1. The discussion of a hashed (inverted) page table indicated that the size of the table was proportional to the size of main memory. Why? Do you also need a regular page table? Why? How large is the regular page table? (12)
2. List and explain as many tradeoffs as you can between writing a program using: (a) one process, (b) multiple processes, and (c) one process with threads, to solve a problem. (12)
3. Suppose that you want to write to a location in memory, and you have a page fault. List and explain all of the steps that have to be taken in order to make that memory reference. Your answer should mention all of the following: disk, page table, TLB, translator, scheduler, page table entry, etc. (“etc.” means that there are other things you also need to mention.) (16)
4. You have processes 1...4 with arrival times and CPU processing requirements as shown. For each of the following scheduling algorithms, show (in a table or diagram) at each time which process is running on the CPU. Compute the average flow time over this set of jobs. You may assume that there are no overhead costs. (16)

<table>
<thead>
<tr>
<th>Process</th>
<th>Arrives</th>
<th>CPU time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2.1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3.2</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>4.3</td>
<td>2</td>
</tr>
</tbody>
</table>

a) RR (quantum = 0.5)
b) SRPT
c) SJF
d) SET (quantum = 0.5)
5. For the following two cases, please either show a complete safe sequence or prove that there isn’t one (12):

<table>
<thead>
<tr>
<th>Process</th>
<th>has-X</th>
<th>has-Y</th>
<th>max-needs-X</th>
<th>max-needs-Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>20</td>
<td>75</td>
<td>50</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>70</td>
<td>50</td>
<td>90</td>
</tr>
<tr>
<td>C</td>
<td>30</td>
<td>10</td>
<td>60</td>
<td>40</td>
</tr>
<tr>
<td>D</td>
<td>50</td>
<td>80</td>
<td>100</td>
<td>220</td>
</tr>
</tbody>
</table>

(a) available: X: 40 Y: 40
(b) available: X: 40 Y: 35
6. The following algorithm, developed by Dekker, is the first known correct software solution to the critical section problem for two processes. The two processes, P0 and P1, share the following variables:

\[
\text{var flag: array [0..1] of boolean; (*initially false*)} \\
\text{turn: 0..1;}
\]

The following program is for process \( Pi \) (\( i=0 \) or 1), with \( Pj \) (\( j=0 \) or 1) being the other process:

\[
\text{repeat} \\
\text{flag[i]:=true;} \\
\text{while flag[j]} \\
\text{do if turn = j} \\
\text{then begin} \\
\text{flag[i]:=false;} \\
\text{while turn=j do noop;} \\
\text{flag[i]:=true;} \\
\text{end;}
\]

\[
\text{...} \\
\text{critical section} \\
\text{...} \\
\text{turn:=j;} \\
\text{flag[i]:=false;} \\
\text{...} \\
\text{remainder section} \\
\text{...} \\
\text{until false;}
\]

Prove that the algorithm satisfies all three requirements (mutual exclusion, progress, bounded waiting) for the critical section problem. (16)
7. Explain what an open (queuing) system is and what a closed system is. Does the scheduling algorithm affect the throughput in a closed system? In an open system? Does it affect the flow time in a closed system? In an open system? Please explain your answers. (16)