Operating System

Assignment - 1

PART A (Short Answer Type):

1. Despite the evolution of hardware, why do modern systems still rely heavily on Operating Systems?

⇒ Modern systems still need OS bcoz it manages resources like CPU, memory and I/O while providing abstractions such as processes & files. It ensures security, multitasking & portability so applications can run without directly handling complex hardware.

2. You are asked to design an OS for a wearable health device that monitors heart rate. Which type of operating system would be most suitable & Why?

⇒ A real-time embedded OS (RTOS) is most suitable. It offers predictable response to sensor inputs, efficient scheduling and low power consumption. These features are critical for a wearable device that must react quickly to heart-rate data and operate on limited battery.

3. Given the choice to build a new OS Kernel for a performance-critical environment, which structure would you avoid (Monolithic, Layered, Microkernel) and Why?

⇒ Avoid microkernel architecture, since its reliance on message passing and frequent context switchess adds overhead. In performance-

critical systems, this reduces efficiency compared to monolithic or layered kernels, which allow faster direct calls.

4. A developer claims that OS structure doesn't matter as long as processes run. Support or refute this claim with reasoning.

⇒ Yes, OS structure matters. It impacts performance, security, modularity & fault isolation. For example, monolithic kernels can be fast but less reliable, while microkernels improve isolation but add overhead. Thus, structure influences efficiency & stability, not just process execution.

5. While debugging, you find a process switching error.

(i) Explain how analysing the Process Control Block (PCB) can point to misinitialized registers or incorrect states.

⇒ PCB contains saved registers, PC and state; errors here reveal misinitialized values or incorrect states.

(ii) If a task is moved unexpectedly from running to waiting, what precisely does context switching involve?

⇒ context switching saves the current process state, updates its PCB, loads the next process state, and resumes execution.

(iii) The OS needs to allocate I/O resources mid execution. Which type of system call would you use (e.g. blocking, non-blocking, synchronous, asynchronous) & why?

⇒ Asynchronous non-blocking calls should be used, allowing execution to continue while I/O is handled, ensuring responsiveness.

PART - B (Application / Numerical Based):

6. (a) Total time = Save + Load + Scheduler
= 2ms + 3ms + 1ms
= 6ms

(b) Each context switch adds overhead with no useful work. Too many switches reduce CPU time for processes, lowering throughput & increasing response time.

7. Single-thread time = 40 sec
Execution Time ≈ 40/n (With n threads in ideal conditions)
Example (n = 4) = 40/4 = 10 sec.
Multithreading improves performance by running tasks in parallel and overlapping I/O & computation reducing total execution time.

8. (a) FCFS (First-Come, First Served)

P_1	P_2	P_3	P_4
0	5	8	16	22

P_1 executes from 0 ms to 5 ms (burst time = 5ms)
P_2 executes from 5ms to 8ms (burst time = 3ms)
P_3 executes from 8ms to 16ms (burst time = 8 ms)
P_4 executes from 16ms to 22ms (burst time = 6 ms).