The requirements of program are as follow

1. The program will make use of the POSIX system I/O calls and pthreads library discussed in lectures.

必須要用 POSIX system I/O calls and pthreads library

2. You can only use open(), close(), pread(), write(), and the pthreads library function calls for multithreading.

只能用 open(), close(), pread(), write(), and the pthreads library

3. NOT permitted to use any C++ nor C standard I/O library functions nor cin nor cout.

不能用 C++ nor C standard I/O library functions nor cin ， cout

4. The program cannot use any C++ nor C standard library functions for string operations.

不能用 C++ nor C standard library String 功能

5. The program will have 2 pthreads in a Producer Consumer pattern using a stack for interthread communication.

程式會用遵循 Producer Consumer 模式創造兩個用堆疊來做內在執行緒溝通的 pthread

6. The mainline thread opens the input file to get a file descriptor, and it also creates the 2 pthreads.

The main thread waits for each pthread to complete its work by using pthread_join()

and the main thread will subsequently generate all output to screen using write().

主線程打開 Input 檔案然後創造兩個 pthread

主執行緒用 pthread_join()會等待其他執行緒完成他們的工作，然後會用 write().各產出 OUTPUT

7. The first pthread, the Producer, reads the input file of integers and pushes each integer on to a stack.

Meanwhile, the second thread, the Consumer, pops the integers from the stack

and keeps track of the minimum integer of all integers popped so far from the stack.

Producer 執行緒會讀 INPUT 檔案然後 PUSH 數字到 STACK，同時 Consumer 執行緒 會從 STACK POPS 這些數字

然後從這些數字被 POP 的找到他們的最小值。

8. After all 10,000 integers have been popped, the minimum integer is obviously the global minimum of the file.

You may use any data structure for the stack, but the stack can only hold a maximum of 100 integers at any time.

在一萬個數字被 POP 後，最小的數字就會出現。你可以用任何資料結構寫你自己的 STACK 結構，但唯一要求是這個 STACK 最大只能一次容納 100 整數。

9. Since the stack is a shared data structure between the Producer and Consumer threads, any code that accesses the stack is a critical section

and you will use "pthread mutex" function calls for the entry and exit sections in both the Producer and Consumer threads.

因為這個堆積是分享結構夾在 Producer and Consumer 之間，所以這個堆積是個臨界區段

所以你需要在 Producer and Consumer 使用 pthread mutex

10. The Consumer Thread will store (to a global integer array of 10 elements) the running current minimum integer so far of all integers popped after each successive 1,000 integers have been read from the file.

Consumer 執行緒每從檔案中讀取然後 POP 1000 整數後，就會開始會找最小值，然後把這個最小值儲存到 int global [10].

10. The program will have to use either pthread_yield or sched_yield functions in the Producer and Consumer Threads.

Producer and Consumer 得執行緒需要使用 pthread_yield 或 sched_yield

This global integer array will be used later by the main thread to write the output results in item.

OUTPUT: All thread output goes to the screen (standard out).

The Main thread writes 10 lines of information

(one line each for every minimum number generated by the Consumer Thread in item above).

Please use the following format for each line:

Output:

Minimum integer after 1000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 2000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 3000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 4000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 5000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 6000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 7000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 8000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 9000 integers = \t PLACE THE MINIMUM HERE \n

Minimum integer after 10000 integers = \t PLACE THE GLOBAL MINIMUM HERE \n