Q1. What are the four necessary and sufficient conditions for deadlock to occur?

Mutual Exclusion, No Preemption, Hold and Wait, Circular wait

3 points per answer

Q2. Imagine we have a 32-bit CPU and we use 2 kilobyte pages. If each page table entry is 4 bytes, how large of a page table would we need?

2^23 (8MB)

All or nothing

Q3. What is the relocation problem? How did we solve it when managing memory using fixed partitions?

Relocation is the idea that things (code and data) move over time. It could be between executions or even while the program is running. Things that move preclude the use of absolute addresses (either in the instruction stream or saved "pointers").

4 points for discussion hitting the main points above

We solved it in fixed partitions most likely by adding the absolute address to the base register value to turn it into a relative address. Other ideas we entertained were to only use relative addressing modes or to rewrite addresses when loading the program into RAM (relocations).

4 points for one of these solutions

Q4. What is the protection problem? How did we solve it when managing memory using fixed partitions?

The protection problem is acknowledging that physical memory is now shared, and one program could generate a physical address that belongs to another process, potentially corrupting it or reading its data.

4 points for discussion hitting the main points above

We solved this by having the CPU have two new registers, base and limit, whose values are set to the partition bounds by the OS on context switch. The CPU must check the effective address it calculates against these bounds and if exceeded, throw an exception to the OS.

4 points for this answer. 2 points of which are for explicitly stating the check must be done at the CPU level.

Q5. Use the Banker's algorithm to determine whether the state is safe or unsafe. If it is safe, indicate a schedule that will allow all jobs to complete.

It is safe, with schedule of A->B->C

All or nothing

Q6. What are internal and external fragmentation? How are they different?

Internal fragmentation is wasted space inside of a "chunk" (minimal unit of allocation) because the minimal unit of allocation is too big compared to the request.

(4 points)

External fragmentation is wasted space between allocations (chunks) that is too small to be used to satisfy future allocations.

(4 points)