1. Pseudocode for Optimal Page Replacement (Main Function)

```
BEGIN
    Initialize `frames[frameCount]` array, fill with -1 (empty)
    Initialize `pageFaults` = 0

    PRINT header for output table

    FOR i = 0 TO pageCount - 1
        `currentPage` = `pages[i]`
        PRINT `currentPage`

        // Assumes call to helper: isPageInFrames()
        IF `isPageInFrames(currentPage, frames, frameCount)` is TRUE THEN
            PRINT "(Hit)"
        ELSE
            // Page Fault
            `pageFaults` = `pageFaults` + 1
            `emptyFrameIndex` = -1

            // Check for an empty frame
            FOR j = 0 TO frameCount - 1
                IF `frames[j]` == -1 THEN
                    `emptyFrameIndex` = j
                    BREAK
                ENDIF
            ENDFOR

            IF `emptyFrameIndex` != -1 THEN
                // Use the empty frame
                `frames[emptyFrameIndex]` = `currentPage`
            ELSE
                // No empty frames, find victim
                // Assumes call to helper: findOptimalVictim()
                `victimFrameIndex` = `findOptimalVictim(frames, frameCount, pages, pageCount, i)`
                `victimPage` = `frames[victimFrameIndex]`
                `frames[victimFrameIndex]` = `currentPage`
                PRINT "(Fault - Replaced " + `victimPage` + ")"
            ENDIF

            // Print current state of frames
            FOR j = 0 TO frameCount - 1
                PRINT `frames[j]`
            ENDFOR
            PRINT newline
        ENDIF
    ENDFOR

    PRINT "Total Page Faults: " + `pageFaults`
END
```

2. Pseudocode for File Locking

```
BEGIN
    FILENAME = "locked_file.txt"

    // 1. Open the file
    fd = OPEN(FILENAME, READ_WRITE, CREATE)
    IF fd < 0 THEN
        PRINT "Error: Could not open file"
        EXIT
    ENDIF

    // 2. & 3. Define the lock
    struct flock lock
    lock.l_type = F_WRLCK     // Exclusive Write Lock
    lock.l_whence = SEEK_SET  // From start of file
    lock.l_start = 0          // At offset 0
    lock.l_len = 0            // For the entire file

    PRINT "Attempting to get lock..."
    // 4. Acquire the lock (blocking)
    IF fcntl(fd, F_SETLKW, &lock) == -1 THEN
        PRINT "Error: Could not get lock"
    ELSE
        PRINT "Lock acquired. In critical section."

        // 5. Critical Section
        SLEEP(5 seconds)

        PRINT "Releasing lock..."

        // 6. Release the lock
        lock.l_type = F_UNLCK // Set to unlock
        fcntl(fd, F_SETLK, &lock)
    ENDIF

    CLOSE(fd)
END
```