(1) What is “process deadlock”?

A process deadlock is a situation in which two (or more) processes are waiting for each other. Since each of them are waiting for the other, they stop running forever (until one of them is killed).

(2) What are the four necessary conditions for a process deadlock to occur?

- Non-preemptive resource
- Hold-and-wait (processes hold resources while they wait for other resources)
- Mutually-exclusive resource
- Circular-wait

(3) Why is it difficult to eliminate the condition of “non preemptive resources” to prevent a process deadlock from occurring?

It is difficult, since converting non-preemptive resources to preemptive resources will ruin the operations of such non-preemptive resources. For example, DVD-burners and printers are preemptive resources. Making them preemptive resources will screw up the results if their use.
(4) Why is it difficult to eliminate the condition of “hold & wait” (by applying “request all after you drop what all what you currently hold” method) to prevent a process deadlock from occurring?

It is still difficult (although not impossible) since applying “request all after you drop what all what you currently hold” method can cause starvation.

(5) Suppose a system has the following status of the resource:

<table>
<thead>
<tr>
<th>Process</th>
<th>Assigned</th>
<th>MAX Need</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

![Available](2)

If process C requests 1 (one) additional instances of this resource, is the system safe? If it is safe, prove that it is safe. If not, prove that it is unsafe.

The request for 1 (one) more instance of the resource by process C still keeps the system safe. It is because there is still a way to finish all three processes after the request is approved by the operating system. The following is how:

(a) After 1 (one) more instance of the resource is assigned to C, we will have 1 (one) more resource available in the system (i.e., “Available = 1”). Process C will have 3 (three) instances of the resource.

(b) After (a), process C can be finished by assigning 1 (one) more instance of the resource and C can finish. As soon as C finishes, all 4 (four) instances of the resources will be available (i.e., “Available = 4”).

(c) With 4 (four) resources, either process A or B can finish.