Tripti Singh
2301010246
Btech CSE (4)

Date / /
Page No. 1

Assignment 1

Ans1: Because OSs provide essential abstractions and services that hardware alone does not, processes and thread management, memory protection & virtual memory, device and I/O abstraction, file systems, security (authorization / authentication), resource scheduling, & a stable API for applications. The OS enables portability, isolation & multiplexing of scarce resources.

Ans2: A real time embedded OS (RTOS) - specifically a hard / firm real-time flavor depending on safety needs. Reason: timely, predictable response to sensor input & alarms, small footprint, power-awareness, & deterministic scheduling.

Ans3. Avoid a microkernel if raw performance is the single top priority. Microkernels improve modularity & reliability but incur IPC/context-switch overhead b/w server processes (drivers, filesystems), which can reduce throughput and increase latency versus a carefully optimized monolithic kernel that keeps services "in-kernel" to minimize crossings.

Ans4 Refute. Structure affects performance (latency, throughput) reliability, security, maintainability and complexity of adding features. Two OS, that both "run processes" may differ hugely in predictability, ability to isolate faults, driver complexity and scalability. Thus structure does matter.

Ans5 (i) PCB holds registers, PC, SP, state, etc. checking it can reveal wrong values (misinitialized registers, incorrect PC/SP, wrong state flags) that indicate errors in save/restore or scheduling.

(ii) Save current CPU state to PCB, mark process state (RUNNING → WAITING) & scheduler picks next process, load its state from PCB, update to RUNNING and resume execution.

(iii) Use non-blocking/asynchronous I/O so CPU work continues while I/O is allocated. Blocking calls stall execution, so asynchronous is preferred for responsiveness.

Ans6 Save state = 2 ms
Load state = 3 ms
scheduler overhead = 1 ms

(i) Total CST =
context switching time = 2 + 3 + 1 = 6 ms.

(iii) Impact on multitasking performance will be:
→ During switch time no useful work is done by user processes.
→ frequent switches (short tasks/ quantum) → more CPU time wasted.
→ Leads to lower CPU utilization & reduced throughput.
→ Can cause response jitter (unpredictable delays).

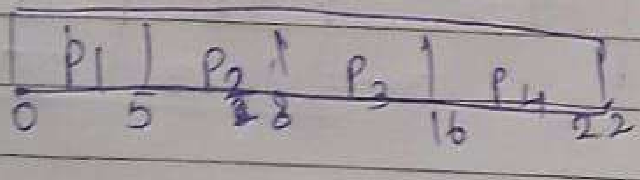
Ans 7. (i) Singly - threaded time = 40s; assuming ~~estimating 2 threads~~ 2 parallel threads and ideal conditions.
Total time will be 40/2 = 20s

(ii) (a) Exploits parallelism on multi-core CPUs (divide work across cores).
(b) Hides latency (I/O or blocking calls) by letting other threads run.
(c) Reduces overall elapsed (wall-clock) time if tasks are CPU-parallel or I/O overlapped.

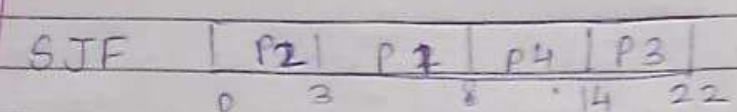
Ans 8

Pcs	Bt Time
P1	5
P2	3
P3	8
P4	6

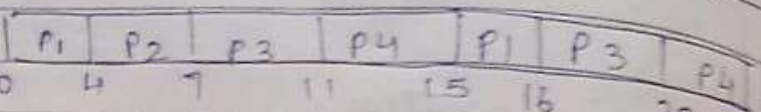
• FCFS



- SJF



- Round Robin



(b) WT and Turn Around time

- FCFS

	Burst	completion	TAT (CT - AT)	WT (TAT - BT)
P1	5	5	5	0
P2	3	8	8	5
P3	8	16	16	8
P4	6	22	22	16
		average:	12.75	7.25

- SJF

SJF	Burst	completion	TAT	WT
P2	3	3	3	0
P1	5	8	8	3
P4	6	14	14	8
P3	8	22	22	14
		av. =	11.75	6.25

- RR (Q = 4ms)

	Burst Time	Completion	TAT	WT
P2	3	7	7	4
P1	5	16	16	11
P3	8	20	20	12
P4	6	22	22	16
		av =	16.25	10.75

(c) SJF: best av. WT (6.25ms) and TAT (11.75ms)
FCFS: Simple, but convoy effect possible.
RR: fairness + responsiveness, but higher av
30; SJF is optimal here.

Ans 9. (i) (a) Microkernel architecture is best for scalability and security because it runs only essential services in the kernel, reducing vulnerabilities and allowing easy updates or scaling of other services in user space.

(b) (i) Virtual machines isolate different service by running separate OS instances, prevent interference. They simplify management through easy deployement and migration and optimize resources by dynamically allocating CPU, memory and storage based on demand.

(ii) (a) The OS uses priority-based scheduling to ensure high-priority tasks like intrusion detection get immediate CPU access, while lower-priority tasks run when possible. Inter-process communication (IPC) allows tasks to exchange data & ~~continue~~ coordinate actions efficiently.

(b) Suitable algorithms include Priority scheduling for fixed priorities, Rate Monotonic Scheduling (RMS), for periodic tasks, and (EDF) for dynamic deadli