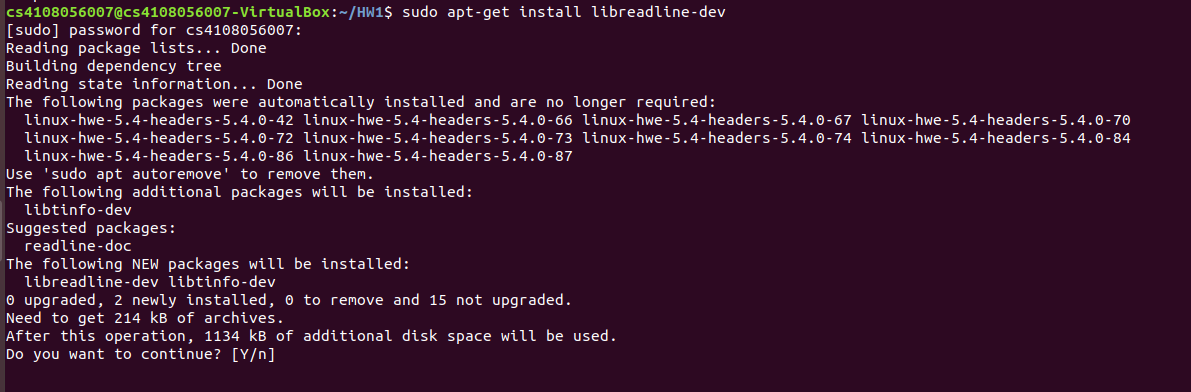
**[README]**

真正的執行過程請見影片

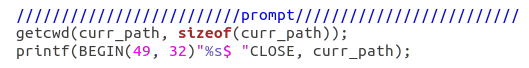
**[前置作業]**

先安裝readline



**[基本功能]**

1. 印出prompt，其中BEGIN和CLOSE是定義prompt的顏色(我是設綠色)

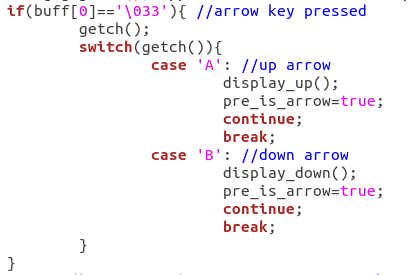




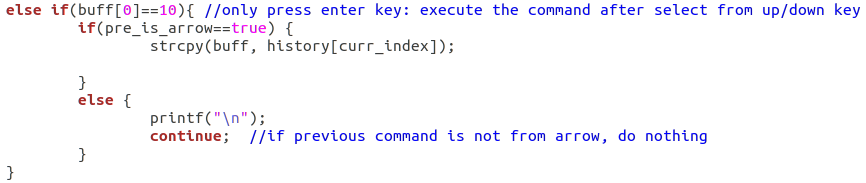
結果:



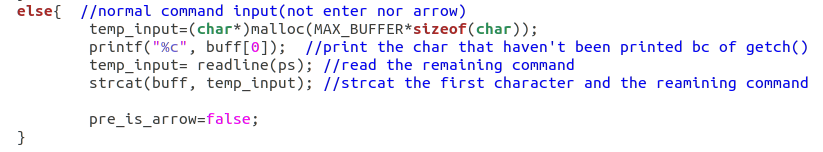
1. 讀第一個字元，此時使用getch()，讓輸入的字先不顯示在鍵盤上，若:
2. 第一個字元為ESC，代表是輸入上/下鍵，所以要再把剩下兩個字元吃掉，並呼叫display\_up()/display\_down() 去印出，history的command



1. 第一個字元為ENTER，去執行剛剛按上下鍵得到的history command

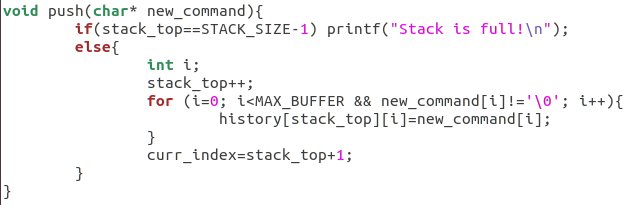


1. 第一個字元非前兩者(代表是一般的指令)，用readline讀入剩下的字元，並合併到buff string

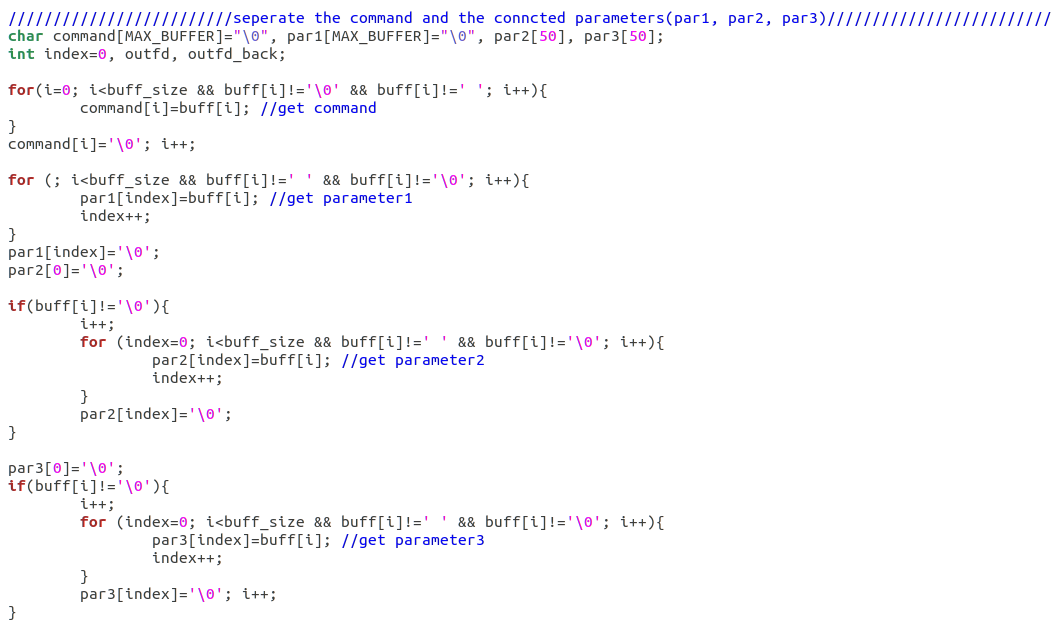


1. 把要執行的指令加到history stack (因為是記錄歷史，所以stack只會push)





1. 將buff裡的指令和參數分開



1. 接著就是用strcmp/strncmp去判斷是哪個指令(程式碼有點長，我就直接放結果，可以去看code的註解)
2. pwd



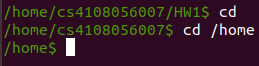
1. echo "string" 或是 echo string 或是echo $環境變數



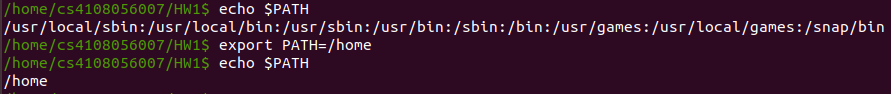




1. cd 或是 cd 絕對路徑



1. export 環境變數=要設定的東西



1. 多一個exit指令去離開這個shell



1. external command: 用child process去執行execlp()



1. 執行過程的影片: <https://youtu.be/BmtSahRHvwY>

**[查詢歷史]**

如基本功能那裡介紹的，會把歷史指令放到一個stack array，然後按上下鍵時就會去讀stack裡面的內容

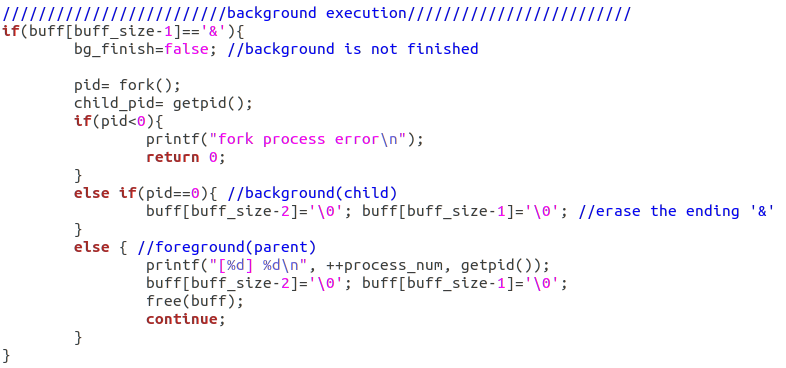
執行過程的影片: <https://youtu.be/snf7EfJDUEE>

影片錄不到上下鍵，所以我把影片key的東西標出上下鍵，如下:



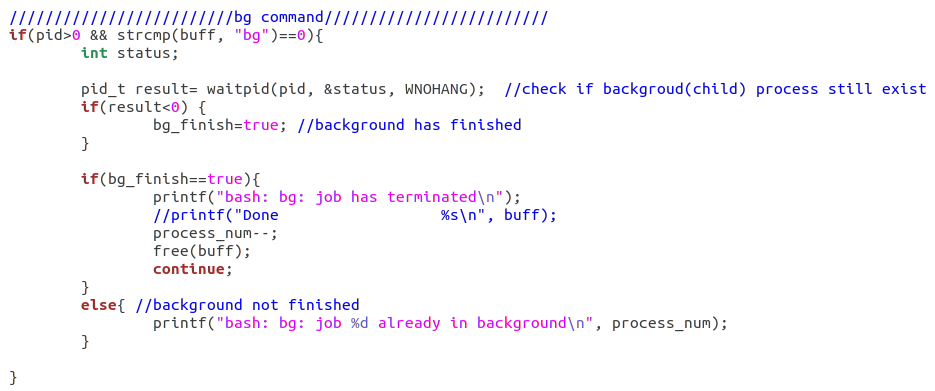
**[background execution]**

1. 如果是'&'結尾的，就fork child process去執行我的background，parent process則是去重新input一個新指令

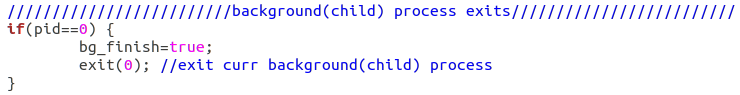


1. bg command會先用waitpid檢查background是否結束 (參數WNOHANG會立即回傳result，而不是像一般的wait會等child)

再依照結果去印出 已結束或還在background



1. 每個創立的background(child) 都會在該輪while迴圈的最後結束掉



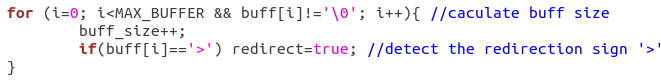
1. 執行影片: <https://youtu.be/fDtYW5c8Wy4>

**[Output redirection]**

1. 會先判斷輸入的指令中，有沒有>>或是>

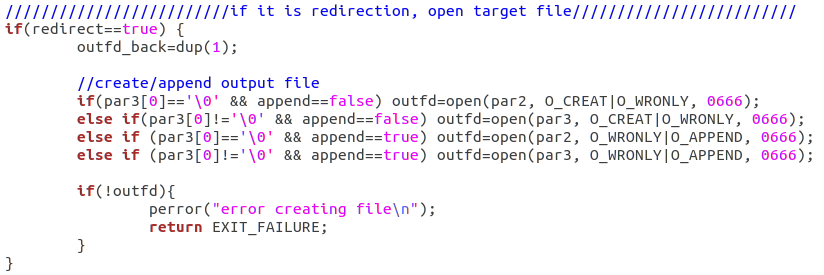
> : redirection==true, append==false

>> : redirection==true, append==true



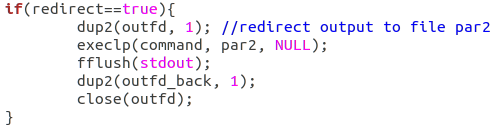


1. 用剛剛解讀出的target file去create/append file



1. 將那些stdout是screen的指令，都用dup2()改成寫入file，寫完後再把stdout改回預設的screen

(參考: https://blog.csdn.net/weixin\_44718794/article/details/106620448)



1. 執行結果

