OPERATING SYSTEM LAB REPORT(BOOK/MANUAL)

Name: Amit Sarkar

Intake: 51

Section: 07

Roll: 22235103283

Department: CSE

University: Bangladesh University of Business and Technology (BUBT)

Semester: Fall 2025

Submitted To: Saimoon Al Farshi Oman

Submission Date: 07-12-2025

Table of Contents

Chapter 1: Introduction to Operating Systems

An Operating System (OS) is system software that manages computer hardware, software resources, and provides common services for computer programs. It acts as an intermediary between users and the computer hardware.

- **Functions of an Operating System:**

  - ✓ 1. Process Management
  - ✓ 2. Memory Management
  - ✓ 3. File System Management
  - ✓ 4. Device Management
  - ✓ 5. Security and Access Control

- **Examples of Operating Systems:** Linux, Windows, macOS, and Android.

Chapter 2: Basic Linux Commands

Linux provides a command-line interface for interacting with the system. Here are some **basic commands:**

pwd: Displays the current working directory.

Example:

```
$ pwd
/home/user
```

ls: Lists files and directories.

Example:

```
$ ls
```

Documents Downloads Pictures

cd: Changes the directory.

Example:

```
$ cd Documents
```

mkdir: Creates a new directory.

Example:

```
$ mkdir my_folder
```

rmdir: Removes an empty directory.

Example:

```
$ rmdir my_folder
```

cp: Copies files or directories.

Example:

```
$ cp file1.txt file2.txt
```

mv: Moves or renames files.

Example:

```
$ mv old.txt new.txt
```

rm: Removes files.

Example:

```
$ rm unwanted.txt
```

cat: Displays the contents of a file.

Example:

```
$ cat notes.txt
```

chmod: Changes file permissions.

Example:

```
$ chmod 755 script.sh
```

Chapter 3: File Management in Linux

File management in Linux involves creating, editing, viewing, and organizing files and directories. Each file has permissions, ownership, and types.

Examples:

```
$ touch myfile.txt   → Create a new file

$ echo 'Hello World' > myfile.txt   → Add content

$ cat myfile.txt   → View file contents
```

Output:

```
Hello World

$ cp myfile.txt copy.txt   → Copy a file

$ mv copy.txt backup.txt   → Rename or move a file

$ rm backup.txt   → Delete a file

$ chmod 777 myfile.txt   → Give full permissions to all
users
```

Chapter 4: User Management in Linux

Linux is a multi-user system, allowing multiple users to access and use resources securely.

Useful Commands:

```
$ adduser student   → Create a new user

$ passwd student   → Set or change password for user 'student'

$ whoami   → Display current logged-in user
Output:
student

$ groups student   → Show groups the user belongs to

$ su - student   → Switch to another user account

$ deluser student   → Remove a user from the system

User information is stored in '/etc/passwd' and group data in '/etc/group'.
```

Chapter 5: Basic Shell Programming

Shell scripting automates repetitive tasks by combining Linux commands into executable scripts.

Example 1: Print Hello World

```
#!/bin/bash
echo "Hello, World!"
```

Output:

```
Hello, World!
```

Example 2: Check Even or Odd Number

```
#!/bin/bash
echo "Enter a number:"
read num
if [ $((num % 2)) -eq 0 ]
then
  echo "Even number"
else
  echo "Odd number"
fi
```

Output:

Enter a number:

```
5
Odd number
```

Example 3: Simple Calculator

```
#!/bin/bash
echo "Enter two numbers:"
```

```
read a b
echo "Sum is: $((a + b))"
```

Output:

Enter two numbers:

```
4 6

Sum is: 10
```

Chapter 6: CPU Scheduling Algorithms

CPU Scheduling determines which process will use the CPU next.

Types of Scheduling Algorithms:

1. FCFS (First Come First Serve)

- Non-preemptive
- Simple but causes Convoy Effect

Example:

```
P1 → P2 → P3
```

2. SJF (Shortest Job First)

 Selects the shortest burst time

 Can be preemptive (SRTF) or non-preemptive

3. Priority Scheduling

 Highest priority process runs first

 Can cause starvation

4. Round Robin (RR)

- Time-sharing
- Each process gets a fixed time quantum

Example: Quantum = 4 ms

`P1(10) → P2(5) → P3(8) …`

5. Multilevel Queue Scheduling

Different queues for foreground/background tasks

Chapter 7: Process Synchronization

Ensures correct execution when multiple processes access shared data.

Critical Section Problem

Conditions:

1. Mutual Exclusion
2. Progress
3. Bounded Waiting

Synchronization Tools:

1. Semaphores

Two types:

- Binary (0 or 1)
- Counting

Example:

```
wait(s) → s = s - 1

signal(s) → s = s + 1
```

2. Mutex Locks

Used for exclusive access.

3. Sleep and Wakeup

Used for producer-consumer problems.

Chapter 8: Deadlock

Occurs when a set of processes are blocked, waiting for each other.

Conditions for Deadlock (Coffman Conditions):

1. Mutual Exclusion
2. Hold and Wait
3. No Preemption
4. Circular Wait

Deadlock Handling Methods:

1. Deadlock Prevention (Break one of the four conditions)
2. Deadlock Avoidance (Banker's Algorithm)
3. Deadlock Detection & Recovery
4. Ignore Deadlock (Ostrich Algorithm)

Banker's Algorithm Example:

Used to check safe state before resource allocation.

Chapter 9: Memory Management & Paging

Memory management allocates and deallocates memory to processes.

Contiguous Allocation:

- Single partition
- Multiple partition

Paging:

- Divides memory into fixed-size blocks (pages)
- Eliminates external fragmentation

Example:

```
Logical Address → Page Number + Offset
```

Segmentation:

- Variable-sized segments used according to program structure

Chapter 10: Virtual Memory & Page Replacement

Virtual memory allows execution of processes larger than physical memory.

Techniques:

1. Paging
2. Demand Paging
3. Copy-on-Write

Page Replacement Algorithms:

1. FIFO (First-In-First-Out)

Oldest page removed first.

2. LRU (Least Recently Used)

Uses usage history.

3. Optimal Page Replacement

Replaces page not needed for the longest time.

Example Table (for calculation tasks):

Pages: 1 2 3 4 2 1 5 1 2 3 4 5

Frames: 3

Chapter 11: File Systems & I/O Management

File System Components:

1. File attributes
2. Directory structure (Single, Two-level, Tree)

3. **File allocation methods:**
    - Contiguous
    - Linked
    - Indexed

I/O Management:

- Buffering
- Caching
- Spooling

Disk Scheduling Algorithms:

1. FCFS
2. SSTF
3. SCAN
4. C-SCAN

Example:

Requests: 82, 170, 43, 140, 24, 16, 190

Initial head: 50

Chapter 12: Conclusion

The Operating Systems lab has provided a strong practical foundation in understanding how modern computer systems function and operate. Throughout this course, I gained hands-on experience with Linux commands, file and user management, and essential shell scripting techniques that form the backbone of system-level operations. In the second half of the semester, I explored advanced OS concepts such as CPU scheduling, process synchronization, deadlock handling, memory management, and file system structures. These topics not only strengthened my theoretical understanding but also enhanced my problem-solving abilities through practical experimentation and algorithm-based tasks.

Overall, this lab has significantly improved my confidence in working with operating systems, managing resources efficiently, and understanding real-world system behavior. The knowledge and skills acquired in this lab will be valuable for future courses, system administration tasks, and professional development in the field of computer science and engineering.

Chapter 13: References

- Class lecture notes
- Linux manual pages
- Silberschatz, Galvin — Operating System Concepts
- Online tutorials (GeeksforGeeks, Ubuntu Docs, TutorialsPoint)