(1) How is “dirty flag” used in virtual memory for?

The dirty flag is used to keep track of whether a virtual memory page at a physical memory needs to be saved or not (can be overwritten by another virtual memory page). The dirty flag is reset to ‘0’ when a virtual memory page is loaded to a physical memory page and set to ‘1’ when the page is updated (any page with the dirty flag set to ‘1’ should be saved to the hard drive when it is being replaced by another virtual memory page).

(2) In the virtual memory (as we discussed in the classroom), how many disk accesses can happen in the worst case?

In the worst case, as many as two hard drive accesses can happen.
(3) What is “demand paging”? What is the primary advantage?

The demand paging means that a whole program (or data set) will not be loaded to the physical memory when a program is started. Instead, only the first few pages are loaded to the physical memory, which lets a processor starts running the program, and the other pages are loaded to the physical memory only when they are needed by the processor (implying that the pages no longer needed will be kicked out from the physical memory).

The primary advantages of the demand paging are:

(a) Programs larger than the physical memory capacity can be executed (or “more programs than the physical memory can hold can be executed”).

(b) Programs (especially large programs or programs that need large data) can be started quicker.

(4) How is “segmentation fault” caused?

Segmentation faults are the situation when the offset address (segment offset) exceeds the limit (the size) of a target memory segment. Segmentation faults are usually caused when a pointer points to a wrong memory address or an array index exceeds the size of an array.

(5) What is the advantage of using segmentation?

Since programmers (mostly for assembly programmers) do not have to keep track of the starting memory addresses for each internal component in a program, managing memory addresses would be easier for programmers (thus programs have less program bugs, resulting more reliable programs).