Threads

My initial program didn't use locks to account for potential exploits using multithreading. As a result, I started to use locks in readat(), writeat(), close(), open() and undo() to ensure that threads wouldn't be overwriting each others. However, as a result, I now needed to do error checking outside of the program, otherwise I'd get a thread lock. I set a lock variable in __init__ and then acquired it at critical operations and then released the lock when the operation finished.

Error Checking

For readat() and writeat() I implemented checks to make sure that all inputs were valid, so we wouldn't get an exception during the thread lock. In addition, I had to also start tracking other variables like the EOF (in order to make sure we weren't writing/reading past EOF) and if the file was closed.

I tracked EOF by updating it at the very beginning in __init__ by doing readat() and tracking how long the returned string was. Subsequently, I tracked EOF on every successful writeat(). In this case, I only potentially updated EOF if a) the writeat extended EOF and b) if it was actually written into the file (not just set to pending_data)

I tracked if the file was closed simply by setting self.closed == Falsed in the __init__ since every time a file is opened in LPopenfile it returns a new LPFile object. Likewise, I set self.closed == True in closed().

Closed()

Initially, I didn't implement writeat() for when the closed() function was called, so I did that. In addition, I used the closed variable to make sure I wasn't trying to close an already closed file. In addition, I implemented a lock, just in case any multithreaded processes tried to write/read while the file was closed.

Undo()

I initially misunderstood the instructions, believing it wanted us to undo the most recent implemented write to the file, rather than remove the pending_data, pending_offset. In order to fix this I changed that.

openfile()

Initially, I didn't change anything for openfile(), but then I implemented thread protection which then required me to implement exception handling. In order to test for FileNotFoundError and FileInUseError, I use try except statements to check that ahead of time.