

### 12. Describe memory-mapped files. How do they work and what are their uses?

**Answer:**

1. **Definition**: File contents mapped directly to virtual memory.
2. **Efficiency**: Enables fast file access without read/write syscalls.
3. **Use Cases**: Inter-process communication, DB systems.
4. **Mechanism**: Uses mmap() system call (in UNIX).

### 13. Discuss kernel memory allocation techniques with examples.

**Answer:**

1. **Buddy System**: Allocates memory in blocks of 2^n sizes.
2. **Slab Allocator**: Allocates pre-initialized memory objects.
3. **kmalloc/vmalloc**: Used in Linux kernel for dynamic allocation.
4. **Fragmentation Handling**: Both systems reduce fragmentation.

### 14. Describe the components and goals of storage management in OS.

**Answer:**

1. **Components**: File system, memory management, device drivers.
2. **Goals**:
   * Efficient space usage.
   * Data reliability and security.
   * High performance.
3. **Storage Hierarchy**: Registers → Cache → RAM → Disk → Tape.
4. **Protection Mechanisms**: Prevent unauthorized access.

### 15. Explain file concepts and different file access methods.

**Answer:**

1. **File Concepts**: A file is a named collection of related information.
2. **Access Methods**:
   * Sequential Access
   * Direct Access
   * Indexed Access
3. **Sequential**: Read linearly.
4. **Indexed**: Like book index for faster search.

### 16. Explain directory structures and file system mounting.

**Answer:**

1. **Structures**:
   * Single-Level
   * Two-Level
   * Tree
   * DAG (for shared files)
2. **Mounting**: Integrate external file system into existing hierarchy.
3. **Mount Point**: Location where FS is attached.
4. **Unmounting**: Safely removing the FS.

### 17. How does OS handle file sharing and protection?

**Answer:**

1. **Sharing Mechanisms**: File descriptors, symbolic links.
2. **Protection Models**:
   * Access Control Lists (ACL)
   * User/Group/Other permissions
3. **Concurrency Control**: Locks, semaphores.
4. **Security**: Prevents unauthorized access/modification.

### 18. Explain file system structure and its implementation layers

**Answer:**

1. **File System Structure**: Organized into boot block, volume control block, directory structure, and file control blocks.
2. **Layered Implementation**:
   * **Logical File System**: Manages metadata, permissions.
   * **File-Organization Module**: Manages files and record blocks.
   * **Basic File System**: Issues generic commands to device driver.
   * **I/O Control**: Interfaces with device drivers.
3. **Device Drivers**: Handles communication with actual hardware.
4. **Benefits**: Modular design, abstraction, easier maintenance.

### 19. Explain the concept of recovery in operating systems

**Answer:**

1. **Recovery Concept**: Restoring consistency after crashes or failure.
2. **Techniques**: Logs, checkpoints, undo/redo operations.
3. **Consistency**: Ensures atomicity and durability.
4. **Goals**: Minimize data loss and restore system quickly.

### 20. Describe types of failures and log-based recovery mechanisms (Undo/Redo), along with checkpointing

**Answer:**

1. **Types of Failures**:
   * Transaction
   * System crash
   * Disk failure
   * Media failure
2. **Log-based Recovery**:
   * Maintains log of actions before execution.
   * Enables redo and undo.
3. **Undo/Redo**: Based on log entries, executed during recovery.
4. **Checkpointing**: Periodic snapshot to reduce log size and recovery time.

### 21. Explain the overview and components of mass storage systems

**Answer:**

1. **Overview**: Long-term, non-volatile storage for large datasets.
2. **Components**:
   * Disks (HDD/SSD)
   * Storage arrays
   * Controllers
3. **Functions**: Data retention, backup, high-speed access.
4. **Characteristics**: Capacity, reliability, cost-efficiency.

### 22. Describe the disk structure and explain how data is organized

**Answer:**

1. **Structure**: Disks have platters, tracks, sectors, cylinders.
2. **Data Layout**: Organized into concentric circles (tracks) and radial arms (cylinders).
3. **Access**: Read/write heads move over tracks.
4. **Efficiency**: Depends on layout and seek strategies.

### 23. Discuss various disk scheduling algorithms

**Answer:**

1. **FCFS**: First-Come First-Serve – Simple but inefficient.
2. **SSTF**: Shortest Seek Time First – Prioritizes nearest request.
3. **SCAN**: Elevator algorithm – Moves head back and forth.
4. **LOOK**: Optimized SCAN – Turns around at final request.

**Comparison Table**:

| Algorithm Performance Fairness Use Case |  |  |  |
| --- | --- | --- | --- |
| FCFS | Poor | High | Simple systems |
| SSTF | Better | Low | Real-time |
| SCAN | Good | Medium | Batch processing |
| LOOK | Efficient | Medium | Mixed loads |

### 24. What is disk management? Mention its key functions

**Answer:**

1. **Disk Partitioning**: Divides disk into logical segments.
2. **Formatting**: Prepares disk with file system.
3. **Bad Block Recovery**: Identifies and remaps damaged sectors.
4. **Space Management**: Tracks free/used blocks.

### 25. Explain swap-space and its management

**Answer:**

1. **Definition**: Disk area used as overflow for main memory.
2. **Purpose**: Allows larger virtual memory than physical RAM.
3. **Management**:
   * Static: Fixed swap partition.
   * Dynamic: Expands/allocates as needed.
4. **Performance**: Poor compared to RAM, but improves multitasking.

### 26. Explain the concept of RAID and describe its advantages

**Answer:**

1. **RAID Definition**: Redundant Array of Independent Disks.
2. **Purpose**: Improves performance, fault tolerance.
3. **Advantages**:
   * Redundancy
   * Faster access (striping)
   * Data protection (parity)
4. **Levels**: 0 (striping), 1 (mirroring), 5 (block-level parity), etc.

### 27. What is storage implementation? Discuss its key techniques

**Answer:**

1. **Definition**: Methods used to handle storage allocation and access.
2. **Techniques**:
   * Caching: Store frequently accessed data in memory.
   * Buffering: Temp storage during data transfer.
   * Spooling: Overlapping output/input operations.
   * RAID: Improves reliability.

### 28. What is tertiary storage? Describe its features and use cases

**Answer:**

1. **Definition**: Lowest level of storage hierarchy.
2. **Examples**: Magnetic tapes, optical disks.
3. **Features**:
   * High capacity
   * Slow access
   * Inexpensive
4. **Use Cases**: Backup, archiving, compliance storage.

### 29. Explain the structure of an I/O system in an operating system

**Answer:**

1. **Components**:
   * I/O devices
   * Device controllers
   * Device drivers
   * OS interface
2. **Data Flow**: Application → System Call → Driver → Device.
3. **Responsibilities**: Buffering, scheduling, error handling.
4. **Efficiency Goal**: Minimize CPU involvement.

### 30. What are the main components of I/O hardware, and what are their functions?

**Answer:**

1. **I/O Devices**: Input (keyboard), Output (monitor), Storage (disk).
2. **Controllers**: Manage signals and protocol with devices.
3. **Ports**: Physical connectors for communication.
4. **Buses**: Carry data between components.

### 31. Describe the purpose and components of the application I/O interface

**Answer:**

1. **Purpose**: Provides abstraction for user-level I/O.
2. **Components**:
   * open()
   * read()
   * write()
   * close()
3. **Goal**: Hide hardware specifics from user.
4. **Benefit**: Easier programming and portability.

### 32. What are the responsibilities of the kernel I/O subsystem?

**Answer:**

1. **Scheduling I/O**: Manages queue of I/O requests.
2. **Buffering**: Temp storage during transfer.
3. **Caching**: Speed up access to data.
4. **Error Handling**: Detect and recover from I/O errors.

### 33. Explain how I/O requests are transformed into hardware-level operations

**Answer:**

1. **System Call**: Application triggers an I/O syscall.
2. **Driver Invocation**: Kernel calls device driver.
3. **Command Translation**: Driver converts to device-specific instructions.
4. **Device Operation**: Controller executes and confirms.

### 34. Explain the structure and working of mass storage systems in operating systems

**Answer:**

1. **Structure**: Controllers, devices, OS management.
2. **Working**: Uses scheduling and buffering.
3. **Features**: Reliability, redundancy, speed.
4. **Examples**: SSDs, HDDs, hybrid arrays.

### 35. Describe disk structure, disk scheduling algorithms, disk management techniques, and swap-space management in detail

**Answer:**

1. **Disk Structure**: Tracks, sectors, cylinders.
2. **Scheduling**: FCFS, SSTF, SCAN, LOOK.
3. **Management**: Partitioning, formatting, bad block recovery.
4. **Swap-Space**: Virtual memory management using disk.

### 36. Explain the RAID structure in detail

**Answer:**

| Level Description Fault Tolerance Performance |  |  |  |
| --- | --- | --- | --- |
| RAID 0 | Striping only | None | High |
| RAID 1 | Mirroring | High | Moderate |
| RAID 5 | Block-level with parity | Moderate | High |
| RAID 6 | Dual parity | Very High | Moderate |
| RAID 10 | Striping + Mirroring | High | High |

### 37. Compare different RAID levels using a table, describe the techniques used in storage implementation

**Answer:**

1. **RAID Table**: (See Q36)
2. **Techniques**:
   * Striping: Data split across disks
   * Mirroring: Duplication for safety
   * Parity: Error correction
3. **Benefits**: Speed + reliability
4. **Use Cases**: Enterprise servers, databases.

### 38. Discuss the characteristics and use cases of tertiary storage

**Answer:**

1. **Characteristics**:
   * Low cost
   * High latency
   * Sequential access
2. **Examples**: Tape drives, Blu-ray
3. **Use Cases**:
   * Long-term archiving
   * Disaster recovery
4. **Advantages**: Low power, durability.

### 39. Describe in detail the structure of I/O systems in an operating system

**Answer:**

1. **Layered Design**: Application → Kernel → Driver → Device
2. **Components**: I/O requests, scheduler, buffer cache.
3. **Goal**: Efficient and secure data movement.
4. **Coordination**: Between software and hardware levels.

### 40. Explain the components of I/O hardware, the application I/O interface, kernel I/O subsystem, and how I/O requests are transformed into hardware operations

**Answer:**

1. **I/O Hardware**: Devices, controllers, ports, buses.
2. **App Interface**: open(), read(), write(), close()
3. **Kernel Subsystem**: Scheduling, buffering, error handling.
4. **Transformation Flow**: System call → Driver → Device command → Execution → Return.