Virtual Memory - Paging

Virtual Address:
- Used by application program
- Consists of a 16 bit segment selector + 32 bit segment offset. (x86 32 bit)
- 16 bit selector specifies page directory and are stored in specific registers

Linear Address:
- Result of virtual/ address and segment translation
- 32 bits Directory Index | Table Index | Frame Offset (2-level paging)

Physical Address:
- Calculated from linear address and paging
- Actual location in physical memory

In JOS, the Virtual Address and Linear Address are the same since page directories are swapped during environment switches. The project 2 spec describes how this is done.
JOS Macros:
- PDX(la) - Page Directory Index based on Linear Address
- PTX(la) - Page Table Index based on Linear Address
- PADDR(va) - Physical Address given Virtual Address
- KADDR(pa) - Virtual Address given Physical Address
- PTE_ADDR(pte) - Physical Address that given page table entry refers to

JOS Functions:
- PageInfo * pa2page(pa) - Returns PageInfo pointer associated with given page’s physical address
- physaddr_t page2pa(*pp) - Returns physical address of page associated with given PageInfo struct
- void * page2kva(*pp) - Returns virtual address of page associated with given PageInfo struct

Questions:

What are internal and external fragmentation?

True or False. A virtual memory system that uses paging is vulnerable to external fragmentation. Why or why not?

What’s an advantage of doubling the page size? (4KB → 8KB)
What’s a disadvantage of doubling the page size?

What type of information does a page table entry store about a page frame? How is this done?

What’s the benefit of having a 2-level paging system?