I. Consider the following segment table:

<table>
<thead>
<tr>
<th>Segment</th>
<th>Base</th>
<th>Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>400</td>
<td>600</td>
</tr>
<tr>
<td>1</td>
<td>1200</td>
<td>140</td>
</tr>
<tr>
<td>2</td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>1350</td>
<td>880</td>
</tr>
<tr>
<td>4</td>
<td>2200</td>
<td>96</td>
</tr>
</tbody>
</table>

(a) What are the physical addresses for the following virtual addresses? (2 points)

   (i) 2,20
   (ii) 0,330

(b) What is one advantage for paging over segmentation and vice versa? (2 point)

II. (a) An OS is using two-level paging to implement a 28-bit virtual address space per process. The page size is 256-bytes, and the machine does not have a TLB. Explain the steps involved in looking up the virtual address 0x03bf04d, when all pages are present in memory. (2 points)

(b) For the system above, what is the maximum number of page faults that could be generated in response to a memory access? (2 points)

III. Consider a virtual address system with the following parameters.

- The memory is byte addressable.
- Virtual addresses are 15 bits wide.
- Physical addresses are 13 bits wide.
- The page size is 128 bytes.
- The TLB is fully associative with 16 total entries.

Recall that a fully associative cache has just one set of entries—The tag field is simply the VPN and we need to search the full TLB to check if the VPN we are seeking is in the TLB. In the following tables, all numbers are given in hexadecimal. The contents of the TLB and the page table for the first 16 virtual pages are as follows. If a VPN is not listed in the page table, assume it generates a page fault.
(a) Which bits represent the VPO, the VPN? (1 point)

(b) For the virtual address 0x0422, indicate the physical address. If there is a page fault, enter “—” for the PPN and Physical Address. All answers should be given in hexadecimal. (1 points)

(c) Repeat for address 0x02F1 (1 points)

IV. Consider a process that has been allocated 5 pages of memory: P1, P2, P3, P4, and P5. The process accesses these pages in the following order:

P1 P2 P3 P4 P1 P2 P5 P1 P2 P3 P4 P5

(i) Illustrate Belady’s anomaly by precisely describing the execution of the FIFO page eviction algorithm in two cases: a) where the machine has 3 pages of physical memory, and b) where the machine has 4 pages of physical memory, and by comparing the number of page faults incurred in these two cases. (When the process begins executing, none of its pages are present in memory.) (2 points)

(ii) Show how the LRU page eviction algorithm would work in the same scenarios a) and b) described above. (2 points)