```
------------------------------------------------------------------------
We use pseudocode below in an attempt to keep the solution brief
and hopefully clear.

SW  = software                  PTA = page table for process A
OS  = operating system          PTB = page table for process A
HW  = hardware                  VPN = virtual page number
MMU = memory management unit    PFN = page frame number

V   = Valid bit                   M = Modified bit
D   = Dirty bit                   U = Use bit
      (can the page be dirtied)
------------------------------------------------------------------------

Process A does a load from address 0x00003C9A.

The 12 last bits of the virtual address correspond to the offset
C9A, while the 20 remaining bits indicate the page number 3.

o [HW/MMU]
  There is an entry with virtual page number (VPN) 3 in the
        TLB, but it is invalid, which results in a
     TLB exception (Read Fault).
     BadVaddr = 0x00003C9A
o [SW/OS]
  There is an entry with VPN 3 in the page table of process A,
     PTA[3].V == 1
  Note that if this was a store the kernel would kill the process because
.
     PTA[3].D == 0
  Frame is 0.
     PTA[3].PFN == 0

  This information is copied in the TLB
        table in entry 1 following the order (1,2,0).
     TLB[1].VPN = 3
     TLB[1].PFN = 0
     TLB[1].V = 1
     TLB[1].D = 0

o [SW/OS]
  If tracking use for page replacement,
  set the Use bit (U=1) in the page table for A.
     PTA[3].U = 1.

o [HW/MMU]
  The instruction is executed again with the new content of the TLB.
  This time there is is a valid TLB hit and the translation occurs.
     TLB[1].VPN == 3 and TLB[1].V == 1.
  If there were a Use bit (U) in the TLB it would get set by the MMU.
     TLB[1].U = 1
  VPN is translated to PFN TLB[1].PFN == 0.

The virtual address is directly translated to the physical address 0x0C9A
```

.

Process A does a store to address 0x00002F08.

The 12 last bits of the virtual address correspond to the offset
F08, while the 20 remaining bits indicate the page number 2.

o [HW/MMU]
  TLB search for valid entry with virtual page number (VPN) 2.
        Fails, so:
     TLB exception (Write Fault)
     BadVaddr = 0x00002F08

o [SW/OS]
  There is a valid entry with VPN 2 in the page table of process A
  but it is invalid and indicates that the corresponding data is on
  sector 0 of the swap partition.
     PTA[2].V == 0
     PTA[2].SW == 0
  This data needs to be loaded in memory (i.e., we have a page fault).

o [SW/OS]
  The CoreMap indicates that there is no free frame in memory.
        As the page replacement algorithm replace pages (4,0,3), page
        4 needs to be evicted.
     Victim = 4.

o [SW/OS]
  The CoreMap indicates that page 4 belongs to process B, with VPN 1.
  The page table of process B indicates that this page is modified (M=1),

  so the corresponding data must be written to the
        swap partition, and that no sector of the swap partition has
        been allocated to this page yet.
     PTB[1].V == 1
     PTB[1].M == 1
     PTB[1].SW == -1

o [SW/OS]
  As the next free sector are (2,3,5), the content of frame
        4 is written to sector 2 of the swap partition.

        The Swap Space map is updated to indicate that Sector 2 is used,
        the entry 1 of the page table of process B is updated to set
        the Modified (M), Used (U) and Valid (V) bits to zero, and the SW
        entry to 2.
     SWAPMAP[2] = X.
     PTB[1].V == 0 (not in memory anymore)
     PTB[1].M == 0 (we cleaned the page)
     PTB[1].U == 0 (clear for future)
     PTB[1].SW == 2 (where to get this page on the next fault)

```
o [SW/OS]
  The content of sector 0 of the swap partition is loaded into
        frame number 4. The corresponding entry in the CoreMap is
        updated to indicate that PFN 4 corresponds to VPN 2 of
        process A.
     Copy Sector 0 from disk to Frame 4 in memory.
     Coremap[4].VPN = 2
     Coremap[4].PGM = A
  Might actually just keep a pointer to the PTE for PTA[2].

o [SW/OS]
  The entry 2 of the page table of process A is updated by
        setting its valid (V) bit to 1 (and setting/changing the PFN to 4
,
        which it already was -- only coincidence).
     PTA[2].V  = 1
     PTA[2].M  = 1  (Write Fault so page will get modified)
     PTA[2].U  = 1  (Write Fault so page will get used)
     PTA[2].SW      (Could leave this alone in case it needs a spot again
)
                     Or free it for someone else to use.

o [SW/OS]
  The entry 2 of the TLB is updated (since entry 1 was
        updated in the previous question and the next replacement/choice
is 2)
  to indicate that VPN 2 corresponds to PFN 4,
  setting the corresponding valid bit (V) to 1 and dirty bit (D) to 1.
     TLB[2].VPN = 2
     TLB[2].PFN = 4
     TLB[2].V   = 1
     TLB[2].D   = 1

o [SW/OS]
  The instruction is executed again, with the new contents of
        the TLB and the instruction succeeds.
     TLB[2].VPN == 2 and TLB[2].V == 1
  If the MMU contains Modified (M) and Use (U) bits they are set
  to one in entry 2.
     TLB[2].M = 1 and TLB[2].U = 1

The virtual address is finally translated to the physical address 0x4F08.
-------------------------------------------------------------------------
```