CS 377 – Operating Systems

Discussion Session 9 Questions

Name: _____

Write your answers individually, without consulting notes, slides, books, or the internet. Be succinct (complete sentences not necessary). **Remember to turn your paper over.**

1. **Segmentation**. How is **segmentation** distinct from **paging** in the context of memory management? How do we combine segmentation and paging in the OS?

   Solution. *Segmentation refers to the logical division of memory into segments, while paging refers to the physical division of memory into equally-sized pages. In a process, there are multiple logical segments, such as code, global variables, a stack, and a heap. Segments are mostly a user-side abstraction, while paging is handled by the OS behind the scenes. In the complete OS, segments are mapped to one or more pages in physical memory – since segments need not be any particular size, a segment may be either smaller or larger than a single page of physical memory.*

2. **Virtual memory**. What are the two main benefits provided by demand paged virtual memory? At a high level, explain how this benefit is provided. What is the main potential downside when using demand paged virtual memory?

   Solution. *Demand paged virtual memory allows running programs that take up more memory than is actually available (i.e., a larger virtual address space than physical address space), and also allow running processes that are not fully loaded into memory. This is done by using the disk to hold pages of memory until they are actually needed. When the data is needed, it is brought back from the disk into main memory (and data it replaces may then be placed into disk). The primary possible downside is one of performance – since the disk is much slower than main memory, too much paging to and from the disk ('swapping') will result in poor performance.*

3. **Page replacement**. What is the role of a page replacement algorithm in demand paged virtual memory? What is an example of such an algorithm?

   Solution. *The page replacement algorithm is responsible for choosing a page to evict from memory on a page fault (that is, when a new page needs to be brought into main memory). Ideally, the page replacement algorithm should minimize the total number of page faults as the machine runs. Example algorithms include random selection, FIFO, and LRU (least recently used).*