趣味科學：程式設計體驗

台中一中參訪成大電機活動

2015/08/25

歡迎來到成大電機系！

WELCOME TO NCKUEE!

調査時間

今天的主角：程式

- 原意是指「指特定的一系列動作、行動或操作」
- 套用到電腦上，則是指「一組指示電腦或其他具有訊息處理能力裝置每一步動作的指令，而這些指令，通常由程式語言撰寫」
- 程式也是與電腦溝通的方法

```
struct group_info init_groups = { .usage = ATOMIC_INIT(2) };
struct group_info *groups_alloc(int gidsetsize){
    struct group_info *group_info;
    int nblocks;
    int i;

    nblocks = (gidsetsize + NGROUPS_PER_BLOCK - 1) / NGROUPS_PER_BLOCK;
    /* Make sure we always allocate at least one indirect block pointer */
    nblocks = nblocks ? : 1;
    group_info = kmalloc(sizeof(*group_info) + nblocks*sizeof(gid_t *), GFP_USER);
    if (!group_info)
        return NULL;
    group_info->ngroups = gidsetsize;
    group_info->nblocks = nblocks;
    atomic_set(&group_info->usage, 1);

    if (gidsetsize <= NGROUPS_SMALL)
        group_info->blocks[0] = group_info->small_block;
    else {
        for (i = 0; i < nblocks; i++) {
            gid_t *b;
            b = (void *)__get_free_page(GFP_USER);
            if (!b)
                goto out_undo_partial_alloc;
            group_info->blocks[i] = b;
        }
    }
    return group_info;
```

與電腦溝通

- 其實並不只有靠程式語言才能與電腦溝通，我們日常做的各種操作，包括聽音樂、玩線上遊戲、打報告等等，其實都是在與電腦溝通

- 程式語言只是其中一種方式

與電腦溝通

- 我們也能讓電腦幫我們處理大量的資料！
- Facebook 戀愛報告書

- 這也是助教的實驗室在做的事情～

Why Programming Languages?

- 那為什麼我們還需要程式語言呢？

- 程式語言能提供對電腦更明確、精確的控制
- 易於辨識及儲存
- 具結構性與意義
- …

程式語言

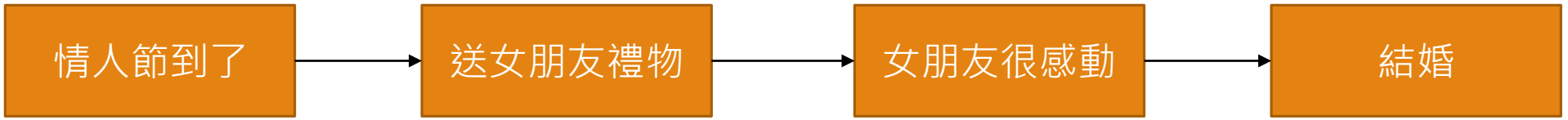
- 程式語言對於電腦，就像是劇本一樣的存在

- 編劇寫完劇本，給演員演
- 程式設計師寫完程式，給電腦跑

- 但劇本只有一條路，一路演到底
- 程式必須應付各種狀況，做出相對應的處理與動作

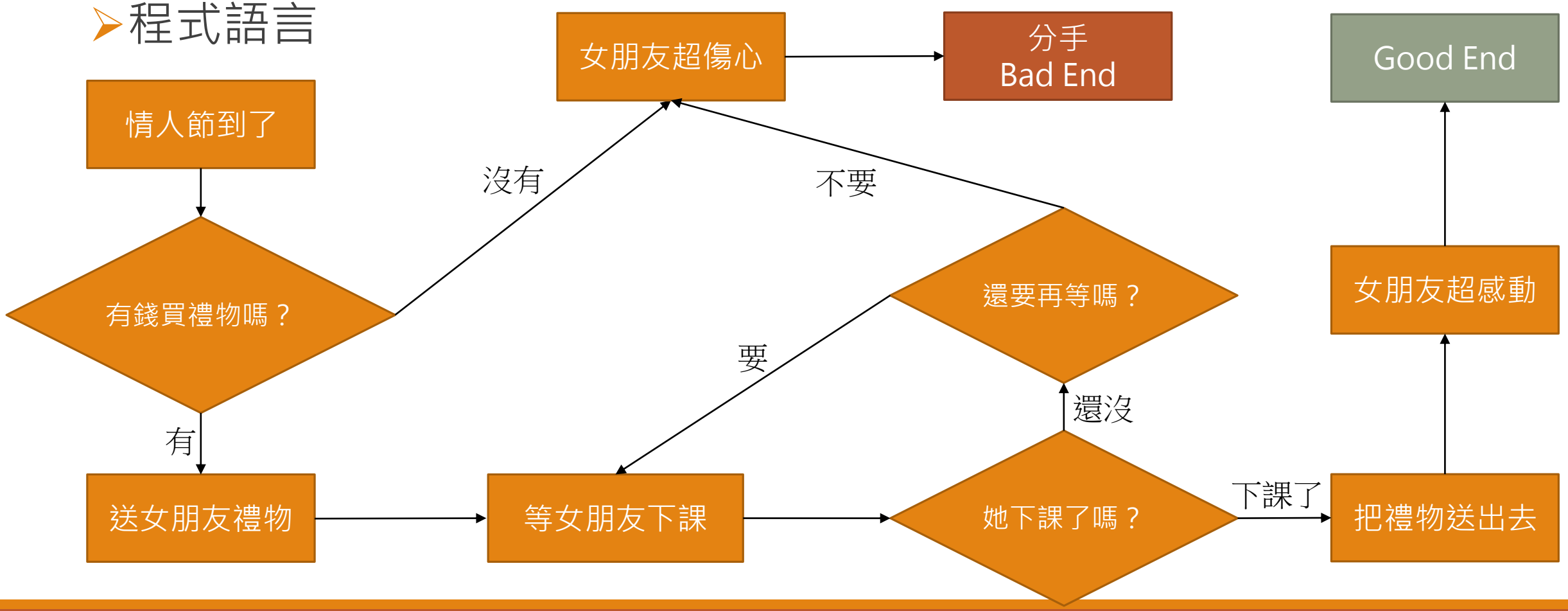
劇本 vs. 程式語言

- 劇本



劇本 vs. 程式語言

- 程式語言



程式語言的種類

- 對於各式各樣的需求，許多種類的程式語言被發展出來

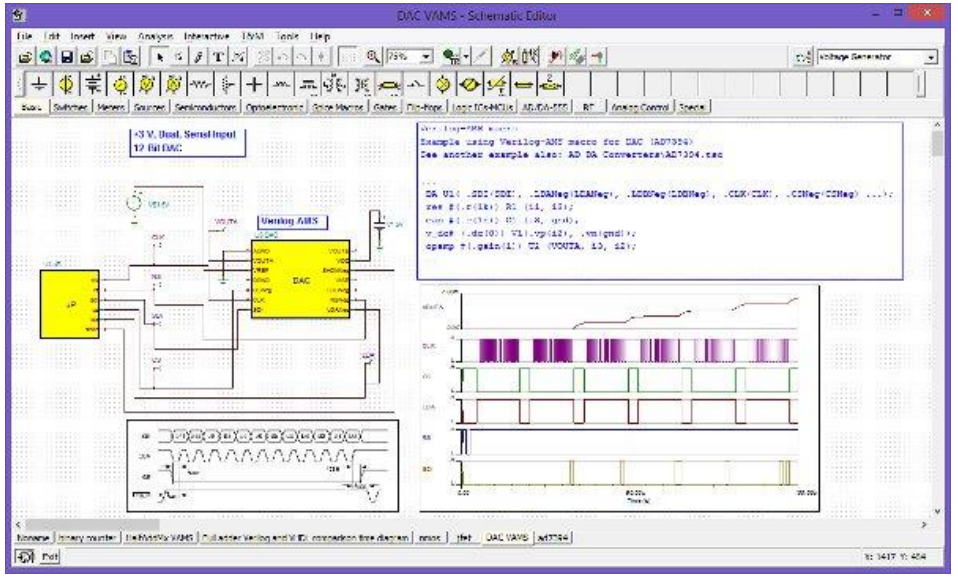
- JavaScript: 互動式網頁
- SQL: 操作資料庫
- MatLab: 科學計算
- C++: 桌面程式等
- …

程式設計與電機系

- 電機系大學部的課程中，將會牽涉許多各式各樣的模擬系統
- 透過程式描述、模擬各種硬體與物理條件，可以在不影響現實的條件下，得知設計的缺陷，從而改進之
- 降低實際失敗的損失與達到教學上的目的

程式設計與電機系

- **IC設計：硬體描述語言**
- 描述電子電路（特別是數位電路）功能、行為的語言，可以被用來進行電路系統設計，並能通過邏輯仿真的形式驗證電路功能



- Verilog
- VHDL

程式設計與電機系

- **嵌入式系統**
- 它們的特性是可以做非常低階(接近硬體等級)的操作，指令很細，用來細部控制硬體等等
- C
- 組合語言

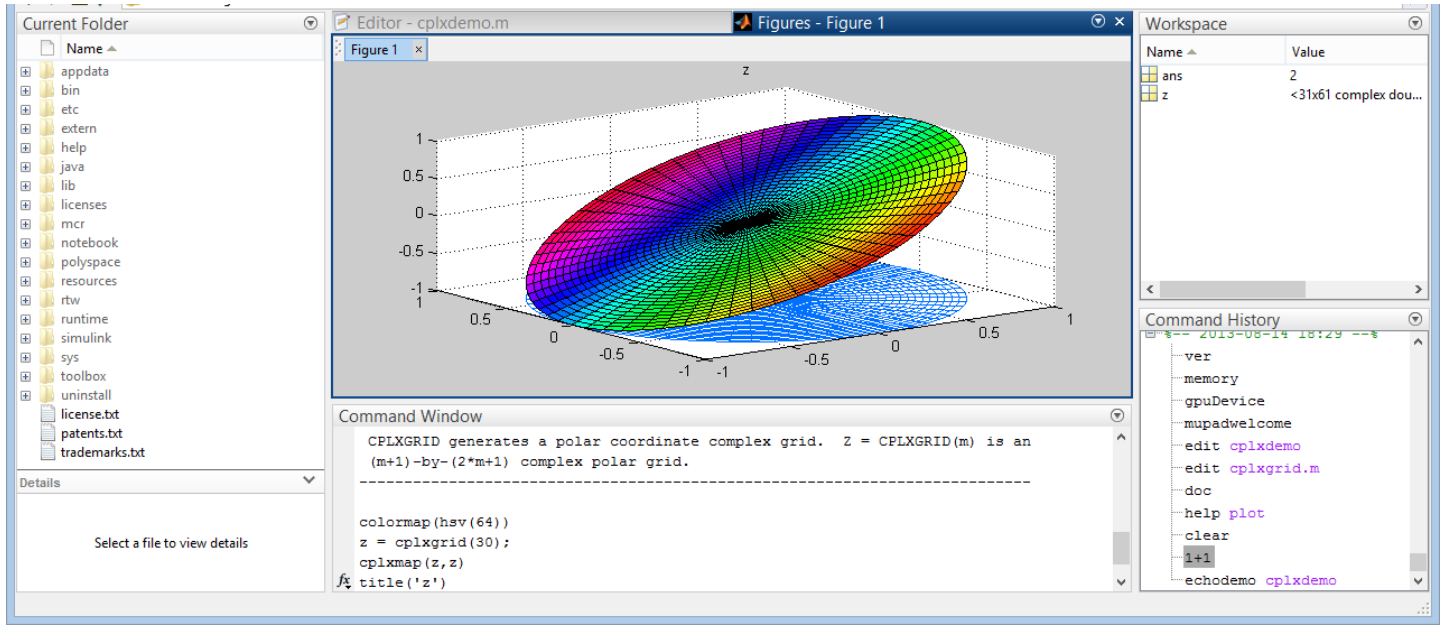
程式設計與電機系

➢科學計算

➢它們被設計成易於快速、有效率地進行科學計算，而且通常會附帶許多製圖、處理大量資料的工具

➢MATLAB

➢R



程式設計與電機系

- **演算法驗證**
- 對電腦與網路組，它們專精的或許不一定是硬體等實際看到的東西，但他們所發展的演算法，卻能影響電腦間的資料怎麼傳送才會有效率、或是Facebook會推薦哪些朋友給你等等

- Java
- C++
- Python

程式語言的幾個共通特性

- 雖然程式語言千百種，但它們都有一些共同的特性：
  - 變數
  - 輸出入
  - 運算子
  - 流程控制
  - 迴圈

- 這也是今天的重頭戲
- 我們將以 C++ 語言為例子，帶領大家認識這些特性

變數

- 一個用來表示值的符號
- 有點像數學方程式中的未知數

- 假設小明有 x 元，這個 x 在程式語言中就是一個「變數」
- 而這個變數代表的意義是「小明的金錢數量」

變數

- 是程式語言的根本
- 其實程式說穿了就是透過許多指令，在不同的情況下對變數做各種操作

- 小明身上有 50 元，x = 50
- 小明撿到 100 元，身上變成有 150 元
  - x = 150 (50 + 100 = 150)

變數的型態

- 由於在電腦中，所有的資料都是以 1 與 0 儲存
- 要賦予一串 1 與 0 有意義，便需要知道這個變數的型態

- 「0101001111011101」在不同型態下會有不同的意思
- 它可能代表整數、或是小數、甚至是一串文字

變數 in C++

- 宣告變數
  - 意義：告訴作業系統要分配多少記憶體
  - 語法：
    - int a; // 宣告一個型態是「整數」的變數，名稱為「a」
    - int b = 20; // 宣告一個型態是「整數」的變數，名稱為「b」，並賦予值為 20
  - 型態 變數名稱 (= 初始值)

變數型態 in C++

- 整數：int
- 浮點數：float 或 double
- 字元：char
- 真假值：bool
  - 其值只有兩種可能：true(真) 與 false(假)

輸出入

- 要有東西進來，電腦才能處理
- 同理，要有輸出我們才知處理的結果

- In C++:
- 輸出一段文字與變數: cout << “I have” << b << “dollars.”;
- 從鍵盤輸入到變數： cin >> xyz;
- 提醒：C++的每一個描述後面都要加分號

算術運算子 in C++

- 簡言之就是數學符號，但一般程式語言中定義又不太一樣
- +-*/：加減乘除
- =：賦值(把等號右邊的東西丟到左邊去)
  - x = 3　將 x 設為 3
  - x = x + 3 將 x 的值加 3
- %：mod (除後取餘數)
  - 範例：　5 % 2 = 取(5/2)的餘數 = 1

算術運算子 in C++

- 範例

- a = 6 + 7; // a = 13
- b = a + 8; // b = 21
- c = b – a; // c = 21 – 13 = 8
- c = c * 2;
- cout << c; // 輸出 16

邏輯運算子 in C++

- 大於：>
- 小於：<
- 大於等於：>=
- 小於等於：<=
- ==：這才是數學上的相等
- !=：不相等

邏輯運算子 in C++

- 生活中常有且與或
  - 你是男生 且 你是女生 與 你是男生 或 你是女生 差很多！

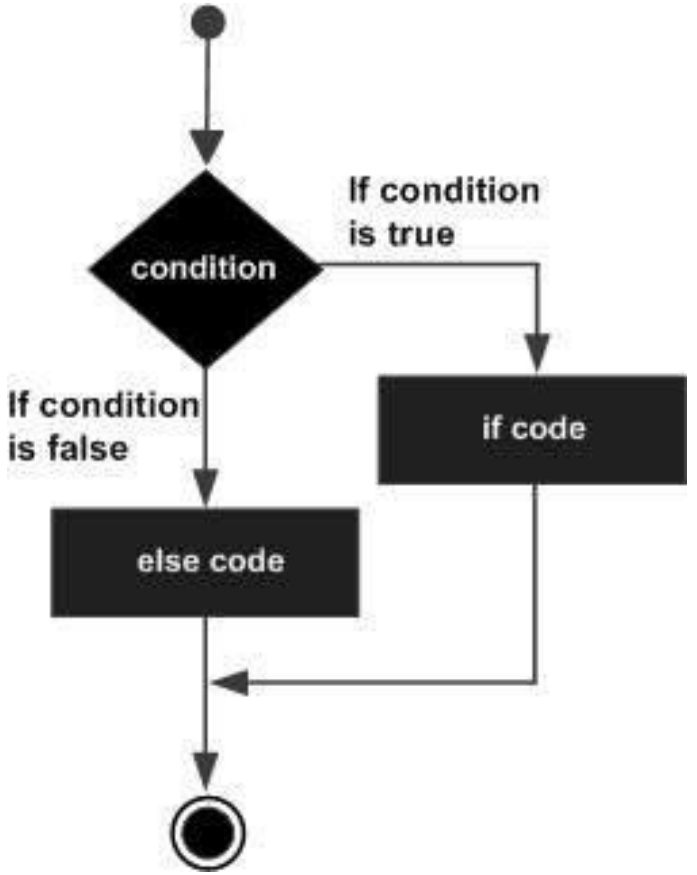
- 在 C++ 中，條件與條件之間可以用邏輯運算子串連
- 且： (條件) && (條件)
- 或： (條件) || (條件)

- E.g. (age < 20) || (age > 50)

流程控制

- 猶如最前面的範例，人生總會面臨抉擇...

```
if(有錢) {
        送禮物;
  }
  else {
        沒得送;
  }
```



流程控制

```
if(很有錢) {
        送戒指;
}
else if(有一點錢)  {
        送鮮花;
}
else {
        沒得送;
}
```

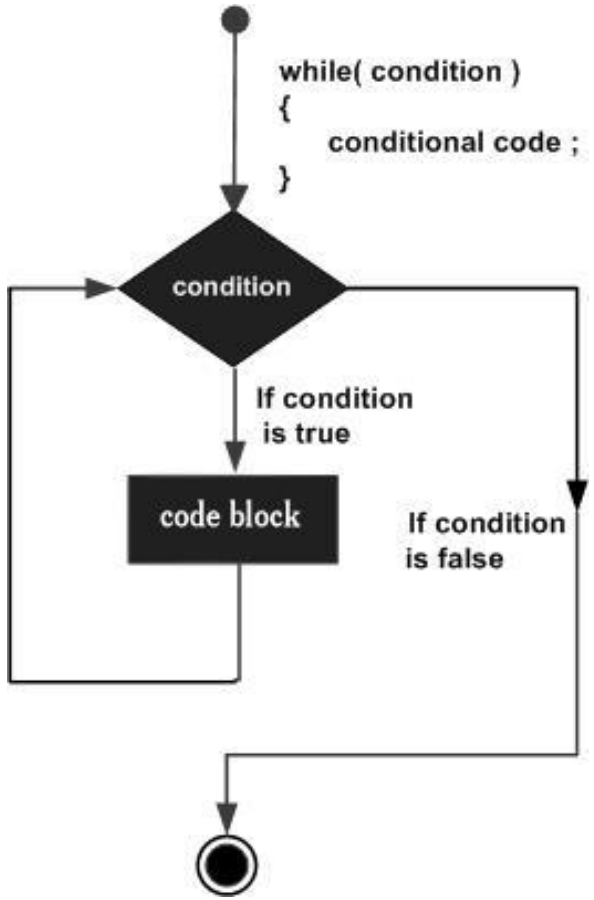
流程控制

```
if(age < 35) {
    cout << “年輕人”;
}
else if((age >= 35) && (age < 60)) {
    cout << “中年人”;
}
else {
    cout << “老人”;
}
```

迴圈

- while(同學還沒下課) {
  繼續等;
  }

- while(同學下課了 && 被留下來) {
  繼續等;
  }



迴圈中的流程控制

```
while(some condition) {
    // do something...
    break; // 直接跳出迴圈
    // 這裡不會被執行到
}

// 接著執行這裡
```

迴圈中的流程控制

```
while(some condition) {
    // 會不斷的執行這裡
    continue; // 直接跳到迴圈開頭
    // 這裡也不會被執行到
}
```

註解

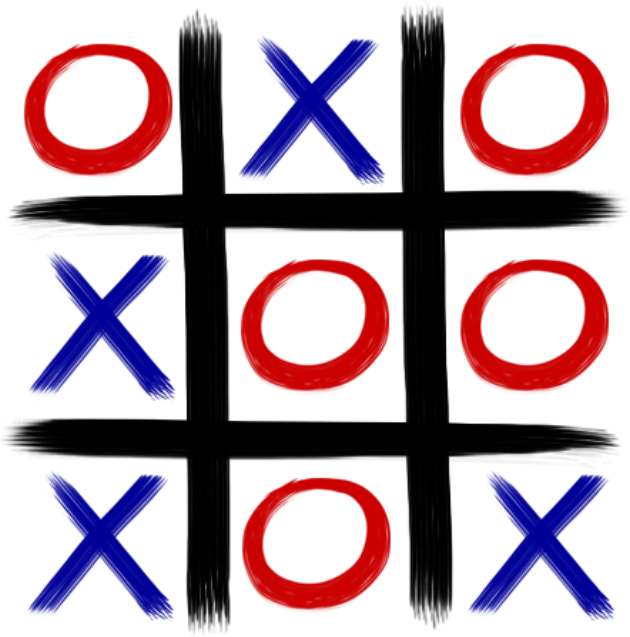
- 雖然我們常說「好的工程師寫的出人都能輕易看懂的程式碼」，但也不總是這樣，所以在程式碼間，我們常常需要許多「人看得懂的文字」來輔助我們理解這段程式碼的意義
- 在 C++ 中，註解有分成單行與多行
- int a = 25; *// 我是單行註解*
- */* 第一行註解
- 第 n 行註解 */

練習時間

圈圈叉叉

- 今天要讓大家來練習寫「圈圈叉叉」
- 玩法大家應該都很熟悉，不用說明了吧XD

- 請大家下載課程網頁上的範例程式碼「專案」
- 解壓縮後使用 Visual Studio 開啟



圈圈叉叉：程式結構

- 我們用 grid 陣列(變數)來表示整個棋盤
- 記住 grid 是從 0 開始！

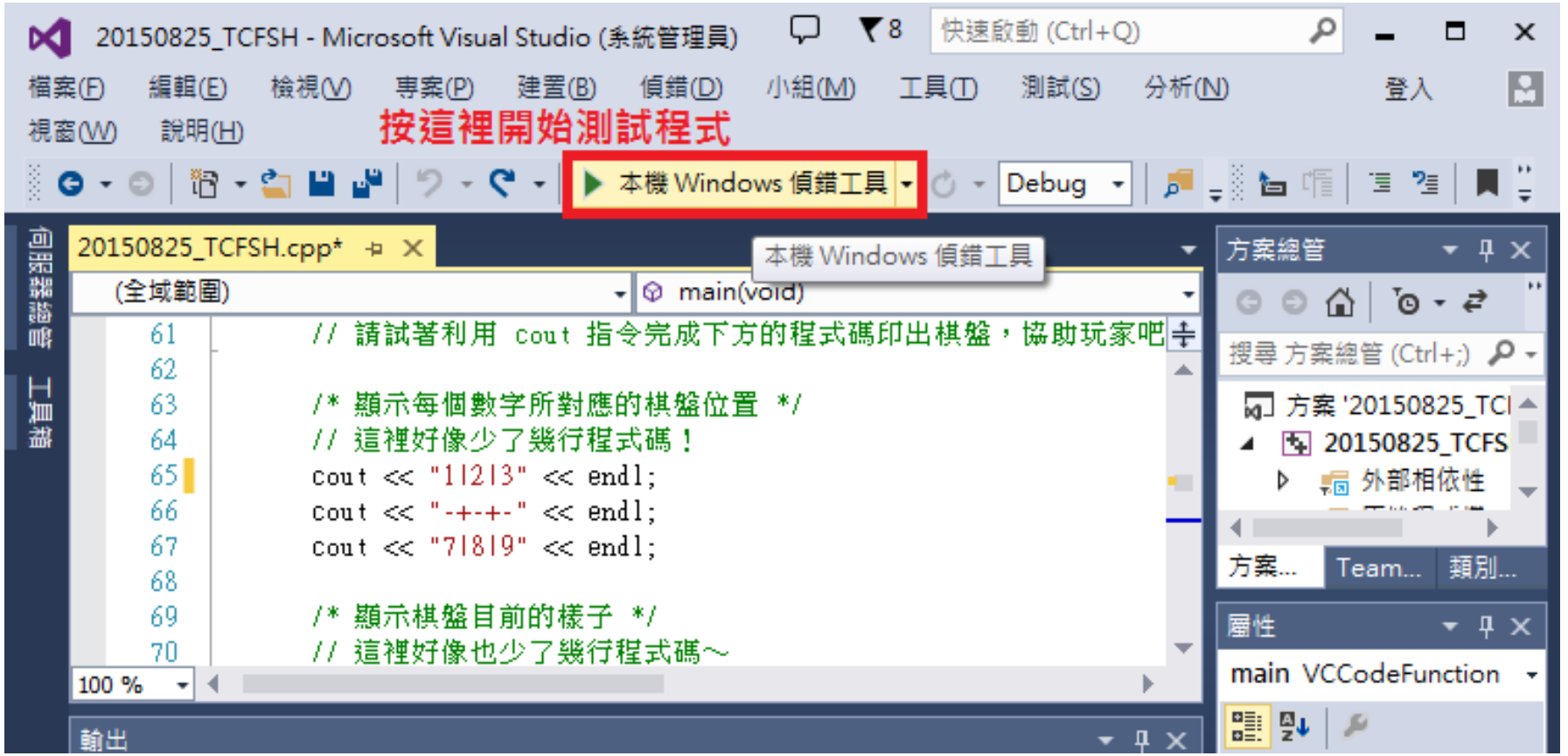
grid[0]	grid[1]	grid[2]
grid[3]	grid[4]	grid[5]
grid[6]	grid[7]	grid[8]

第1格	第2格	第3格
第4格	第5格	第6格
第7格	第8格	第9格

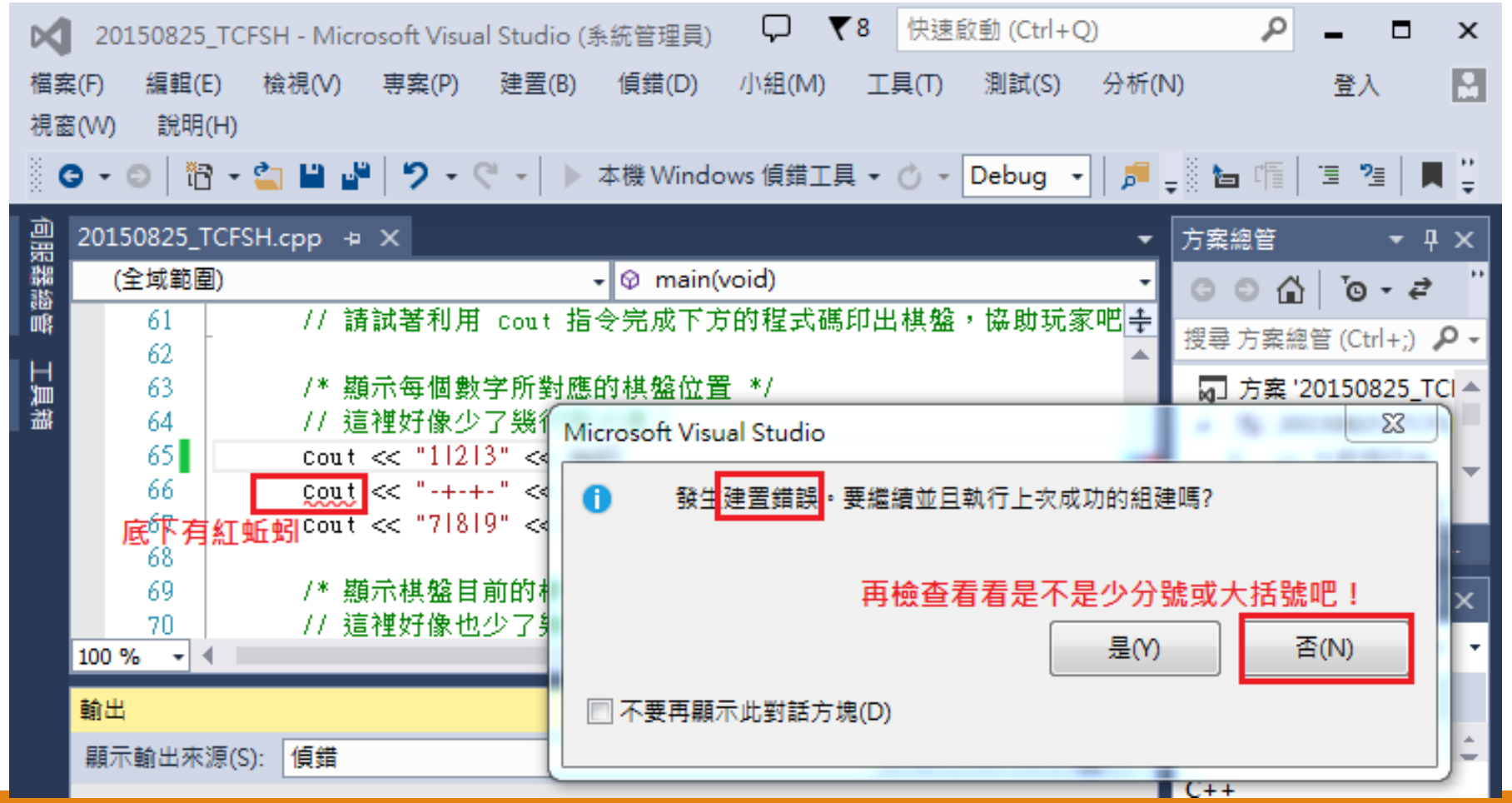
圈圈叉叉：流程

1. 根據回合數判斷是誰下
2. 下的對不對
3. 有人贏了嗎？或是有和局嗎？
   1) 有：跳到 4.
   2) 無：下一回合，跳到 1. 重新開始
4. 輸出誰贏或是和局

圈圈叉叉：測試程式


20150825_TCFSH - Microsoft Visual Studio (系統管理員)
快速啟動 (Ctrl+Q)
檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 偵錯(D) 小組(M) 工具(T) 測試(S) 分析(N) 登入
視窗(W) 說明(H)
按這裡開始測試程式
本機 Windows 偵錯工具
Debug
20150825_TCFSH.cpp*
本機 Windows 偵錯工具
(全域範圍)
main(void)
// 請試著利用 cout 指令完成下方的程式碼印出棋盤，協助玩家吧
/* 顯示每個數字所對應的棋盤位置 */
// 這裡好像少了幾行程式碼！
cout << "1|2|3" << endl;
cout << "-+-+-" << endl;
cout << "7|8|9" << endl;
/* 顯示棋盤目前的樣子 */
// 這裡好像也少了幾行程式碼～
100 %
輸出
方案總管
搜尋 方案總管 (Ctrl+;)
方案 '20150825_TCI
20150825_TCFS
外部相依性
方案...
Team...
類別...
屬性
main VCCodeFunction

圈圈叉叉：程式碼出錯了！


20150825_TCFSH - Microsoft Visual Studio (系統管理員)
快速啟動 (Ctrl+Q)
檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 偵錯(D) 小組(M) 工具(T) 測試(S) 分析(N) 登入
視窗(W) 說明(H)
本機 Windows 偵錯工具
Debug
20150825_TCFSH.cpp
(全域範圍)
main(void)
61 // 請試著利用 cout 指令完成下方的程式碼印出棋盤，協助玩家吧
62
63 /* 顯示每個數字所對應的棋盤位置 */
64 // 這裡好像少了幾
65 Cout << "1|2|3" <<
66 Cout << "-+-+-" <<
67 Cout << "7|8|9" <<
68
69 /* 顯示棋盤目前的
70 // 這裡好像也少了
底下有紅蚯蚓
100 %
輸出
顯示輸出來源(S): 偵錯
方案總管
搜尋 方案總管 (Ctrl+;)
方案 '20150825_TCI
Microsoft Visual Studio
發生建置錯誤。要繼續並且執行上次成功的組建嗎?
再檢查看看是不是少分號或大括號吧！
是(Y)
否(N)
不要再顯示此對話方塊(D)
C++

圈圈叉叉：Cheat Sheet

- 印出第 2 格的資訊： cout << grid[1];
- 印出第 2 格的文字：cout << symbol[grid[1]];
- 從鍵盤輸入東西到變數 x： cin >> x;
- 如果第 3 格是 O，則做...
  - if(grid[2] == 1) { ... }
- 如果第 3 格是 O，而且第 5 格是 X，則做...
  - if((grid[2] == 1) && (grid[4] == 2)) { ... }
- 如果使用者下第 4 格或第 7 格，則做...
  - if((player_action == 3) || (player_action == 6)) { ... }

線上資源

- 如果大家還意猶未盡，其實線上也有許多學習程式設計的管道
- 這些線上開放式課程網站(MOOCs)有許多實用的學習資源！

- Codecademy
- Code.org
- Coursera
- …

謝謝大家的聆聽！

祝大家有個愉快又充實的一天！