CS2106 Tutorial 9

AY 25/26 Sem 1 — github/omgeta

Q1. (a.) TLB Miss:

```
if (valid == 0) segfault
else if (resident == 1) insert into TLB
```

Page Fault:

```
if (valid == 0) segfault
writeback victim page into disk if writebit = 1
replace faulting page from swapfile into victim frame
update faulting PTE and set resident = 1
update TLB with PTE
```

(b.) Page 3: TLB Hit → Memory access
Page 1: TLB Miss → PTE Access (Resident) → Update TLB (Evict page 0) → Memory access
Page 5: TLB Miss → PTE Access (Non-resident) → Page Fault (Evict page 0 in frame 4, replace with swap page 15) → Update PTE → Update TLB (Evict page 0) → Memory access

(c.) Best (Memory resident in TLB): 1ns (TLB) + 30ns (Memory) = 31ns
Worst (Page fault): 1ns (TLB) + 30ns (PTE) + 5ns (Disk) + 30ns (Memory) + 5ns (Write-back victim page) = 71ns

(d.) Yes, as on page fault, two separate page table accesses are required.

Q2. (a.) Page table entries = $2^{48-12} = 2^36$ so bytes needed is $2^36 \times 8 = 2^39$ bytes

(b.) 4

(c.) 1 entry (covering the array)

(d.) 1 frame (for the page directory)

(e.) 2 frame (for the 2 different 513 frame regions)

(f.) 513 frames

Q3. (a.) (i) Frame 2

(ii) Set A04 PTE to Non-resident, write back if dirty and remove any TLB entry. Update A08 PTE to resident and set frame 2.

(b.) (i.) Frame 0

(ii.)

Frame	Page	Ref
0	B13	1
1	A31	0
2	A8	1
3	A17	0

(iii.) On page fault, non-resident is not in the inverted page table and we replace the victim page with the non-resident information.