1. The command ln path/to/target.txt link1.txt creates a hardlink link1.txt to the file path/to/target.txt. Similarly, ln -s path/to/target.txt link2.txt creates a symbolic link link2.txt. If we open link1.txt and link2.txt, how many inode accesses are required respectively? (hardlink 1pt, symlink 1pt)

Ans:

link1.txt is a hardlink, so there is 1 inode access (find inode directly)

Link2.txt is a symbolic link, so there are four inode accesses. (find the file with content “path/to/target.txt” first, and path, to, target for one access each, total are four accesses.)

2. Suppose two threads do lseek() and write() on the same file descriptor. Describe an example that offset racing occurs (1.5pt), and give a solution to this case (1.5pt).

Ans: Given thread1, thread2, and we mark lseek with lseek_1, lseek_2. Mark write with write_1, write_2. Offset racing when below occurs:

->thread1
lseek_1()
    ->context switch to thread2
    lseek_2()
    write_2()
->context switch to thread1
write_1()

The race occurs because thread1 write the content after the place that thread2 write, not the place of lseek_1(). Hence, the unexpected behavior occurs.

Solution: Using atomic operations, like changing add the flag O_APPEND to write() but it can just write at the end of file. Another solution is pwrite().