University of Pennsylvania
CSE380 – Operating Systems
2nd Midterm Exam – 11/16/2004

Instructions: Write your answers in an exam book. Remember to print your name clearly on the exam book cover, to erase or cross out any material you don't want graded, and to indicate with which question number each answer is associated. For answers that include programs or program segments, you may use C or any reasonable pseudocode whose syntax and semantics are clear.

Long answers are not required. The short answer / essay questions are designed to be answered in about one paragraph each, and some can be answered fully in one or two economical sentences.

Don't cheat. This is a closed-book exam. Note the pledge printed in the exam books and sign it if you think it will help.[1] [2]

1. **(20 points total):** In an effort to become known as the next Rob Pike[3], Prof. Feckless C. Coder, PhD (our hero), has been conducting measurements of memory usage and allocation patterns in the Unix and Windows virtual memory systems. He observed that after several hundred processes have been run, real memory is very heavily fragmented, with pages from one process interleaved with the pages from others. He proposes to include a "memory de-fragmenter" in these operating systems that periodically re-arranges the real memory into contiguous chunks (e.g., by copying pages to different locations and updating the MMU).

   1a. (10 points): Do Prof. Coder's measurements seem plausible? That is, would you expect a standard general computing workload (where processes come and go frequently) to produce fragmented real memory on virtual-memory-based operating systems such as these? (If so, explain why. If not, describe the mechanisms used by these operating systems to prevent real memory fragmentation).

   1b. (10 points): What impact on peformance would you expect Prof. Coder's real memory de-fragmentation system to have? That is, what are the expected costs of running it, and what are the expected benefits? Can you conclude whether this idea is a good one based on the information given here, or do you need more data? Explain.

2. **(20 points):** In an effort to become known as the next Nikolai Ivanovich Lobachevsky[4], Prof. Feckless C. Coder, PhD is trying to prove that LRU isn't a very good algorithm for VM performance. Consider a machine with 10 virtual pages (numbered 0-9) and 4 real memory pages (numbered 0-3).

[1] Or sign this pledge if you prefer (but still don't cheat):

> I shall not cause harm to any vehicle, nor to the personal contents thereof; nor through inaction let that vehicle, or the personal contents thereof come to harm.

[2] **Extra Credit A (1 point):** What is the pledge in the previous footnote from?
[3] **Extra Credit B (1 point):** Who is Rob Pike?
[4] **Extra Credit C (2 points):** Who was great Lobachevsky?

2a. (10 points): Assuming a cold start (no pages mapped in main memory), give a reference string of length 8 that *maximizes* the number of page faults with LRU (least recently used) page replacement. Give the state of real memory at each stage in your reference string under LRU.

2b. (10 points): What is the "optimal" virtual memory page replacement algorithm? For each entry in the reference string you gave in the previous answer, give the state of real memory at each stage under the optimal algorithm.

3. (20 points total): Recall that CPU utilization is less than optimal not only if there are too few processes, but also if there are too many. But the optimal number of processes in a virtual memory system depends on several different factors, including the computer configuration (e.g., amount of memory), OS algorithms (e.g., replacement rules, quantum, etc), and the properties of the processes themselves.

3a. (10 points): What one or two properties of the computer configuration, operating system or the processes running most influences CPU utilization if the number of processes is *less* than optimal? Other than increasing the number of processes, what, if anything, can be done in practice to improve utilization? If nothing practical is likely to help, explain why.

3b. (10 points): What one or two properties of the computer configuration, operating system or the processes running most influences CPU utilization if the number of processes is *greater* than optimal? Other than decreasing the number of processes, what, if anything, can be done in practice to improve utilization? If nothing practical is likely to help, explain why.

4. (30 points total): In an effort to become known as the next Kirk McCusick[5], Prof. Feckless C. Coder, PhD is building a backup system for Unix file systems (i.e., those with file attributes and pointers to file blocks stored in i-nodes and file names stored in directories). The backup program is intended to make a copy of a all the files in a directory onto another Unix file system.

4a. (10 points): Prof. Coder's first attempt copied the directories themselves to the other file system verbatim (byte by byte). (Both the files and the directories were copied by creating them on the new file system using the create system call interface and then copying data from the source file system to the target using open, read and write). The copied directories didn't work on the new file system. Why?

4b. (10 points): In fixing the program in question 4a., Prof. Coder also realized that he had failed to copy file attributes. He is considering copying the file's entire inode (byte-by-byte) from the old file system to the appropriate entry in the new file system. Are there any problems with this idea? Explain.

4c. (10 points): After much toil, Prof. Coder finally got a program working that copied entire directory hierarchies from one file system to another. Unfortunately, it makes extra copies of files with multiple links to them. Why might this happen? Can it be fixed?

5. (10 points): Even though LRU is usually a very good algorithm for both file system caching and virtual memory page replacement, many systems use LRU only for the file system and "compromise" algorithms that give higher page fault rate than LRU to manage VM pages. Why would an inferior algorithm ever be used in the VM system? Why would LRU still be suitable for the file system cache?

[5] **Extra Credit D (1 point):** Who is Kirk McCusick?