**1.With a linear page table, you need a single register to locate the page table, assuming that hardware does the lookup upon a TLB miss. How many registers do you need to locate a two-level page table? A three-level table?**

Only one register for the first level page directory base register (PDBR)

**2. Use the simulator to perform translations given random seeds 0, 1, and 2, and check your answers using the -c flag. How many memory references are needed to perform each lookup?**

```
ARG seed 0
ARG allocated 64
ARG num 10

PDBR: 108  (decimal) [This means the page directory is held in this page]

Virtual Address 611c:
  --> pde index:0x18 [decimal 24] pde contents:0xa1 (valid 1, pfn 0x21 [decimal 33])
    --> pte index:0x8 [decimal 8] pte contents:0xb5 (valid 1, pfn 0x35 [decimal 53])
      --> Translates to Physical Address 0x6bc --> Value: 08
Virtual Address 3da8:
  --> pde index:0xf [decimal 15] pde contents:0xd6 (valid 1, pfn 0x56 [decimal 86])
    --> pte index:0xd [decimal 13] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 17f5:
  --> pde index:0x5 [decimal 5] pde contents:0xd4 (valid 1, pfn 0x54 [decimal 84])
    --> pte index:0x1f [decimal 31] pte contents:0xce (valid 1, pfn 0x4e [decimal 78])
      --> Translates to Physical Address 0x9d5 --> Value: 1c
Virtual Address 7f6c:
  --> pde index:0x1f [decimal 31] pde contents:0xff (valid 1, pfn 0x7f [decimal 127])
    --> pte index:0x1b [decimal 27] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 0bad:
  --> pde index:0x2 [decimal 2] pde contents:0xe0 (valid 1, pfn 0x60 [decimal 96])
    --> pte index:0x1d [decimal 29] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 6d60:
  --> pde index:0x1b [decimal 27] pde contents:0xc2 (valid 1, pfn 0x42 [decimal 66])
    --> pte index:0xb [decimal 11] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 2a5b:
  --> pde index:0xa [decimal 10] pde contents:0xd5 (valid 1, pfn 0x55 [decimal 85])
    --> pte index:0x12 [decimal 18] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
```

```
      --> Fault (page table entry not valid)
Virtual Address 4c5e:
  --> pde index:0x13 [decimal 19] pde contents:0xf8 (valid 1, pfn 0x78 [decimal 120])
    --> pte index:0x2 [decimal 2] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 2592:
  --> pde index:0x9 [decimal 9] pde contents:0x9e (valid 1, pfn 0x1e [decimal 30])
    --> pte index:0xc [decimal 12] pte contents:0xbd (valid 1, pfn 0x3d [decimal 61])
      --> Translates to Physical Address 0x7b2 --> Value: 1b
Virtual Address 3e99:
  --> pde index:0xf [decimal 15] pde contents:0xd6 (valid 1, pfn 0x56 [decimal 86])
    --> pte index:0x14 [decimal 20] pte contents:0xca (valid 1, pfn 0x4a [decimal 74])
      --> Translates to Physical Address 0x959 --> Value: 1e
```

**Virtual address 611c. We first convert to 15bits binary 0b110000100011100**
**The first 5 bit is 0b11000 (0d24). Since our PDBR = 108 then we look at page table 108, and the 24th entry which is 0xa1 (0b10100001). Notice the first bit is 1 means valid, then we look at convert rest of bit to decimal which is 0b33. Implies our Page table is at page 33. Moreover, the next 5 bits of 611c is 0b01000 (0d8), then the PFN is at page 33, and the 8th entry which is 0xb5 (0b10110101) since the first bit 1 means valid then the PFN is 0b110101. Our offset is 11100, then our physical address number is 0b11010111100 (0x6BC)**

**Virtual address 17f5. We first convert to 15bits binary 0b001011111110101**
**The first 5 bit is 0b00101(0d5). Since our PDBR = 108 then we look at page table 108, and the 5th entry which is 0xd4(0b11010100). Notice the first bit is 1 means valid, then we look at convert rest of bit to decimal which is 0b84. Implies our Page table is at page 84. Moreover the next 5 bits of 17f5 is 0b11111(0d31), then the PFN is at page 84, and the 31th entry which is 0xce (0b11001110) since the first bit 1 means valid then the PFN is 0b1001110. Our offset is 10101, then our physical address number is 0b1100111010101(0x9D5)**

Notice from above translation, it needs 1 memory reference to find PDE by using PDBR and PDN, then use PDE information to execute another memory reference to find PTE. Thus, there are total 2 memory reference. If the PTE is invalid(the first bit is not 1) then it only need 1 memory reference.

**3. Given your understanding of how cache memory works, how do you think memory references to the page table will behave in the cache? Will they lead to lots of cache hits (and thus fast accesses?) Or lots of misses (and thus slow accesses)?**

Memory references won't affect the TLB hit rate. Only the pages size, and the implementation of handle cache entry evict will make difference.