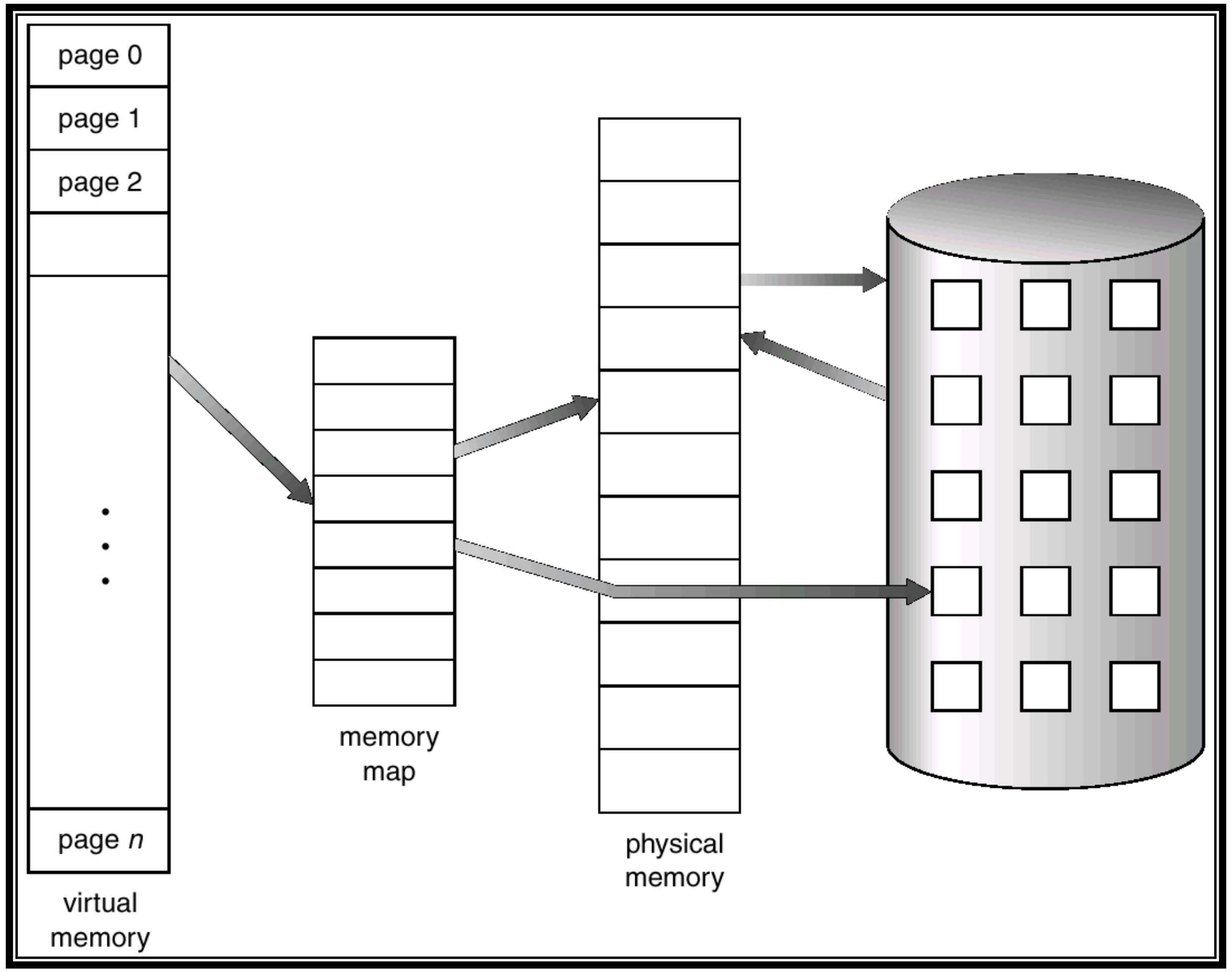
Virtual Memory Management

- Virtual memory is a technique that allows the execution of processes that are not completely in memory.
- One major advantage of this scheme is that programs can be larger than physical memory.
- Further, virtual memory abstracts main memory into an extremely large, uniform array of storage, separating logical memory as viewed by the user from physical memory.

- Virtual Memory is a space where large programs can store themselves in form of pages while their execution and only the required pages or portions of processes are loaded into the main memory.
- This technique is useful as large virtual memory is provided for user programs when a very small physical memory is there.

- Virtual memory involves the separation of logical memory as perceived by users from physical memory.
- This separation, allows an extremely large virtual memory to be provided for programmers when only a smaller physical memory is available.
- Virtual memory makes the task of programming much easier.
- Virtual memory also allows files and memory to be shared by two or more processes through page sharing.

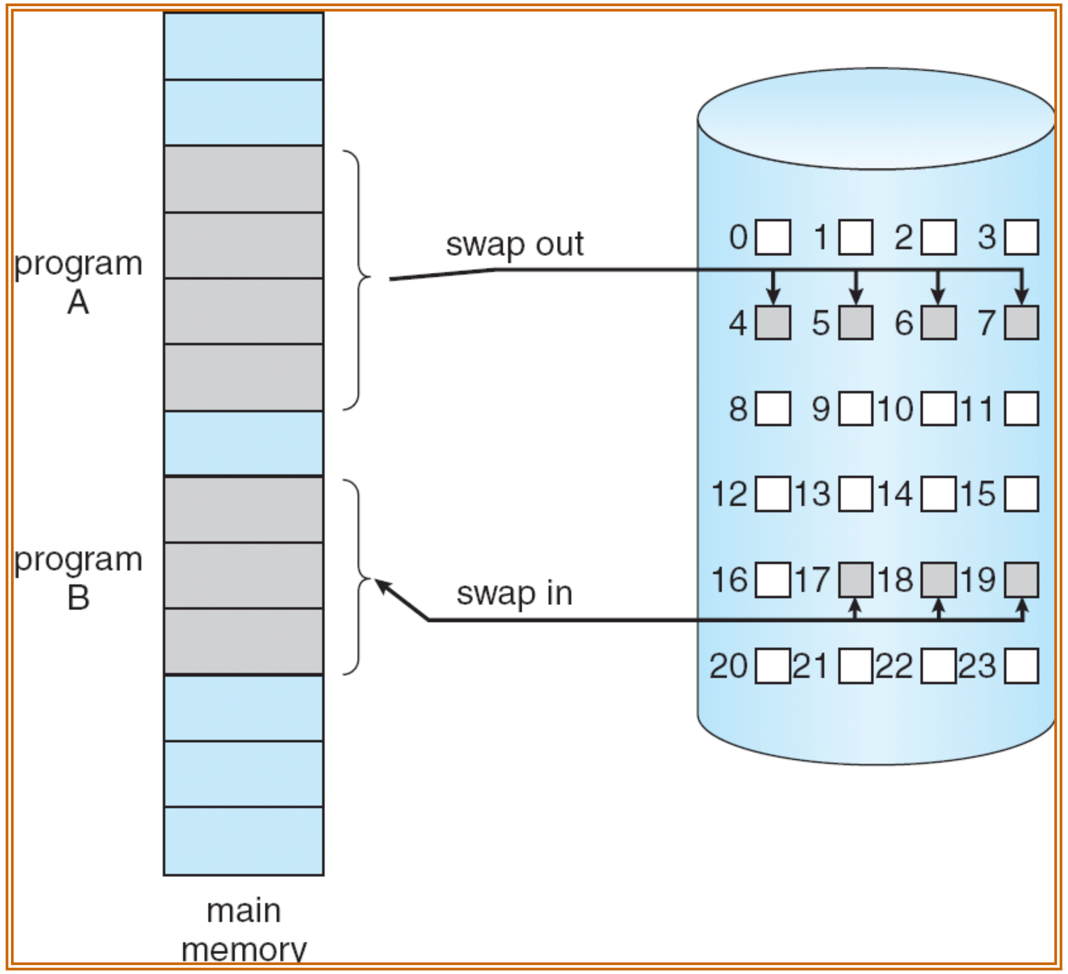
Virtual Memory Larger Than Physical Memory



Demand paging

- Virtual memory is implemented through demand paging
- Bring a page into memory from secondary storage only when it is needed .
- With demand-paged virtual memory, pages are only loaded when they are demanded during program execution.
- Pages that are never accessed are thus never loaded into physical memory.
- A demand-paging system is similar to a paging system with swapping where processes reside in secondary memory.

Transfer of a Paged Memory to Contiguous Disk Space

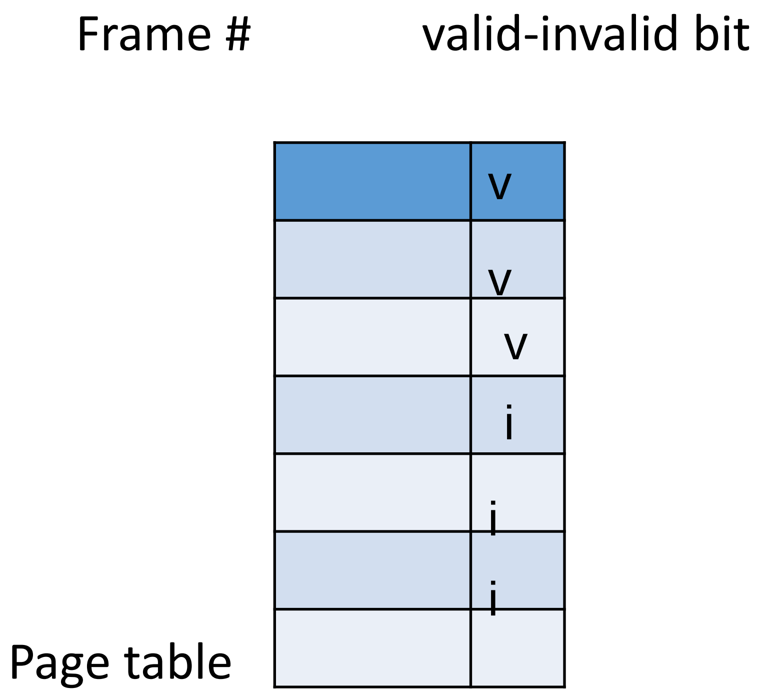


- When we want to execute a process, we swap it into memory.
- Rather than swapping the entire process into memory, we use a **lazy swapper.**
- A lazy swapper never swaps a page into memory unless that page will be needed.
- A **swapper** manipulates entire processes, whereas a **pager** is concerned with the individual pages of a process.

- When a process is to be swapped in, the pager guesses which pages will be used before the process is swapped out again
- Instead of swapping in a whole process, the pager brings only those necessary pages into memory.
- With this scheme, we need some form of hardware support to distinguish between the pages that are in memory and the pages that are on the disk.

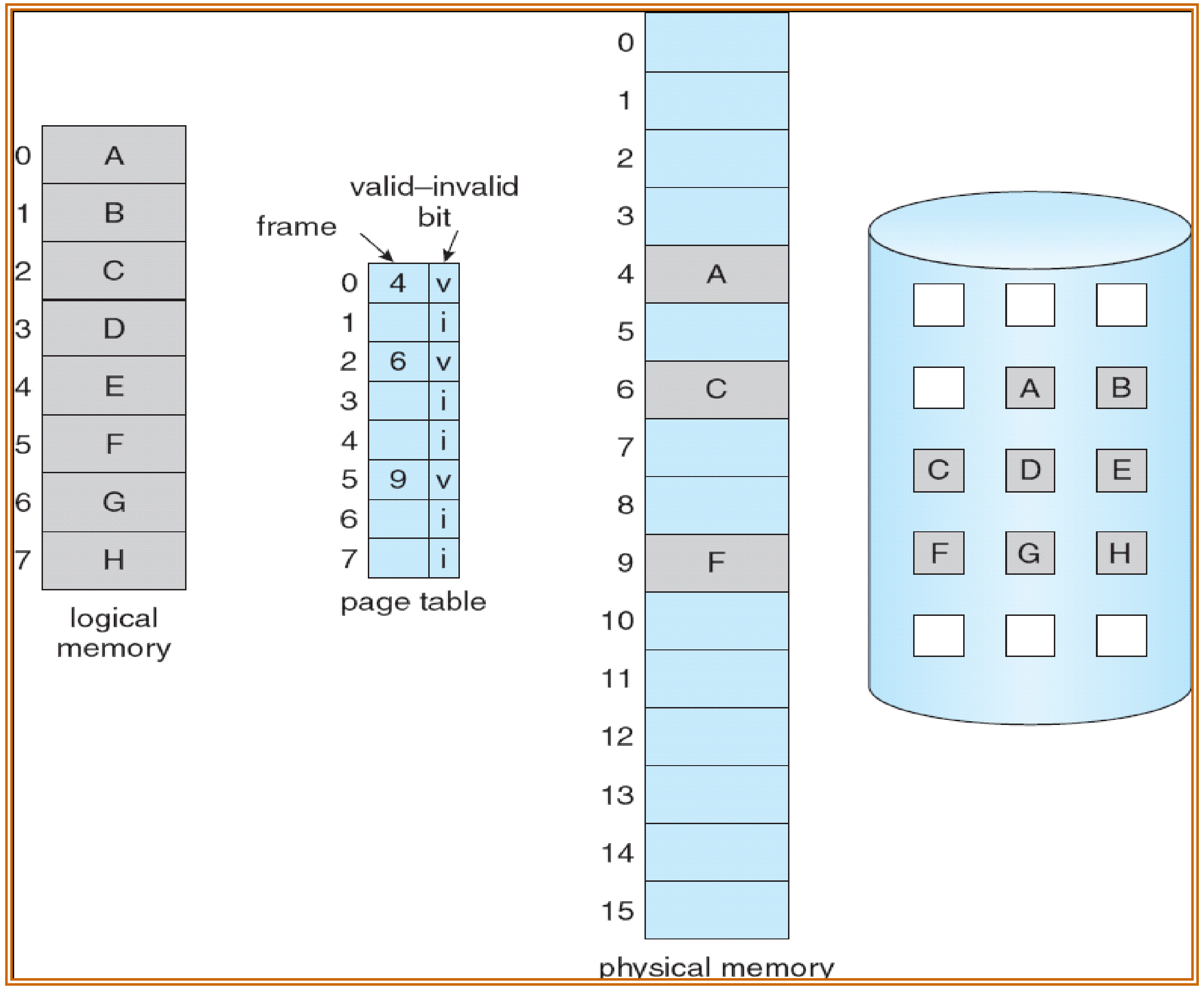
Valid – invalid bit

- With each page table entry a valid–invalid bit is associated
  (**v** ⇒ in-memory, **i** ⇒ not-in-memory)
- Initially valid–invalid bit is set to **i** on all entries



- When this bit is set to "valid" the associated page is both legal and in memory.
- If the bit is set to "invalid," the page either is not valid (that is, not in the logical address space of the process) or is valid but is currently on the disk.

Page Table When Some Pages Are Not in Main Memory



- If the process tries to access a page that was not brought into memory causes a **page-fault trap.**