(1) Why do operating systems have two processor modes (what problems will happen if there is no “user mode” – all programs run in the kernel (system) mode)?

“Without user mode” implies that every process, including user application processes, will be executed in the system (kernel) mode. If it is the case, any user application process can access hardware (hardware resources) without going through the operating system. There is no way for operating systems to manage hardware resources available in a computer system without user mode.

(2) How can “race condition” happen? Show “how” using an example (it is necessary for you to describe your example – just showing an example is not good enough).

Example:

Description:

When P₁ is preempted (interrupted) right after it executes the second instruction (ADD), P₂ may start execution. When P₁ resumes execution of its 3rd instruction, after P₂ executes its three instructions, the content of ‘A’ will be “5”, which is a wrong result. This is a result of the race condition.

Note: your example must use processor instructions (assembly instructions) to show how race condition can happen. I do not think any example that does not use processor instructions can explain how race conditions can happen.
(3) What is “critical section”?

“Critical Section is a section in a program where at most one process can exist at any given time”

OR

“Critical Section is a section in a program in which the machine codes cause race condition”

(4) What is “mutual exclusion”?

The term, “mutual exclusion”, means that if one process exists (or is performing something), the only process prevents any other process to be in the same place in a program (or prevent any other process from performing the same thing). In another word, “one process at a time”.

(5) What “wait” system call to a semaphore exactly performs?

Wait

• If $S > 0$, do $S = S - 1$ then proceed
• If $S = 0$, wait on this semaphore