struct producer\_data {

int drone=0, producer\_id;

// drone用來統計各個producer生產的數量，而producer\_id用來判斷producer是哪一個

};

struct dispatcher\_data {

int dispatcher\_id,component\_num[3];

// component\_num用來統計dispatcher各個零件生產的數量，而dispatcher\_id用來判斷dispatcher是哪一個

};

全域變數

waiting = 0,total\_drone=0, features, component,producer\_condition[3][3];

// waiting==0為dispatcher在跑; waiting==1為producer在跑

// total\_drone為統計所有producer生產的總量和

// producer\_condition為目前producer持有零件的狀況，producer\_condition[producer0~2][哪一個零件]，值為1代表擁有

一開始先讀檔，argv[1]為features，argv[2]為seed，我會先判斷如果features不為1或0，且seed的範圍不是0到100，那我會輸出input error。

先初始化兩個鎖，建立features+3個thread

基礎功能:

dispatcher:

如果total\_drone < 50，就一直跑以下功能，我是用!pthread\_mutex\_trylock(&dispatcher\_work)它會幫我判斷thread有沒有鎖住，如果沒有它會幫我鎖上，然後再判斷waiting的值還有total\_drone < 50 → 因為dispatcher每跑一次迴圈就會unlock，所以當thread已經lock住了，但是還沒判斷waiting的值，這之間如果剛剛好producer的thread剛跑完，那這樣dispatcher就會繼續跑，如此當total\_drone已經等於50了，但是dispatcher還會多輸出一行。

之後我會判斷features值為何，為0代表是基礎功能，分配零件的部分我是使用component=rand()%3，藉此得到0~2的數值，那我會丟到一個check1的function，它會判斷producer\_condition[0~2][component]是不是都為1，是的話代表要再一次component=rand()%3，也因此要再一次丟進check1裡面，如果不是就return value，如此零件的分配就設置完成了，之後waiting=1換producer工作。

producer:

如果total\_drone < 50，就一直跑以下功能，我是用!pthread\_mutex\_trylock(&producer\_work)它會幫我判斷thread有沒有鎖住，如果沒有它會幫我鎖上，因為producer1不用拿到aircraft，所以我先判斷component為1和producer的id為0，如果是的話，代表此次工作失敗waiting為1，unlock，換另外一個thread繼續跑。不是的話，判斷producer\_condition[…][component]是否為1，如果是的話，代表此次工作失敗waiting為1，unlock，換另外一個thread繼續跑。不是的話，將producer\_condition[…][component]設為1，工作成功，然後判斷producer是不是都拿到3個零件(producer1則判斷是否拿到除了aircraft的零件)，是的話將所有的零件(producer1除了aircraft的零件)都設為0，drone跟total\_drone都++，之後waiting=0，unlock。

進階功能:

dispatcher:

基本上和基礎功能差不多，判斷features值為何，為1代表是進階功能，因為進階功能是兩個dispatcher，dispatcher1是生產前面兩個的零件，2為生產後面兩個的零件，至於dispatcher\_id我是用0或1去存，但是因為這種情況有可能dispatcher1或2在執行時，可以生產的零件，所有的producer都已經擁有了，所以我多加一個判斷看看當dispatcher1在執行時或是dispatcher2在執行時，它們可以生產的零件是不是所有的producer都已經擁有，如果有waiting=0，unlock，如果沒有就繼續以下程式，分配零件的部分我是使用component=rand()%2+dispatcher\_id，因為dispatcher1是生產前面兩個的零件，2為生產後面兩個的零件，那因為dispatcher\_id我是用0或1去存，也因此這個程式碼能讓我得到0,1或是1,2這樣的組合，之後我會丟到check2的function，一樣它會判斷producer\_condition[0~2][component]是不是都為1，是的話代表要再一次component=rand()%2+dispatcher\_id，也因此要再一次丟進check2裡面，如果不是就return value，如此零件的分配就設置完成了，之後waiting=1換producer工作

producer:

和基礎功能一模一樣。

最後在main function輸出dispatcher生產3個零件的數量以及producer生產的數量

執行方式：

g++ s1091447\_03.cpp -lpthread -o out

./out [參數0 or 1] [參數0~100]

有完成:

基礎和進階所有功能