Computer Security CS-UY 3923 - **Assignment 1, Part 2**
Sarah Moughal

The vulnerability in my reference monitor was that it didn't defend against race conditions. When multiple threads try to modify the same file at the same time my monitor's behavior is indeterminate. When more than one thread tries to write, read, undo or close with my reference monitor, while one is in the middle of an operation another thread could try and modify the same variable or a variable another thread is depending on to stay the same throughout an operation. For example if one thread tries to write something committing the pending "Hi" to offset 0 and another thread reads 2 bytes from offset 0 immediately after, if there is too little time in between operations the read operation may fail because the end of file may not be updated or my file may not have completed writing the pending "Hi". To address this vulnerability I added a lock to my file object. Before writing, reading, undoing, or closing I acquire the lock and I release the lock at the end of the operation. If an exception is raised I catch it, release the lock and then raise the exception to avoid deadlocking.

I also removed a couple lines of code where I cleared the pending data and offset before overwriting it. Now I just overwrite pending_data and pending_offset.

The last change I made was adding a pending_len attribute for code clarity. Before, I would calculate the pending_len when I needed it so I wouldn't have to update it but I don't have to update it too often so I think the attribute makes it slightly clearer.