41147011S 鄭聿喬 資工115

報告：使用 UNIX 系統呼叫實現行數與單字數計算

目標與背景

本次任務旨在熟悉 UNIX 系統呼叫（如 `fork()`、`execlp()`、`dup()` 和 `wait()`）的使用，並設計一個程式來執行系統指令 `wc`，計算指定檔案的行數與單字數。程式需達到以下要求：

1. 主程式需接收檔案名稱並開啟該檔案，使用 `fork()` 建立子程序執行計算，並以 `wait()` 等待子程序完成。
2. 子程序需開啟另一個檔案來儲存結果，並使用 `dup()` 重導標準輸入和輸出，最後以 `execlp()` 執行 `wc` 指令。
3. 主程式需在子程序執行完成後，讀取儲存結果的檔案並顯示計算結果。

程式設計與實作

詳細程式碼在 myoperation_hw7 有 cat 出來，可以直接看！

執行結果

步驟與指令：

1. 建立測試檔案 `example.txt`，內容如下：

   ```
   Hello world
   This is a test file
   ```

2. 編譯程式：

   ```
   gcc -o wc_program wc_program.c
   ```

3. 執行程式並輸入檔案名稱：

   ```
   ./wc_program
   ```

   輸入：

   ```
   example.txt
   ```

4. 檢視結果檔案：

   ```
   cat result.txt
   ```

   結果：

   ```
   2 5 24
   ```

結果與討論

根據執行結果，程式成功實現以下功能：

1. 主程式以 `open()` 開啟指定檔案並建立結果檔案。
2. 使用 `fork()` 建立子程序並透過 `dup()` 完成標準輸入與輸出的重導。
3. 子程序以 `execlp()` 執行 `wc` 指令，成功計算檔案的行數、單字數和字元數，並儲存到 `result.txt` 。
4. 主程式在等待子程序完成後，正確讀取並顯示結果。

學習收穫：

- 深入理解 UNIX 系統呼叫的基本操作，特別是檔案描述子的管理與子程序間的流程控制。
- 學會利用 `dup()` 和 `execlp()` 等工具實現 I/O 重導與系統指令的執行。
- 培養對程式錯誤處理的習慣，確保程式穩定運行。

改進建議

1. **輸入檢查**：檢查檔案是否存在或是否為空，並給出提示。
2. **更多選項**：讓使用者選擇是否僅顯示行數或單字數。
3. **程式模組化**：將檔案操作、子程序管理與結果顯示分別模組化，提升程式的可讀性與可維護性。

結論

本次實驗成功運用 `fork()` 、 `execlp()` 、 `dup()` 和 `wait()` ，完成了指定檔案行數與單字數的計算任務，並強化了對 UNIX 系統呼叫的實際應用能力。