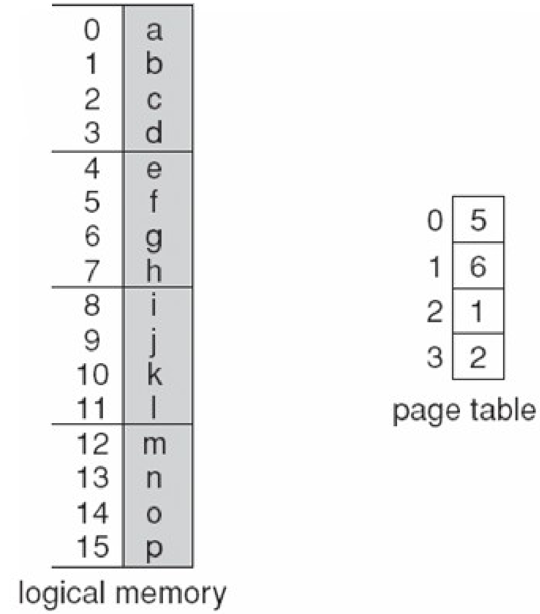
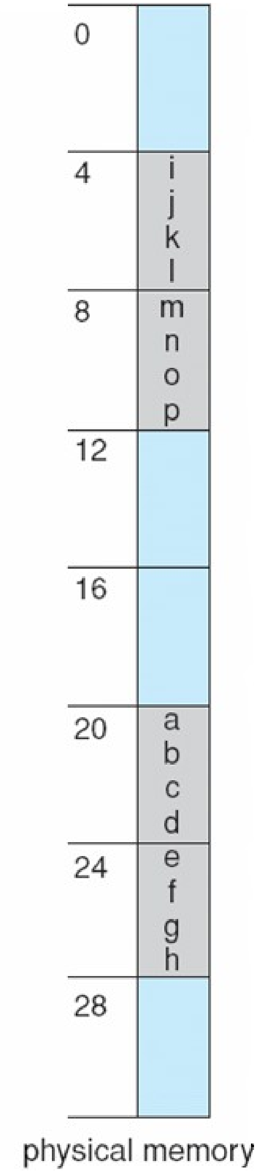
Sample Problem #1 Solution

- We have a machine with 4GiB of RAM
- We have a page size of 8KiB
- We allow processes to have 1GiB address spaces
- How many bits are used for physical addresses?
  - 4GiB of RAM = 2^32 bytes: 32-bit addresses
- How many bits are used for logical addresses?
  - 1GiB of RAM = 2^30 bytes: 30-bit addresses
- How many bits are used for logical page numbers?
  - 1 GiB of RAM = 2^30 bytes
  - 1 page = 8KiB = 2^13 bytes
  - number of pages in address space: 2^30 / 2^13 = 2^17
  - number of bits for logical page numbers: 17

Sample Problem #2 Solution

- 32-byte memory
- 16-byte address space
- 4-byte pages
- 4-bit logical addresses
- 5-bit physical addresses





- What is the physical address corresponding to logical address 6?
- logical: byte **2** in page 1 (i.e., the 3rd byte)
- physical: byte **2** in frame **6** (per the page table)
- therefore: physical address = **6** * <frame size> + **2** = 26

Sample Problem #3 Solution

- Page size: 32KiB
- Logical addresses: 39 bits
- Page table entry size: 8 bytes
- Question: using 2-level paging, how is a logical address split into its 3 components (p1, p2, offset)?
- Answer:
  - How many bits for the offset? 32 KiB page -> 15-bit offset
  - How many page table entries do we need in total? $2^{39} / 2^{15} = 2^{24}$
  - How many page table entries can fit in a page? 32KiB / 8 bytes = $2^{15} / 2^{3} = 2^{12}$
  - How many page table pages do we need? $2^{24} / 2^{12} = 2^{12}$
  - The first-level page table thus fits nicely into a single page that contains 2^{12} pointers to 2^{12} different second-level page table pages. Each such page table page contains 2^{12} pointers to 2^{12} different actual pages.
  - Final answer: p1 = 12, p2 = 12, offset = 15