CS 352 – Prof. Mark **Assignment #6** **Due April 5, 2007**

Problems #1-5

P&H 7.39 [15]

P&H 7.40 [15]

P&H 7.41 [10]

P&H 7.43 [15] <§7.4> Page tables require fairly large amounts of memory (as described in the Elaboration on page 519), even if most of the entries are invalid. One solution is to use a hierarchy of page tables. The virtual page number, as described in Figure 7.20 on page 513, can be broken up into two pieces, a "page table number" and a "page table offset." The page table number can be used to index a first-level page table that provides a physical address for a second-level page table, assuming it resides in memory (if not, a first-level page fault will occur and the page table itself will need to be brought in from disk). The page table offset is used to index into the second-level page table to retrieve the physical page number. One obvious way to arrange such a scheme is to have the second-level page tables occupy exactly one page of memory. Assuming a 32-bit virtual address space with 4 KB pages and 4 bytes per page table entry, how many bytes will each program need to use to store the first-level page table (which must always be in memory)? Provide figures similar to Figures 7.19, 7.20, and 7.21 (pages 512–517) that demonstrate your understanding of this idea.

P&H 7.44 [15] <§7.4> Assuming that we use the two-level hierarchical page table described in Exercise 7.43 and that exactly one second-level page table is in memory and exactly half of its entries are valid, how many bytes of memory in our virtual address space actually reside in physical memory? (Hint: The second-level page table occupies exactly one page of physical memory.)