1. A machine has 48-bit virtual addresses and 32-bit physical addresses. Pages are 8KB. How many entries are needed for the page table?

2. The amount of disk space that must be available for page storage (swap area, usually a separate disk partition) is related to the maximum number of processes, \( n \), the number of bytes in the virtual address space, \( v \), and the number of bytes of RAM, \( r \). Give an expression for the worst-case disk space requirements.

3. A computer has 4 page frames. The time of loading, time of last access, and the R and M bits for each page are as shown below (the times are in clock ticks):

<table>
<thead>
<tr>
<th>Page</th>
<th>Loaded</th>
<th>Last ref.</th>
<th>R</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>125</td>
<td>270</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>230</td>
<td>265</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>140</td>
<td>270</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>110</td>
<td>285</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(a) Which page will NRU replace?
(b) Which page will FIFO replace?
(c) Which page will LRU replace?
(d) Which page will second chance replace?
4. Please refer to the handout (available online) about memory for the summary of LRU scheme. Answer the following questions about the different LRU implementations:
   a. In the implementation that uses a hardware counter, could we use a counter of 32 bits instead of 64 bits? What happens if the counter overflows, and become 0 again? How serious this problem is?
   b. Could you prove (or convince yourself) that the nxn bit matrix data structure works, i.e., the row number with the smallest value gives us the page number that is least recently used.
   c. For the aging scheme, can you write an expression to summarize how the counter is updated? More specifically, let C, represents the value of the counter after the i-th timer interrupt, and R1, R2, R3, ... represents the values of R bits at the 1st, 2nd, 3rd clock tick. Can you express Ci in terms of Ci-1 and R1, R2, ... , Ri?
   d. Intuitively, why aging is better than NFU (where the counter is updated simply by adding R bit value to it at each timer interrupt).

5. Suppose that the virtual page reference stream contains repetitions of long sequences of page references followed by a random page reference. For example, the sequence, 0, 1,2,...,511, 431, 0, 1, 2,...,511,332, 0, 1, 2,..., 511 followed by a reference to some random pages, e.g., 431 and 332.
   a. Why the standard replacement algorithms are not effective in handling this workload, for a page allocation less than the sequence length?
   b. If the program is allocated 500 page frames, can you come up a page replacement approach that performs better than LRU, FIFO, or Clock?