1. No. if the access is read for both threads, then concurrency error will not occur.

2. `b)`, `c)` and `d)` are true

> **Correct solution**
>
> `c)` and `d)` are true

Notes

Question What does it mean when mutex is held by this thread?

Question What I do know is that `pthread_cond_wait` puts thread to sleep. My question here is, how come the mutex is not held when thread is in a blocked state/sleep?

3. a) Only `b)` causes starvation.

   b) Conditional variable is a queue that allows threads to be put themselves on to sleep (in blocked state) when thread it is not desired using `pthread_cond_wait` function.

   Since there are no threads inside `cv1`, there is nothing to awake using `pthread_cond_signal`.

   So, nothing will occur.

   c) System call is a subset of interrupt caused by user application to switch from user mode to kernel mode to perform previleged operations for the application.

   Interrupt is a signal sent by hardware (e.g keyboard, mouse, hard drive) or software.

   It tells the cpu to stop its activities and execute appropriate part of the operating system.

   Notes

   - I need to review how interrupt works. I had to look up the information.

   Question How does interrupt work?

   d) No. This statment is false.

   User level threads are generated in user-mode without kenerel being aware about it.

Notes

Question What is the difference between user-level thread and kernel-level thread?

Question Why is thread that is generated at user level using procedure call faster than kernel level thread?

Question What is procedure call? How does it work?