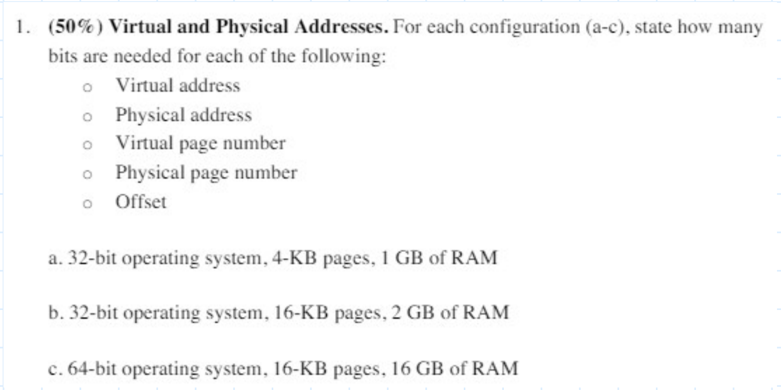
$bit := 1$

$byte := 1$ $KB := 1024\ byte$ $MB := 1024\ KB$ $GB := 1024\ MB$



1. **(50%) Virtual and Physical Addresses.** For each configuration (a-c), state how many bits are needed for each of the following:
   - Virtual address
   - Physical address
   - Virtual page number
   - Physical page number
   - Offset

a. 32-bit operating system, 4-KB pages, 1 GB of RAM

b. 32-bit operating system, 16-KB pages, 2 GB of RAM

c. 64-bit operating system, 16-KB pages, 16 GB of RAM

1) 32-bit OS, 4-KB pages, 1 GB of RAM

$OS := 32\ bit$ $\quad page := 4\ KB = 4096\ byte$ $\quad RAM := 1\ GB = 1073741824\ byte$

$VA := OS = 32\ bit$ $\quad PA := \log(RAM, 2) = 30\ bit$

$offset := \log(page, 2) = 12\ bit$ $\quad VPN := VA - offset = 20\ bit$

$PPN := PA - offset = 18\ bit$

2) 32-bit OS, 16-KB pages, 2 GB of RAM

$OS := 32\ bit$ $\quad page := 16\ KB = 16384\ byte$ $\quad RAM := 2\ GB = 2147483648\ byte$

$VA := OS = 32\ bit$ $\quad PA := \log(RAM, 2) = 31\ bit$

$offset := \log(page, 2) = 14\ bit$ $\quad VPN := VA - offset = 18\ bit$

$PPN := PA - offset = 17\ bit$

3. 64-bit OS, 16-KB pages, 16 GB of RAM

$OS := 64\ bit$ $\quad page := 16\ KB = 16384\ byte$ $\quad RAM := 16\ GB = 17179869184\ byte$

$VA := OS = 64\ bit$ $\quad PA := \log(RAM, 2) = 34\ bit$

$offset := \log(page, 2) = 14\ bit$ $\quad VPN := VA - offset = 50\ bit$

$PPN := PA - offset = 20\ bit$

2.What are some advantages of using a larger page size?

3.What are some disadvantages of using a larger page size?

Pros:

- Larger pages means less pages are needed to cover the same amount of physical memory. Less pages means smaller page table.
- Less page faults.

Cons:

- Greater waste for small data requests. If a program requests a small amount of memory relative to the page size, we could be wasting a lot of space.
- Hit miss penalty worsens. If we miss, it takes more to load in a different page.

2. **(50%) Using the TLB.** As described in your textbook, virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. To speed up this translation, modern processors implement a cache of the most recently used translations, called the *translation-lookaside buffer* (TLB). This exercise shows how the page table and TLB must be updated as addresses are accessed.

   The following list is a stream of virtual addresses as seen on a system. Assume 4-KB pages and a four-entry fully associative TLB with LRU replacement policy. If pages must be brought in from disk, give them the next largest unused page number (i.e., starting at 13).

Given the address stream, initial TLB, and initial page table, show the final state of the system (TLB and page table). Also show for each memory access whether it is a hit in the TLB (H), a hit in the page table (TLB miss, or M), or a page fault (PF).