Problem 0
Read book chapter 5.7 to 5.8.

Problem 1 (20 pts)

The following list provides parameters of a virtual memory system.

<table>
<thead>
<tr>
<th>Virtual Address (bits)</th>
<th>Physical DRAM Installed</th>
<th>Page Size</th>
<th>PTE Size (byte)</th>
</tr>
</thead>
<tbody>
<tr>
<td>43</td>
<td>16 GiB</td>
<td>4 KiB</td>
<td>4</td>
</tr>
</tbody>
</table>

For a single-level page table, how many page table entries (PTEs) are needed? How much physical memory is needed for storing the page table?

Problem 2 (20 pts)

There are several parameters that impact the overall size of the page table. Listed below are key page table parameters

<table>
<thead>
<tr>
<th>Virtual Address Size</th>
<th>Page Size</th>
<th>Page Table Entry Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 bits</td>
<td>8 KiB</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>

Given the parameters shown above, calculate the total page table size for a system running 5 applications. If we have a 1GB physical DRAM, what is the maximum number of applications that can be run simultaneously due to the storage issue of page tables?

Problem 3 (60 pts)

Virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following data constitutes a stream of virtual addresses as seen on a system. Assume 4 KiB pages, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

4669, 2227, 13916, 34587, 48870, 12608, 49225
3.1) Given the address stream shown, and the initial TLB and page table states provided above, show the final state of the system. Also list for each reference if it is a hit in the TLB, a hit in the page table, or a page fault. You can assume that the initial TLB is filled from top to bottom (e.g., the top one is the oldest; note that you should always try to fill the empty (invalid) one first in fully-associate TLB).

[Hint: the virtual page number for 4669 is #1, so it misses in TLB as no tag matches. It hits in the page table (i.e., the second entry in the page table is for virtual page #1). However, this is a page fault and the page needs to be brought from disk. Based on the assumption that “If pages must be brought in from disk, increment the next largest page number”, the physical page number for this new page would be 13. Then we update the page table and the TLB. The second entry in the updated page table is (1, 13), and the 4th entry in TLB is (1, 1, 13).]

3.2) Repeat 3.1, but this time use 16 KiB pages instead of 4 KiB pages.

3.3) Show the contents of the TLB if it is direct mapped.