Viray Partap Singh.
2301010231
BTech CSE 4

Assignment → 1.

1. Modern system still rely on OS because:
Abstraction: OS hides hardware complexity (device, CPU).
so app run on many platform.
Resource Management:- OS schedule CPU, memory, I/O

2. Real time embedded OS (RTOS) is used because of predictable timing, low power footprint, small memory/CPU, deterministic interrupt handling.

3. While building a new Kernel we should avoid microkernel if raw performance is the only goal. Microkernel give modularity and safety but add IPC overhead because many services run in user space.

4. OS structure doesn't matter as long as process run. because of structure effect:-
- Performance.
- Security.
- Maintainability and extensibility.
- Feature Support.

5) i). PCB stores saved register, PC, stack pointer, state bits.
- PCB's saved PC points to an unexpected address. the next resume will jump to wrong code.

5 ii). Save current CPU context into it's PCB.
Change process state to waiting and enqueues on wait queue.
Update memory mapping.

iii). • Use asynchronous non-blocking I/O when you don't want the task to block and you want responsiveness.
• If task must wait for I/O result before proceeding, use blocking synchronous ~~et~~ calls

6. Save state = 2ms
Load state = 3ms
Scheduler overhead = 1ms

a) Total context switching time = 2+3+1 = 6ms
b) Impact on multitasking:
→ Every switch costs 6ms of CPU. High context-switch rate reduces effective CPU for user work.
→ Frequent switching increases cache.

7. Single threaded total time = 40s
Threads per process = 2

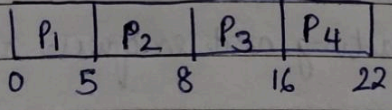
• Estimate execution time = $\frac{40}{2}$ = 20 seconds

• Multithreading helps allows parallelism on multiple core and hides latency by overlapping I/O with computation.

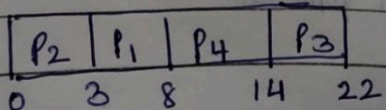
8.

Process	P_1	P_2	P_3	P_4
Time(ms)	5	3	8	6

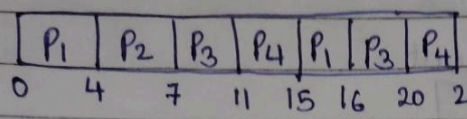
a) Gantt chart
• FCFS



• SJF



- Round Robin (Quantum = 4ms)



b) Waiting time = Start time - arrival time
TAT - burst

Turnaround time = Finish time - arrival time

FCFS: $P_1 = 5$, $P_2 = 8$, $P_3 = 16$, $P_4 = 22$

TAT = $5 + 8 + 16 + 22 = 51$

$= \frac{51}{4} = 12.75$

Waiting time = TAT - burst

$= 0 + 5 + 8 + 16 = 29.$

Avg. WT = $\frac{29}{4} = 7.25$

SJF $P_2 = 3$, $P_1 = 8$, $P_4 = 14$, $P_3 = 22$

TAT = $8 + 3 + 22 + 14 = 47$

$= \frac{47}{4} = 11.75$

Waiting time = TAT - burst

$8 - 5 = 3$
$3 - 3 = 0$
$22 - 8 = 14$
$14 - 6 = 8$

} Sum = 25

Avg. WT = $\frac{25}{4} = 6.25$

Round Robin $P_1 = 16$, $P_2 = 7$, $P_3 = 20$, $P_4 = 22$

TAT = $16 + 7 + 20 + 22 = 65.$

Avg. TAT = $\frac{65}{4}$ = 16.25

Waiting time = TAT - burst =

16 - 5 = 11
7 - 3 = 4
20 - 8 = 12
22 - 6 = 16

} Sum = 43

Avg. WT = $\frac{43}{4}$ = 10.75

c) SJF gives lowest average turnaround 11.75 and lowest avg. waiting 6.25 among the three.

9 i) a) Microkernel or modular layered architecture is preffered for scalibility and security.

- Microkernel : small trusted core, most services run in user space.
- Layered modular : It is designed also aids maintainbility and clear separation.

b). Isolation : Each virtual machine has separate guest OS and virtual hardware.

- Management : Virtual machine support snapshots, cloning, living migration, templates for rapid deployment.
- Reduce optimisation : Hypervisors multiplex CPU/memory/disk across virtual machine support overcommit, dynamic scaling and scheduler - level QoS.

ii)a) Ensuring high priority tasks are handled without delay.

- Priority-based scheduling :- Assign high priority to intrusion detection tasks.
- Real-time scheduling : Use Real time scheduler for critical tasks so deadlines are met.
- Priority inheritance : Avoid priority inversion when lower-priority tasks hold resources needed by high-priority tasks.

b)- Rate Monotonic Scheduling (RMS) : Good for periodic hard-real time tasks with fixed periods.

- Earliest Deadline First (EDF) : Optimal for dynamic deadlines and mixed workloads - better CPU utilization if tasks have different deadlines.
- Hybrid approach : Use real-time scheduler (RMS) for safety/critical tasks.