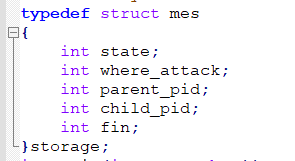
用一個struct來當作父子行程間的共用位置。



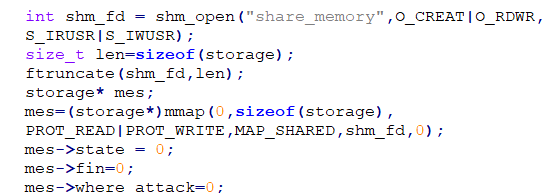
State: 判斷該做什麼

Where\_attack: 攻擊哪個位置

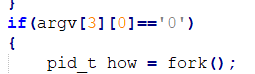
Parent/child pid:父子行程的pid值

Fin:判斷是否炸完

使用shm\_open跟mmap來設定父子行程的共用位置，並使用ftrumcate配置記憶體空間避免bus error，再進行初始化。



如果是argv[3]是0，就進入基本模式並執行fork()產生一個子行程。



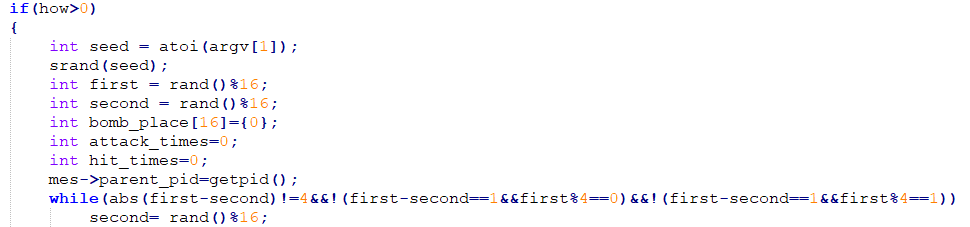
若fork()的值>0，就進去父行程，設定亂數並產生船的位置，將父行程的pid存入mes。

船的產生方式為先隨機產生一個0~15的數字，放到4\*4平面上，再產生另一個值，判斷他是有在前一個值的四周，若是就保留，否則就重新產生。

Bomb\_place:記錄炸過的位置。

Attack\_time:炸的次數。

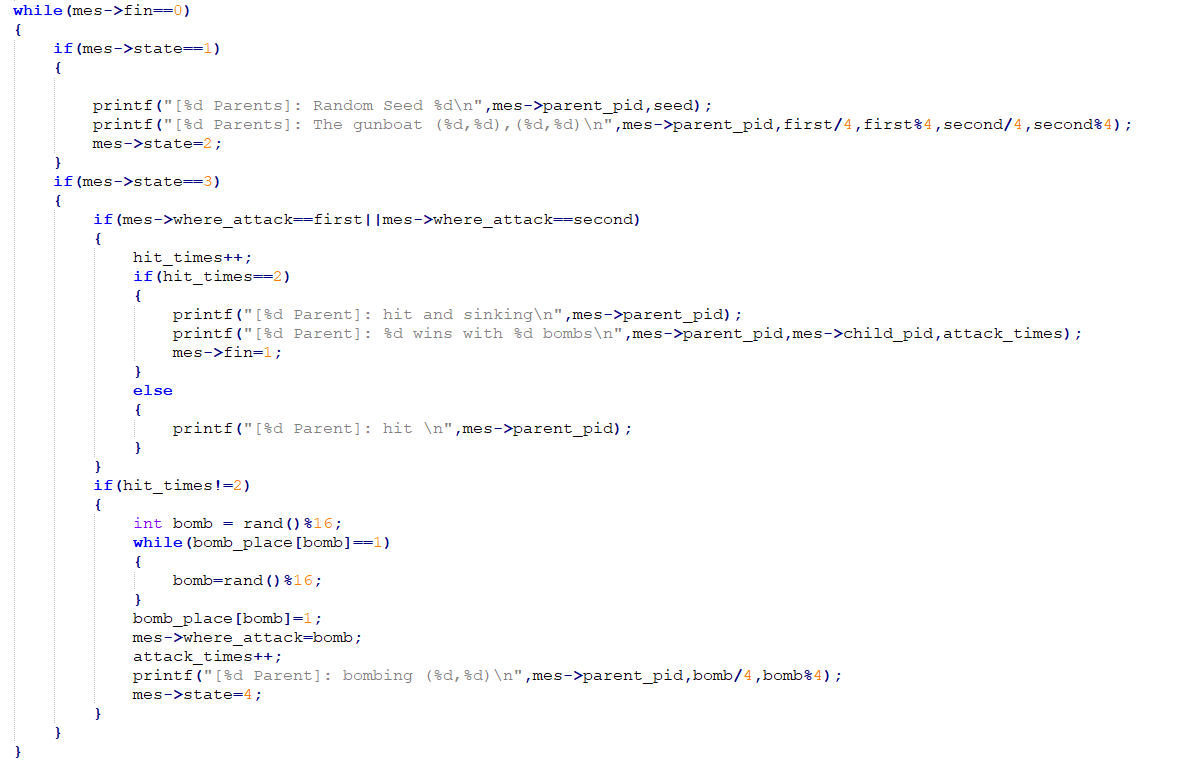
Hit\_times:child炸中的次數。



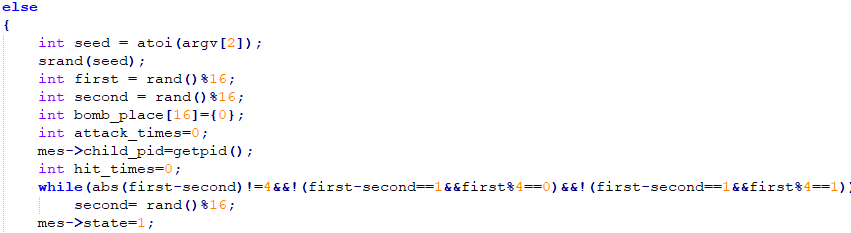
State 1 為輸出parent船的種子及位置並切換到state 2

State 3 為child攻擊後確定是否攻擊到，若攻擊完後船被完全破壞，就輸出child勝利及相關輸出，反之，則將攻擊位置，存入mes並切換到state 4。

攻擊位置的方式，是先產生一個0~16的亂數，並從mes中的bomb做判斷，若是該位置為1，則繼續產生，反之將其設為下次攻擊目標，並將bomb中的奇位置設為1。



接著是子行程，一樣先設定亂數後產生船的位置，將子行程的pid存入mes。



State 2 為輸出child船的種子及位置並切換到state 3

State 4 為parent攻擊後確定是否攻擊到，若攻擊完後船被完全破壞，就輸出parent勝利及相關輸出，反之，則將攻擊位置，存入mes並切換到state 3。

