Lab8 實作流程：

1. 首先我們知道原始檔中含有
   - 1-01/ 1-02 / … / 1-52
   - 2-01 / 2-02 / … / 2-52
   - 3-01 / 3-02 / … / 3-52
   - 4-01 / 4-02 / … / 4-52
   - 5-01 / 5-02 / … / 5-52

   合計 52*5 個 txt 檔

2. 接著我們要在我們的父程序中產生五個子程序:
   - 第一個子程序負責 1-01 到 1-52 的 txt 檔
   - 第二個子程序負責 2-01 到 2-52 的 txt 檔
   - 第三個子程序負責 3-01 到 3-52 的 txt 檔
   - 第四個子程序負責 4-01 到 4-52 的 txt 檔
   - 第五個子程序負責 5-01 到 5-52 的 txt 檔

   /* 可以用 for 實現，且五個子程序同時運行 */

3. 然後讓每個子程序依序打開自己負責 01 到 52 的 txt 檔
   /* 可以用 for + sprintf() + fopen() 實現 */

4. 當每次打開一個 txt 檔時，依序對每天的 96 個用電量做加總後存入 sum.txt
   /* 直接呼叫 void accumulation() */

5. 所以一個子程序總共會呼叫 52*7 次的 accumulation()，有五個子程序所以 accumulation()總共會被呼叫 52*7*5 次，**如果有兩個以上的子程序同時去呼叫 accumulation()就有機會發生 Race condition，**所以我們要避免這種情況

6. 如何避免 → 利用以下函式 (定義在 tellwait.c 內)
   - TELL_WAIT(void)
   - TELL_PARENT(pid_t pid)
   - WAIT_PARENT(void)
   - TELL_CHILD(pid_t pid)
   - WAIT_CHILD(void)

7. 子程序呼叫 accumulation()前先等待父程序，等父程序說可以時候才呼叫 accumulation()，而程式中有五個子程序所以只有最

快那個可以先呼叫, 其他子程序要等到被父程序告知後才可
以去呼叫 accumulation()
/* 不斷 Loop 直到都呼叫完畢,所以這部分父程序及子程序的步驟及迴圈次數
如下 */

- 父： TELL_CHILD(pid[n]) → WAIT_CHILD() 【5*52*7】
- 子： WAIT_PARENT() → accumulation(d_sum) →
  TELL_PARENT(getppid())【52*7】

8. 最後程式會把 5 年全部的電力消耗總和儲存在 sum.txt,
   請抓取總和除上所有天數 → 得到平均每日耗電量(Day_Average) = 4743