

Git Lab 3

一、本節目地

- Git 進階指令、分支開設的使用及介紹

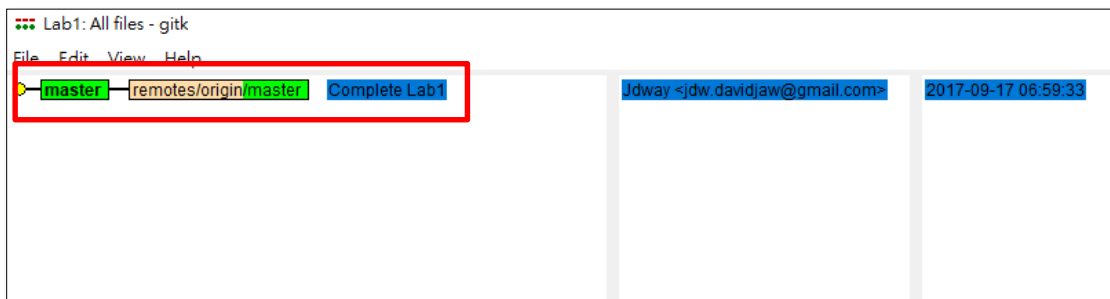
二、設計重點

- 本次的練習是模擬多人開發環境下會遇到的狀況以及處理，包括：
 - 開設 Branch
 - 切換 Branch
 - Merge Branch

三、設計步驟

1. 查詢專案狀態

承接 Git Lab 2，目前我們在遠端跟本地端都僅有一個分支 master，並且僅有一個 commit 點。而本地端的專案狀態可以使用 `gitk` 指令來查詢：



master 代表本地端的分支名稱，remotes/origin/master 代表遠端的分支。由於目前兩邊是同步的，因此顯示在同一點上。

2. 開設新分支

開設新分支的語法為：`git checkout -b “分支名稱”`

這邊我們開設一個新分支名為 `std_id`，目的是要印出你的學號

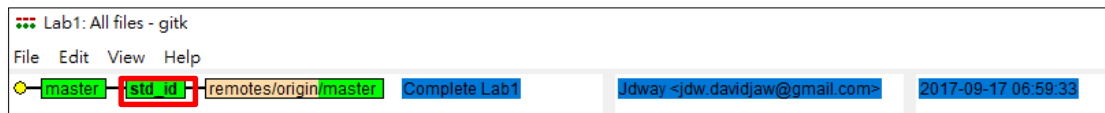
```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git checkout -b "std_id"
Switched to a new branch 'std_id'

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ |
```

注意紅框處，本來為 `master` branch，創建新分支後變為了 `std_id` 分支

3. 查詢專案狀態

此時我們使用 `gitk` 指令來看目前的專案狀態，會發現多了一個分支：

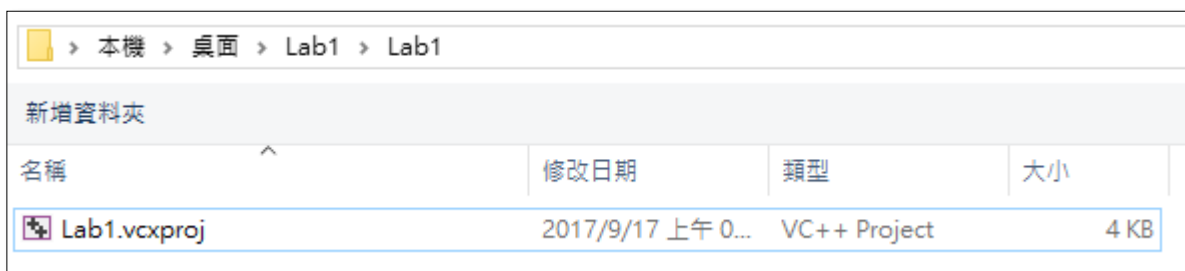


4. 修改 main.c 檔案

開啟 Visual Studio 2013，選擇開啟專案



到專案存放目錄，進入專案名稱的資料夾，選擇 `Lab1.vcxproj` 開啟



在 main.c 新增程式碼如下圖，將數字替換成自己學號

```
main.c [X]
(全域範圍)
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    printf("Welcome to C!\n");
    printf("My student id is: 106666666\n");
    system("pause");
    return 0;
}
```

執行程式如下圖：

```
Welcome to C!
My student id is: 106666666
請按任意鍵繼續 . . .
```

輸入 `git status` 查看檔案狀態：

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git status
On branch std_id
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab1.sdf
        modified:   Lab1.v12.suo
        modified:   source/main.c
        modified:   temp/debug/Lab1.exe
        modified:   temp/debug/Lab1.ilc
        modified:   temp/debug/Lab1.log
        modified:   temp/debug/Lab1.pdb
        modified:   temp/debug/Lab1.tlog/link.read.1.tlog
        modified:   temp/debug/main.obj
        modified:   temp/debug/vc120.idb
        modified:   temp/debug/vc120.pdb

no changes added to commit (use "git add" and/or "git commit -a")

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ |
```

這邊我們發現一個問題，僅修改一個檔案，卻有非常多的檔案被更動了，而那些更動並不是我們想要傳到伺服器上的，因此我們下一步僅會將 main.c 加入到佇列

5. 加入佇列並上傳

我們使用 `git add` 指令來僅加入 `main.c`：

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git add source/main.c
```

然後我們使用 `git status` 觀看狀態，可以發現僅有 `main.c` 被加入到提交佇列中

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git status
On branch std_id
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   source/main.c

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab1.sdf
        modified:   Lab1.v12.suo
        modified:   temp/debug/Lab1.exe
        modified:   temp/debug/Lab1.ilk
        modified:   temp/debug/Lab1.log
        modified:   temp/debug/Lab1.pdb
        modified:   temp/debug/Lab1.tlog/Lab1.lastbuildstate
        modified:   temp/debug/Lab1.tlog/link.read.1.tlog
        modified:   temp/debug/main.obj
        modified:   temp/debug/vc120.idb
        modified:   temp/debug/vc120.pdb

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ |
```

接著使用 `git commit` 以及 `git push` 指令來上傳到遠端資料庫

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git commit -m "Add student ID"
[std_id b889be1] Add student ID
1 file changed, 1 insertion(+)

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git push
fatal: The current branch std_id has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin std_id

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ |
```

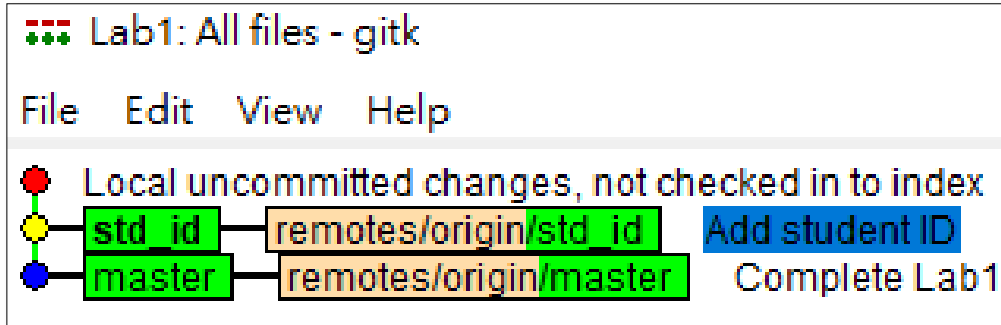
紅框訊息告訴我們遠端伺服器尚未有這個分支，因此我們輸入指令來在遠端伺服器建立相同分支並推送更新。

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git push --set-upstream origin std_id
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 369 bytes | 369.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/davidjaw/Ch0-Lab-1.git
 * [new branch]      std_id -> std_id
Branch std_id set up to track remote branch std_id from origin.

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ |
```

6. 確認目前專案狀態

我們使用 `gitk` 來觀察目前專案狀態



我們可以看到：

- 紅點是尚未提交的變更(專案自動建立的檔案 / 除錯用的檔案)
- 黃點是我們新建立的分支，領先 `master` branch 一個 commit 點
- 藍點是 `master` branch

7. 回到 `master` branch, 建立新分支

首先我們要先切換分支回 `master` branch：

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_id)
$ git checkout master
Switched to branch 'master'
M       Lab1.sdf
M       Lab1.v12.suo
M       temp/debug/Lab1.exe
M       temp/debug/Lab1.ilc
M       temp/debug/Lab1.log
M       temp/debug/Lab1.pdb
M       temp/debug/Lab1.tlog/Lab1.lastbuildstate
M       temp/debug/Lab1.tlog/link.read.1.tlog
M       temp/debug/main.obj
M       temp/debug/vc120.idb
M       temp/debug/vc120.pdb
Your branch is up-to-date with 'origin/master'.

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ |
```

目前所在的分支可以藉由藍色括號內的文字確認

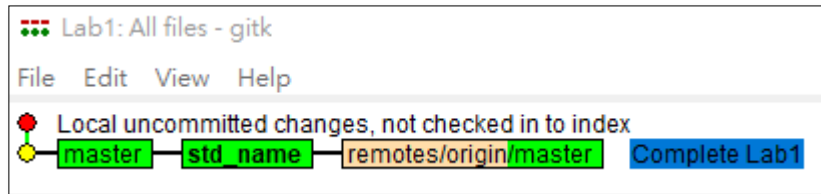
接著我們使用相同的指令來建立新分支 `std_name`：

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git checkout -b std_name
Switched to a new branch 'std_name'
M       Lab1.sdf
M       Lab1.v12.suo
M       temp/debug/Lab1.exe
M       temp/debug/Lab1.ilc
M       temp/debug/Lab1.log
M       temp/debug/Lab1.pdb
M       temp/debug/Lab1.tlog/Lab1.lastbuildstate
M       temp/debug/Lab1.tlog/link.read.1.tlog
M       temp/debug/main.obj
M       temp/debug/vc120.idb
M       temp/debug/vc120.pdb

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$ |
```

8. 確認目前專案狀態

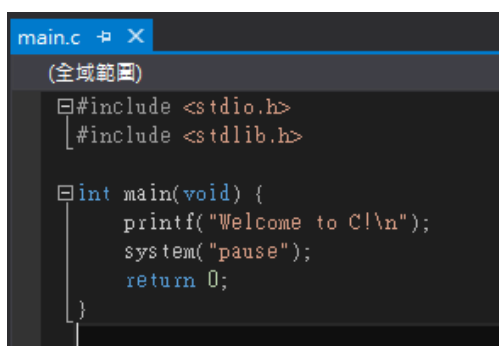
一樣，我們使用 `gitk` 指令來看目前專案狀態



我們會發現我們沒辦法看到 `std_id`，這是因為 `std_id` 跟 `std_name` 都是從 `master` 上長出來的互相獨立的分支

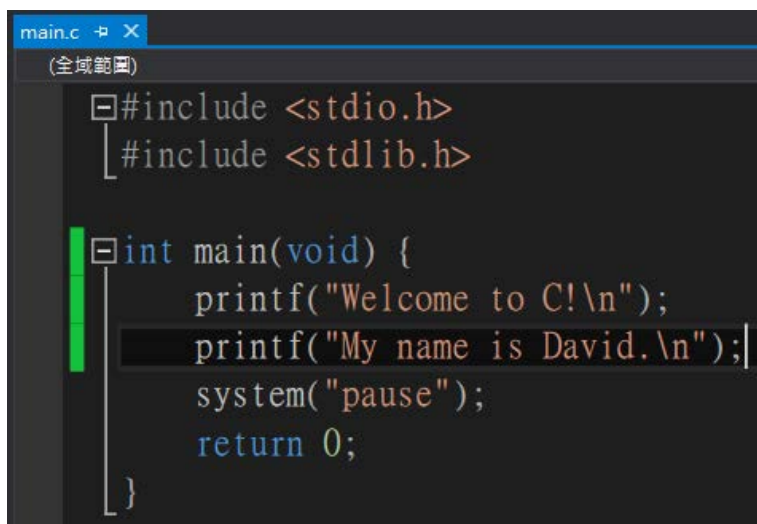
9. 開設新分支，修改並上傳

打開 Visual Studio 2013，打開 `main.c`：



我們會發現 `std_id` 所作的新增並沒有在這邊作用

我們輸入以下程式碼(將名字改為你自己的)：



依照之前所教的步驟進行 `add`, `commit`, `push`

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$ git add source/main.c

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$ git commit -m "Add student name"
[std_name dfbc60e] Add student name
1 file changed, 1 insertion(+)

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$ git push --set-upstream origin std_name
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 360 bytes | 360.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/davidjaw/Ch0-Lab-1.git
 * [new branch]      std_name -> std_name
Branch std_name set up to track remote branch std_name from origin.

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$
```

10. 切換回 master，確認目前專案狀態

我們使用 `git checkout` 來切換回 `master` branch

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (std_name)
$ git checkout master
Switched to branch 'master'
M       Lab1.sdf
M       Lab1.v12.suo
M       temp/debug/Lab1.exe
M       temp/debug/Lab1.ilc
M       temp/debug/Lab1.log
M       temp/debug/Lab1.pdb
M       temp/debug/Lab1.tlog/Lab1.lastbuildstate
M       temp/debug/Lab1.tlog/link.read.1.tlog
M       temp/debug/main.obj
M       temp/debug/vc120.idb
M       temp/debug/vc120.pdb
Your branch is up-to-date with 'origin/master'.

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ |
```

要注意紅框處已經換回 `master`

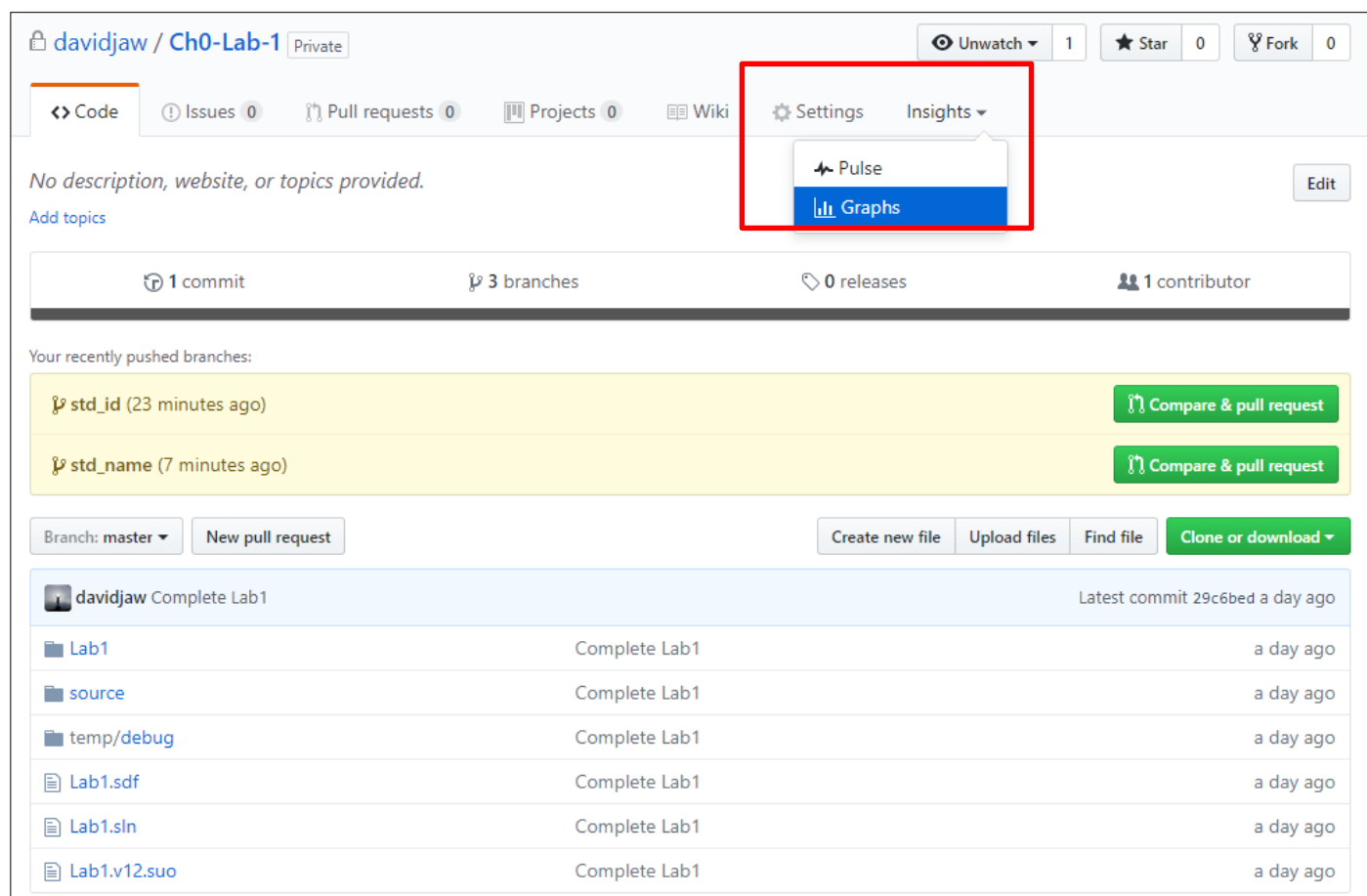
使用 `git branch` 來確認目前分支情況

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git branch
* master
  std_id
  std_name

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ |
```

11. 確認遠端專案狀態

打開專案網頁，選擇 insights Graphs 按鈕



Davidjaw / Ch0-Lab-1 Private

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Settings Insights

No description, website, or topics provided.

Add topics

1 commit 3 branches 0 releases 1 contributor

Your recently pushed branches:

- std_id (23 minutes ago) Compare & pull request
- std_name (7 minutes ago) Compare & pull request

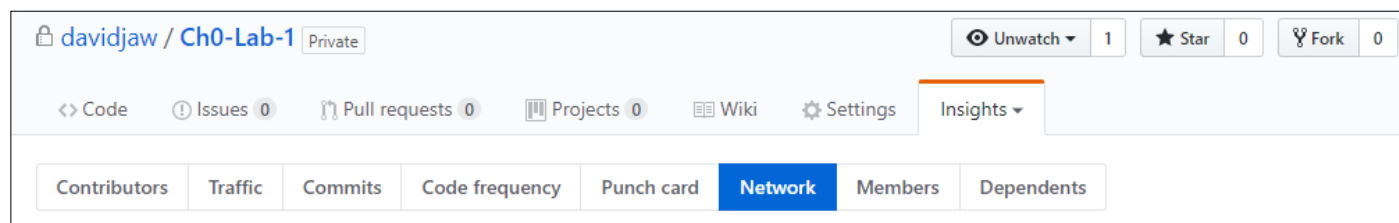
Branch: master New pull request

Create new file Upload files Find file Clone or download

Davidjaw Complete Lab1 Latest commit 29c6bed a day ago

Lab1	Complete Lab1	a day ago
source	Complete Lab1	a day ago
temp/debug	Complete Lab1	a day ago
Lab1.sdf	Complete Lab1	a day ago
Lab1.sln	Complete Lab1	a day ago
Lab1.v12.suo	Complete Lab1	a day ago

按下 Network



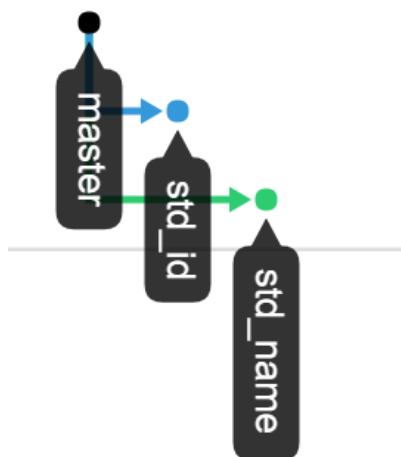
Davidjaw / Ch0-Lab-1 Private

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Settings Insights

Contributors Traffic Commits Code frequency Punch card Network Members Dependents

我們可以觀察到目前的分支情況：



12. 整合兩個 branch 的變動到 master 上

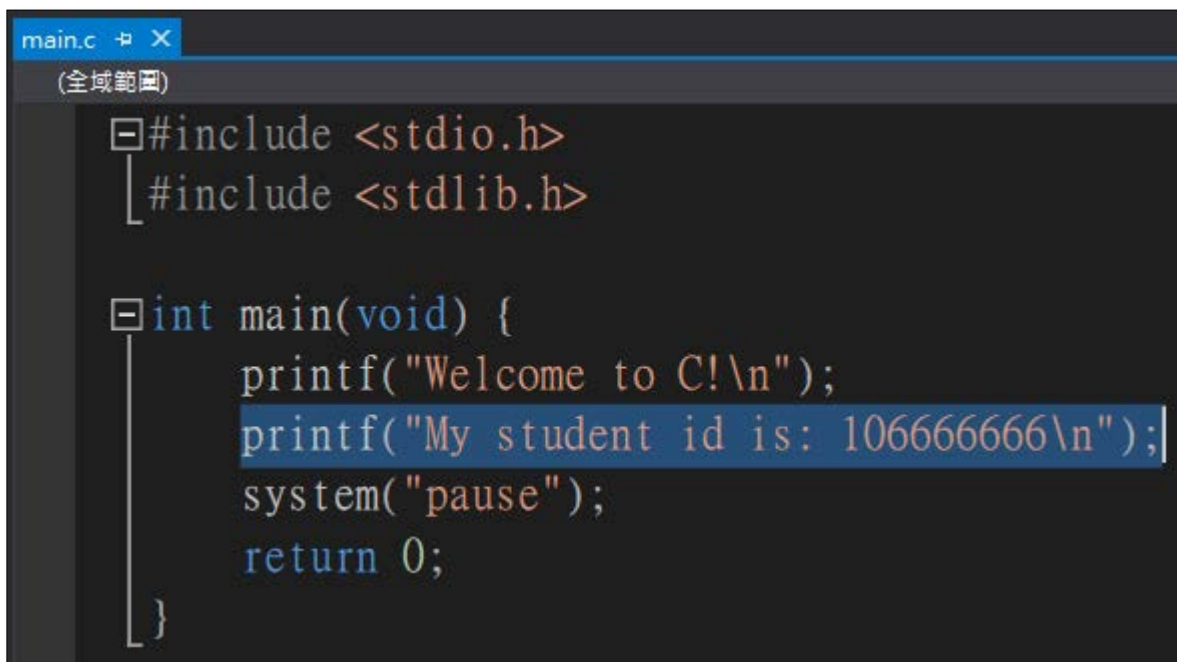
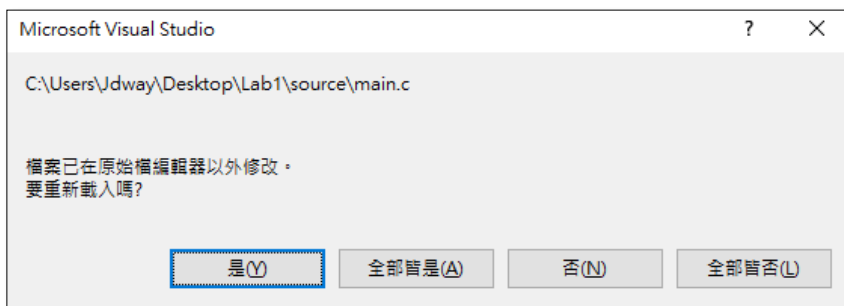
目前我們有 `std_id` 分支來輸出學號、`std_name` 來輸出姓名，我們現在要做的就是將兩個變動整合到主分支 `master` 上。

首先我們使用 `git merge` “分支名稱”指令來整合 `std_id`：

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git merge std_id
Updating 29c6bed..b889be1
Fast-forward
 source/main.c | 1 +
 1 file changed, 1 insertion(+)

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ |
```

我們打開 `main.c`，會發現 `std_id` 分支所加入的學號已經被整合到 `master` 上：



接著我們整合 std_name 分支：

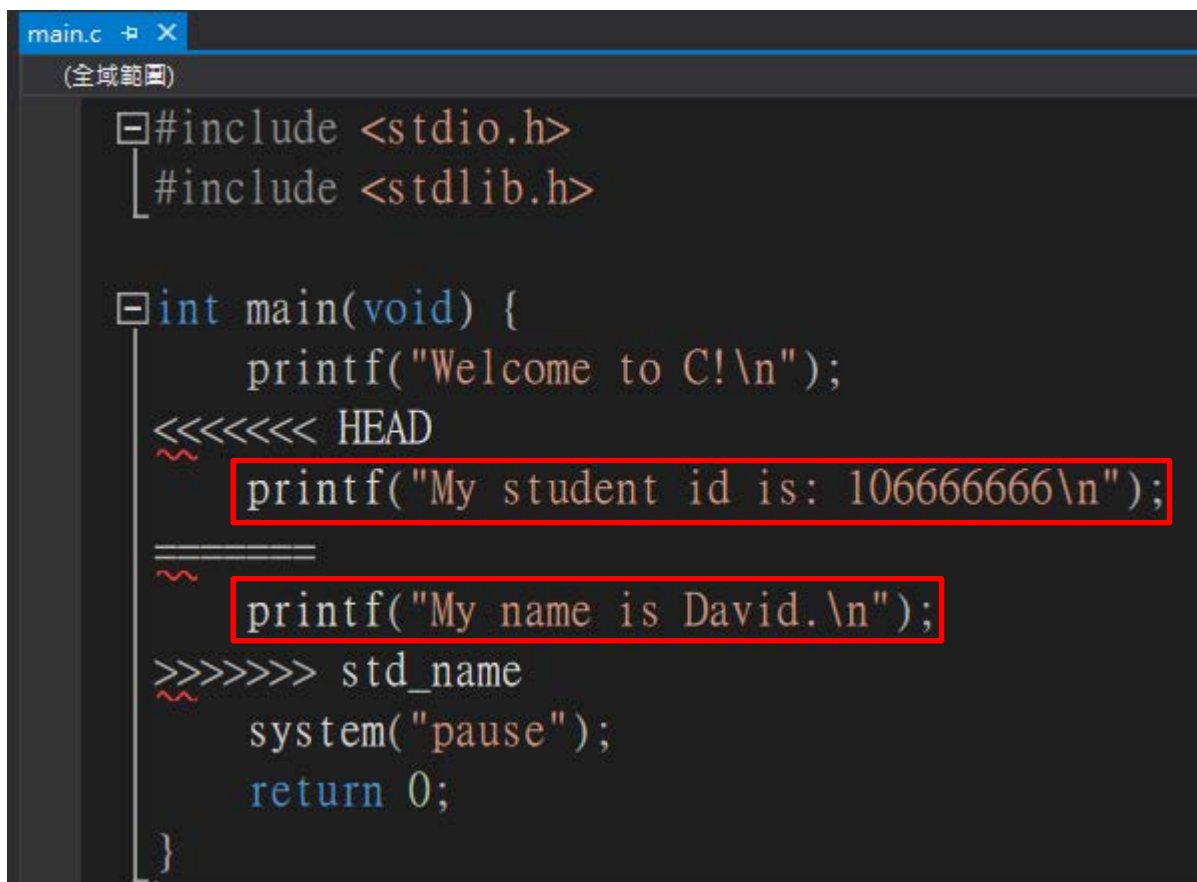
```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git merge std_name
Auto-merging source/main.c
CONFLICT (content): Merge conflict in source/main.c
Automatic merge failed; fix conflicts and then commit the result.

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master|MERGING)
$ |
```

會發現發生 conflict(衝突)情況

此情況的發生是因為兩個 branch 分別對同樣的檔案做了修改，並且修改的位置相同，執行 merge 時無法自動的解決這種情況，因此需要額外的人工修正。

我們打開 main.c，會發現內容變成：



```
main.c  +  X
(全域範圍)

#include <stdio.h>
#include <stdlib.h>

int main(void) {
    printf("Welcome to C!\n");
    <<<<<<< HEAD
    printf("My student id is: 106666666\n");
    =====
    printf("My name is David.\n");
    >>>>>>> std_name
    system("pause");
    return 0;
}
```

其中 HEAD 代表的是”當前分支的內容”，等號下面代表的是”欲整合分支的內容”，本次練習中，我們兩種資訊都要，因此我們只需要把提示訊息那幾行刪除即可

刪除後的檔案內容：

```
main.c # X
(全域範圍)
#include <stdio.h>
#include <stdlib.h>

int main(void) {
    printf("Welcome to C!\n");
    printf("My student id is: 1066666666\n");
    printf("My name is David.\n");
    system("pause");
    return 0;
}
```

最後，我們一樣使用 add, commit, push 將本地端的資料推送到遠端。

```
Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master|MERGING)
$ git add source/main.c

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master|MERGING)
$ git commit -m "Resolve Conflict"
[master c706706] Resolve Conflict

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ git push
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 389 bytes | 389.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/davidjaw/Ch0-Lab-1.git
  29c6bed..c706706  master -> master

Jdway@DESKTOP-GBM49C1 MINGW64 ~/Desktop/Lab1 (master)
$ |
```

四、 檢查重點

1. 程式執行結果

```
Welcome to C!  
My student id is: 106666666  
My name is David.  
請按任意鍵繼續 . . .
```

2. 網頁的分支圖

