題目Ａ

adduser stua

adduser stub

groupadd csie

usermod –a –G csie stua

usermod –a –G csie stub

cd /home

mkdir /home/project

cd /home/project

touch program

exit

sudo –s //切換回root，並輸入密碼，才能使用chown指令

student00

chown –R root:csie /home/project

su stua

vim /home/project/program

stua無法修改program，表示只能讀，不能寫入和執行的權限，因為program的權限為644，只有檔案擁有者可以寫入。

題目Ｂ

rm –R /home/project

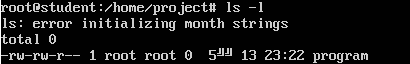
umask 002

mkdir /home/project

cd /home/project/

touch program

ls –l



Program的群組權限由原本只有可讀，變為rw-可讀可寫，若使用者建立為『檔案』則預設『沒有可執行( x )權限』，亦即只有 rw 這兩個項目，也就是最大為 666 ，若使用者建立為『目錄』，則由於 x 與是否可以進入此目錄有關，因此預設為所有權限均開放，亦即為 777，而此範例是新建立一個檔案，檔案預設權限666 XOR 002=664 (-rw-rw-rw-) – (--------w-)=-rw-rw-r。

cd

chown –R root:csie /home/project

su stua

vim /home/project/program

stua here!

exit

su stub

vim /home/project/program

stub here! //接著輸入此行

cat /home/project/program //查看檔案內容



發現當切換到stua 和stub兩個使用者，都可以共同編輯project資料夾下的檔案，所以這樣是對的！因為program檔案權限現在是664，所以檔案擁有者可以編輯與寫入，同群組的使用者也可以編輯與寫入。

題目Ｃ(1)

sudo –s

mkdir /home/project\_new

chmod 1777 /home/project\_new

su stua

vim /home/project\_new/programa

exit

su stub

vim /home/project\_new/programb

exit

測試以stua是否能開啟，刪除，修改programa及programb?

su stua

rm /home/project\_new/programa //可開啟與修改與順利刪除

rm /home/project\_new/programb //無法刪除



vim /home/project\_new/programb //發現可讀，但是無法修改儲存，並出現readonly標示。

Macintosh HD:Users:peterlu:Desktop:螢幕快照 2014-05-14 上午1.43.32.png

題目Ｃ：(2)

mkdir /home/project\_new2

chmod 777 /home/project\_new2

su stua

vim /home/project\_new2/programa

exit

su stub

vim /home/project\_new2/programb

exit

測試stua是否能開啟、刪除、修改programa及programb?

以777權限的資料夾，以stua登入，有開啟刪除修改programa檔案的權限，而對不是自己建立的programb檔案，可以讀取和刪除取，不能修改，但可以強制用:wq!做強制修改。

chmod 1777與 chmod 777比較：

1777對於檔案多了一層保護，不會讓其他人隨意更改與刪除，而777就是將權限單純修改，無保護功能，還是可以做強置修改和刪除其他所有著的檔案。  
1777比較符合本題目需求。

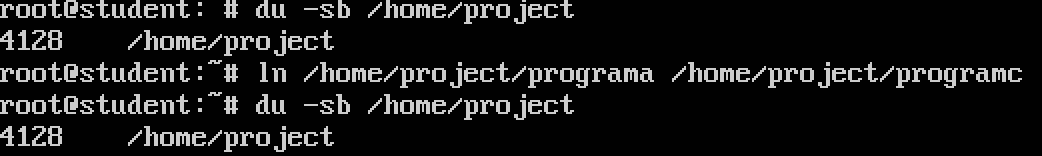
題目Ｄ(1)

cp –a /home/project\_new/programa /home/project

cp –a /home/project\_new/programb /home/project

du –sb /home/project

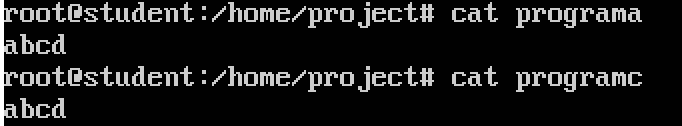
ln /home/project/programa /home/project/programc

du –sb /home/project

cd /home/project

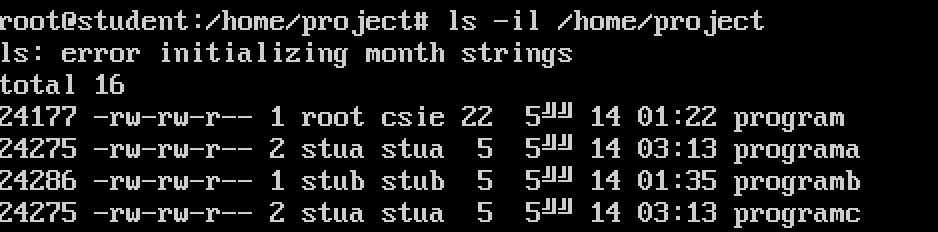
cat programa //查看programa和programc是否一樣

cat programc



//沒錯，兩者資料相同

ls –il /home/project



//programa 和 programc 兩個檔名都連結到24275這個inode號碼，其檔案的權限/屬性完全一樣，因為這兩個黨名其實是一模一樣的『檔案』，而第二個欄位都為二，意義為：『有兩個檔名連結到這個inode號碼』； program和programb第二個欄位都是1。

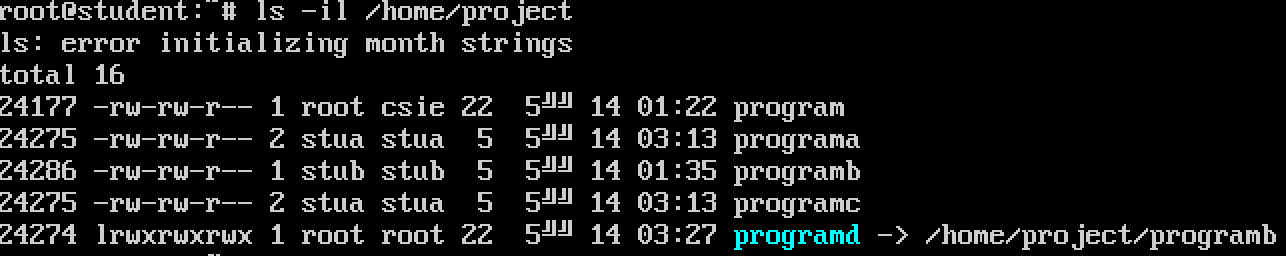
題目Ｄ(2)

ln –s /home/project/programb /home/project/programd

du –sb /home/project



//由原本4128->4150，容量變大了

ls –il /home/project

//programb與programd的inode不同，第二個欄位也就不會加一，各自為一，programd這個檔案會讓資料的讀取指向他link的那個檔案的檔名（programb）。

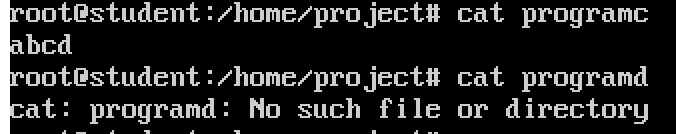
rm /home/project/programa

rm /home/project/programb

cd /home/project

cat programc

cat programd



//programc可以開啟因為programa使用hard link設定連結檔時，磁蝶的空間與inode的數目都不會改變，hard link只是在某個目錄下的block多寫入一個關聯資料， 實體連結在刪除原始連結檔，捷徑還是可以連結到正確資料，programd因為採用Symbolic Link，由於只是利用檔案來做為指向的動作，所以當來源檔被刪除之後，symbolic link的檔案會開不了。

題目Ｅ：

一. 如果一個使用者可以進入(指令cd)某一目錄，該使用者對這個目錄具有X的權限。 如果使用者能夠在該目錄下執行ls指令，那麼該使用者具有r與x的權限。

二. 如果使用者在某個目錄能夠讀取(指令包括cat、more、less)一個檔案，那麼該使用者至少俱有rx權限。 如果使用者在此目錄內建立、刪除(指令包括touch、rm)一個檔案，那麼該使用者對這個目錄至少具有wx的權限。