Operating Systems (part 1) Exercises

These exercises are grouped into three parts. Part I contains routine exercises to help you understand the ideas directly presented in lectures. The exercises in Part II are designed to grow and deepen your understanding of principles that underpin Operating Systems. These exercises extend the lecture material and invite you to think about 'why' questions and about optimisations to what was lectured as well as alternatives to what was lectured. Part III are open-ended, stretching questions that you could tackle in your mini research project.

I hope you enjoy working through these questions, puzzles and research questions!

John Fawcett, July 2023

Part I

1. What could happen if the operating system allowed programs to talk to the real printer instead of to a virtual printer?
2. We said that virtual memory can be used to attach permissions to each page: readable, writable, executable. Explain how this could be implemented by adding 3 bits to the translation table entries, additional checks in the TLB hardware, and a new interrupt number.
3. Design a 2-level paging system for a computer that provides 32-bit virtual address spaces using 32-bit physical memory addresses. The page size should be 4 kiB.
4. Design a paging system for a computer with 64-bit virtual address spaces and 48-bit physical addresses, using 4 kiB pages.
5. Design a paging system for a computer with 32-bit virtual address spaces and 48-bit physical addresses, using 1 MiB pages.
6. Design a paging system for a computer with 48-bit virtual address spaces and 48-bit physical addresses, using 4 kiB pages.
7. Think of three ways to select a victim page. For each algorithm, give an example program for which it would make excellent decisions and another for which it would make poor decisions (i.e. it would victimise pages that are likely to be used again soon).
8. Slide 19, optimisation 2, suggested that dirty bits could reduce the page replacement time. What additional hardware or software functionality is required to make this work?

Part II

1. Design the operating system's virtual printer system using the idea of interrupt service routines. What interrupts and interrupt service routines would you create, and how would they work?
2. How could shared pages be implemented? That is, one area of physical memory that two processes can both read and write.
3. What advantages and disadvantages would result from using a larger page size? What about a smaller page size? What new difficulty would be created if an operating system tried to support two page sizes?
4. In practice, the TLB does not hold the full translation table; it only has space for 64 entries! How should this knowledge change the way you write programs?

Part III

1. In the days of MS DOS, Microsoft’s Disk Operating System, the extended memory manager (emm386.exe) was used to implement ‘solution 1’ on slide 3. Research how this worked and what shortcomings it had compared to the paging approach described in lectures. Does the DOS approach have any advantages?
2. We said that the operating system keeps a translation table for each process but we did not explain how it remembers where it put each translation table! Research the unix operating system’s concept of *process control block (PCB)* and write up what is stored there and what it’s used for.