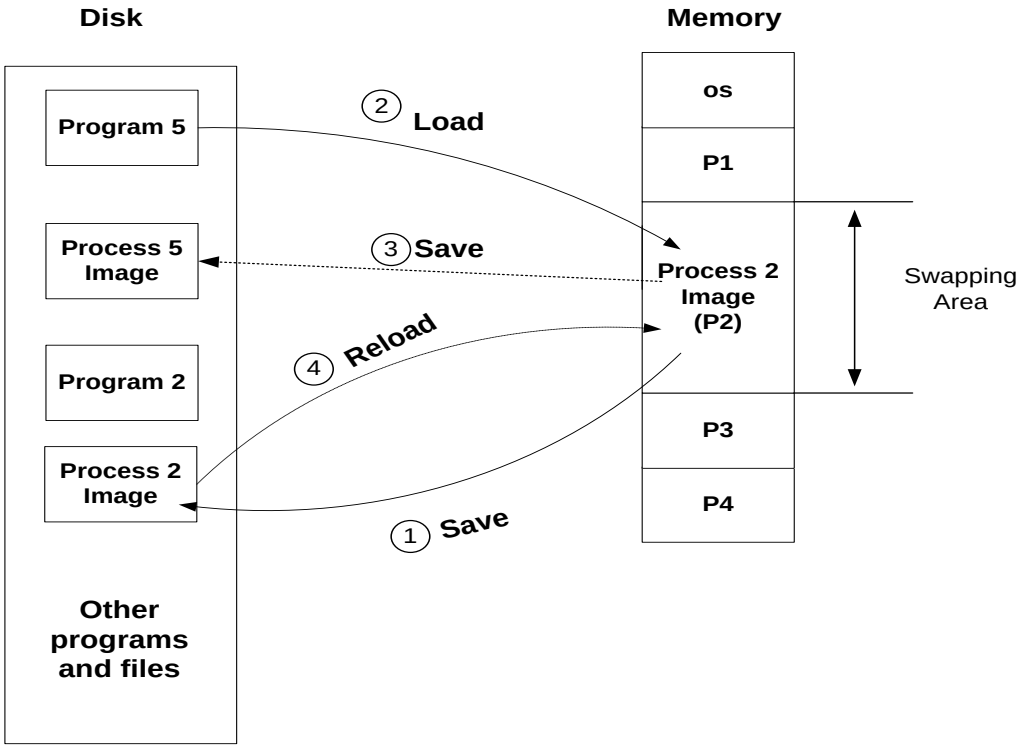
Multiprogramming with Memory Swapping

-When memory is full and new program is required to be executed then OS has to carry out a swapping process as follows:

1. Save resident process image to disk. The process should be in ready or blocked state (or suspended state which is preferred as will be shown later). The memory space of that image becomes free (empty) and may be used for new process. This free space is called “Swapping Area".
2. Load the new program from disk to swapping area and create a PCB for it (create new process).

-When the old process is needed to run again, it has to be reloaded to its original space i.e. to the swapping area, however, after saving the new process to disk.



Fragmentation:

- External Fragmentation: total memory space exists to satisfy a request, but it is not contiguous

- Reduce external fragmentation by compaction

  - shffle memory contents to place all free memory together in one large block

  - Compaction is possible only if relocation is dynamic, and is done at Execution time.

  - I/O problem

    @ Latch job in memory while it is involved in I/O

    @ Do I/O only into OS buffers

- Internal Fragmentation: allocated memory may be slightly larger than requestedmemory, this size difference is memory internal to a partition, but not being used