Topic 0

Class Introduction

資料結構與程式設計

Data Structure and Programming

09/12/2018

Class Information

- Class Website
  - https://ceiba.ntu.edu.tw/1071_DSnP
- Discussion board
  - FB → NTU_DSnP
  - Please go to FB/NTU_DSnP to apply, and make sure we can identify your displayed name as this board is open only to registered students.
  - ~~In case you don't get admitted in a few days, please go to https://goo.gl/Ua8k82 to leave your name~~
- My office:
  - EE building II - 444
  - (FB/Line/Skype/WeChat ID) ric2k1
  - (e-mail) cyhuang@ntu.edu.tw
  - Office hour: stop by or by PM/e-mail appointm
- Class TA(s)
  - FB 陳家暄
  - Others TBD

Class Information

- Required textbook: none
- Suggested reading
  - Class slides and source codes
    - Download from the Ceiba website
  - Any of your Data Structure and C++ programming textbooks
- Highly recommended (DO THEM ASAP)
  - Review C++
  - Get access to and be familiar with Linux-compatible working environment

Grading (May subject to change)

Homework	70%
Final project	30%
Bonus	TBD

The final grades are subject to linear adjustment. Instructor will determine the average and standard deviation

選課方式

- 本課程為二類加選(不開放初選)，想要領加簽單者，請至 https://goo.gl/D8yrDG sign up, 並且在 9pm, 09/18 (二) 前完成作業一上傳，批改通過之後，我們會將授權號碼用 e-mail 寄給你
  - 不用滿分，但也不能零分
- 請詳閱作業說明
  - 作業說明-- 很 長 --是本課程的特色，請提早習慣
- 請注意作業相關規定
  - 沒有按照規定命名檔案以及上傳者，會被扣分
- 我們有強大的抓抄襲程式，會在「事後檢查」是否有抄襲的現象，請勿以身試法，會有嚴重的後果。

Overview of this course

Part 1: Introduction

Part 2: Polishing Your Programming Skills

Part 3: Data Structure Revisited

Part 4: Putting What You Learn Together

(Last Year) 106-1 Class Schedule

09/14	Intro, C++ Review (Basic)		
09/21	C++ Review (Basic)	HW2 out	HW1 due
09/28	C++ Review(More on func, vars, classes)		
10/05	C++ Review(overloading, polymorphism)	HW3 out	HW2 due
10/12	C++ Review(overloading, polymorphism)		
10/19	Memory Mgr & Exception Handling	HW4 out	HW3 due
10/26	Complexity, List & Array		
11/04	Tree (Part I)	HW5 out	HW4 due
11/09	C++ Review - More on IO Streams		

(Last Year) 106-1 Class Schedule

11/16	Graph and Circuit	HW6 out	HW5 due
11/23	Special Topic: Lex and Yacc		
11/30	Linux Prog, Heap/Set/Map		
12/07	Cache and Hash	HW7 out	HW6 due
12/14	Final Project Discussion	Proj. out	
12/21	Final Project Discussion		HW7 due
12/28	Tree (Part II)		
01/06	~~Special Topic: C++11~~		
01/13	Final exam week		
01/20	Final project week		Proj. due

What would be different this year?

- Kind of 翻轉ing
  - 寫程式， or in general 學習 CS 相關知識， 很多觀念其實有些抽象，所以光用聽的，很容易忘記/沒感覺，一定要配合實際動手做、體驗、觀察，才能深刻體悟， 內化成自己的實力。
  - 每堂課最後，都會出幾個小練習當作下一堂課的教材，請大家回家配合下一堂課的投影片自行練習。
  - 有練習， 下次上課有體悟。沒練習，下次上課趕進度。
- 練習要不要交？算不算分數？
  - 我們還是會開 Ceiba 讓大家繳交，但不會算分也不會批改。
  - 但它可以在期末作為「bargaining power」，也就是說如果你的分數不小心差 0.5 分而掉了一個等地，你可以用它來證明你的認真程度， 可以當作唯一期末要分的理由。

107-1 Class Schedule

09/12	Intro, C++ Review (Basic)		
09/19	C++ Review (Basic)	HW2 out	HW1 due
09/26	C++ Review(More on func, vars, classes)		
10/03	C++ Review(overloading, polymorphism)	HW3 out	HW2 due
10/10	國慶日放假		
10/17	C++ Review(overloading, polymorphism)	HW4 out	HW3 due
10/24	Memory Mgr & Exception Handling		
10/31	Complexity, List & Array	HW5 out	HW4 due
11/07	Tree (Part I)		

107-1 Class Schedule

11/14	C++ Review - More on IO Streams		
11/21	Graph and Circuit	HW6 out	HW5 due
11/28	Special Topic: Lex and Yacc		
12/05	Linux Prog, Heap/Set/Map	HW7 out	HW6 due
12/12	Cache and Hash	Proj. out	
12/19	Final Project Discussion		HW7 due
12/26	Final Project Discussion		
01/02	Tree (Part II)		
01/09	Final exam week		
01/16	Final project week		Proj. due

Other administrative information

- For 旁聽生
  - 原則上不限旁聽，但如果教室過於擁擠， 請旁聽生將座位優先讓給修課的同學， 至隔壁博理 112 看轉播，謝謝合作！
  - For 台大學生，請寄給我你的中文姓名、系級、學號
  - For 非台大學生，請寄給我你的中文姓名、校名系級 or 職位，e-mail address
    - 寄到 cyhuang@ntu.edu.tw，我再幫你加為 Ceiba 的旁聽生。
  - 旁聽生沒有 Ceiba 作業區功能。想要拿到課前練習或者是作業，請自行想辦法， 不要來找我或是助教。

What HW#1 tells you…

- C++ 很煩!! 為什麼不直接學簡單又漂亮的 Python, 或者是很潮的 JavaScript 呢？
  - 身為工程學系的學生，我們除了要能夠很快的把事情做好之外，還要有把東西 optimize 10X 以上的能力
  - 不過現實是，看看你們 HW#1 的 code, 可以想像如果讓它再長大十倍、百倍、千倍… 會變成什麼樣子嗎？

關於【資料結構與程式設計】，這門課想要傳遞的觀念是…

- 語言的嚴謹性與設計的美感
  - 電腦語言 vs. 人類語言
- 程式的架構需要設計
  - 你寫的程式除了要讓電腦看得懂之外，讓人類(尤其是自己)看得懂更是重要
- 資料結構的重要性
  - 試想你有一堆等待被運算或是分析的資料，如何確定某筆資料存在？如何確認所有資料被運算過一次？如何有效率的增加或是刪除資料？
- 資料 vs. 物件, 結構 vs. 類別
  - 希望大家可以將「資料結構」與「程式設計」融會貫通

這門課的目標是：

除了對於資料結構能有

正確的觀念之外,

起碼要有自行 handle

1000 行程式碼的信心!

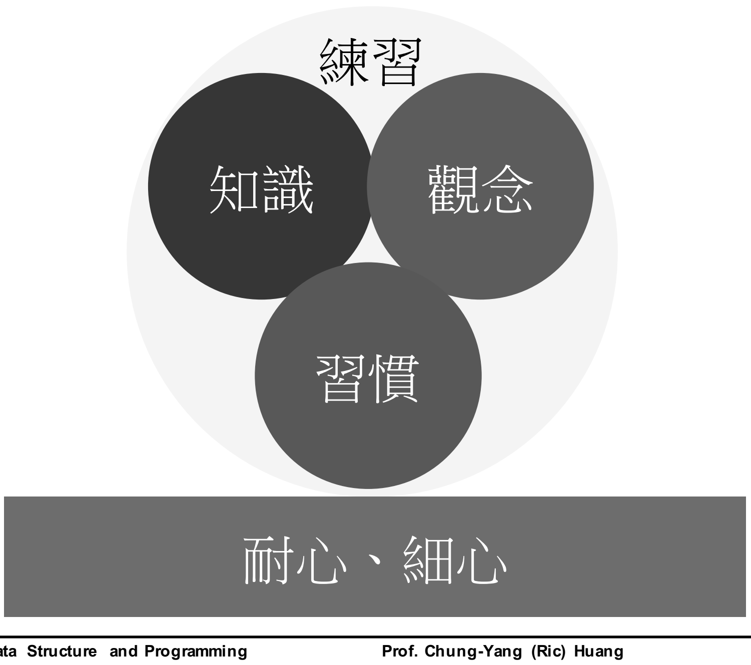
寫 1000 行程式，很難嗎？

- 當然，要看是寫什麼
- 如果是有功能性，可以解決一些問題的程式，1000 行的程式的確已經有相當的複雜度

  重點是 --- structured design and thinking!
- 人腦的思考複雜度有一定的限制，
  如果有超過一定數量的元素要一起考量，
  就會無法掌握
- 但階層式的、歸納式、模組化的思考，
  有助於化繁為簡，讓程式在可以 handle 的
  範圍內被最佳化

寫 1000 行程式，很難嗎？

- 10+ 行的程式
  → 課本的作業，練習語法，no brainer
- 100+ 行的程式
  → 熟悉語法之後，並持著一股浩然正氣的意念用力寫下去，大家都做得到
- 1000+ 行的程式
  → 如果有能力將 100+ 行的程式模組化，那 1000+ 行的程式要 handle 的只是最上層的 control flow, 何難之有？
- 10000+ 行的程式
  → s/1000/10000, s/10000/100000, repeat this!



Overview of this course

Part 1: Introduction

Part 2: Polishing Your Programming Skills

Part 3: Data Structure Revisited

Part 4: Putting What You Learn Together

-- Note --

Lecture notes and homework assignments are subject to change!

1. C++ Review - The Basic (Variables, Classes, IO Streams)

- Part I: Understanding "Variables"
  - What is a variable?
  - The concept of "memory"
  - Object, pointer, reference
- Part II: Understanding "Classes"
  - What is a "class"?
  - Constructor, destructor
  - new, new [], delete, delete []
  - A*, A**, A***....
  - Access privilege: private/protected/public
  - Friend

1. C++ Review - The Basic (Variables, Classes, IO Streams)

- Part III: Understanding "I/O Streams"
  - C++ standard I/O
    - Introduction
    - Class hierarchy and included files
    - Class data members and member functions
  - File I/O
  - I/O manipulators

Homework #2

- Target due: Week #4 (Tuesday, 10/01)
  - A command line reader
  - Thorough understanding of "pointers"
  - Basic program design
  - Ref code: 840/950 lines C++ (last year's)
    - Ref src / Ref prog.
  - New feature(s) may be added...

2. C++ Review - More on Functions, Variables, Classes

- Part I: Understanding "Functions"
  - Global vs. member functions
  - Function signature, prototype , definition
  - Function parameters, arguments
- Part II: More on "Variables"
  - "const" keyword
  - Array vs. pointers
  - Pointer arithmetic
  - Memory sizes of variables
  - Return value of a function
  - Compilation issues
  - Compiler preprocessors

2. C++ Review - More on Functions, Variables, Classes

- Part III: More on "Classes"
  - Class, struct, union, enum
  - Bit-slicing
  - Class wrapper
  - "static" keyword

3. C++ Review – Overloading and Polymorphism

- Class inheritance
  - Access privilege: private/protected/public
  - Virtual function and polymorphism
  - Abstract class and pure virtual function
  - Data encapsulation
  - Multiple inheritance
- Function overloading
- Operator overloading
- Class template class
- Template function
- Functional object

Homework #3

- Target due: Week #6 (Saturday, 10/19)
  - Complete command interface and a simple database system
  - Learn how to read a formal spec
    - Homework description file: 19 pages
  - Learn how to write a structured code
  - Ref code: 2215(2541)/2959 lines C++
    - Ref src (Ref src + hidden files)/ Ref prog.

4. Memory Management and Exception Handling

- Memory related problems
  - Illegal memory address access
  - Memory leaks
  - Fragmentation
  - Performance issues
- Memory management
  - Basic concept
  - Categorization
  - How to implement
  - Basic concepts of data structure
- Exception Handling
  - Try, throw, and catch
  - Interrupt handling

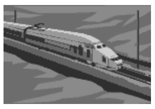
Homework #4

- Target due: Week #8 (Sunday, 11/03)
  - Memory management
  - Computer architecture concept
  - Pointers (again), basic data structure
  - Ref code: 1478(2869)/3038 lines C++
    - Ref src (Ref src + hidden files)/ Ref prog.

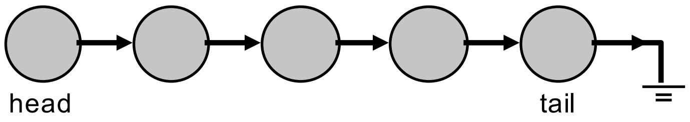
5. Computational Complexity

- Review of complexity analysis
- Why should I care?
- What's the most frequently encountered problem?
- What's your best bet?

6. Dynamic Array vs. Linked List

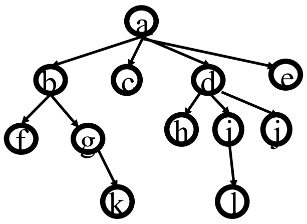


- Abstract data types
- Linear data types
- Static vs. dynamic array
- Why dynamic array? Why not linked list?
- How to evaluate their performance?
  - Runtime vs. memory usage



7. Tree (Part I)

- Non-linear data types
- Decision trees
- Tree traversal
- Balanced trees
- Implementation issues



Homework #5

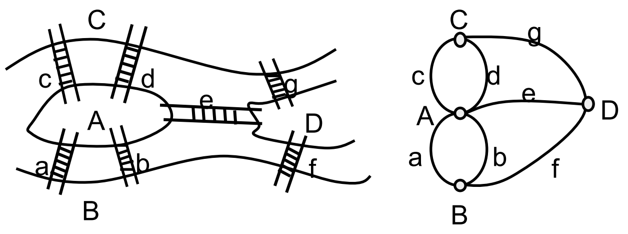
- Target due: Week #11 (Monday, 11/18)
- Implementation and comparison of various data structures
  - Linked list
  - Dynamic array
  - Binary search tree
- Ref code: 1520(2902)/3327 lines in C++
  - Ref src (Ref src + hidden files)/ Ref prog.

8. C++ Review - More on IO Streams

- More on I/O manipulator
- Formatted and unformatted I/O
- States and flags in I/O streams
- Tying I/O streams
- File pointers
- Random access files
- Stringstream and streambuf

9. Graph and Circuit

- Tree vs. graph
- Basic graph theories
- Graph traversal problems
- Loop handling
- How to design data structure for a circuit netlist?



Homework #6

- Target due: Week #13 (Tuesday, 12/03)
- A circuit parser
  - I/O and file streams
  - Graph/Circuit data structure
  - Hash/Map usage
  - Boolean logic
- Ref code: 1535(2917)/4189 lines in C++
  - Ref src (Ref src + hidden files)/ Ref prog.

10. Special Topic: Lex and Yacc

- What is a (programming) language?
- Lexical analysis
- Syntactical analysis
- Language parser
- Tutorial: an command-line calculator

11. Programming on Linux Workstations

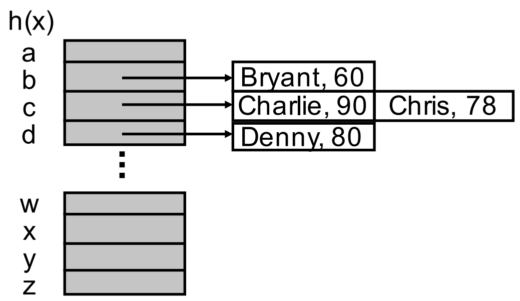
- Why Linux? Why not MS Windows?
- History of Linux OS
- Basic survival guide on Linux
- Writing programs on Linux
  - Shell commands
  - Compiler
  - Makefile
  - Debugger

12. Heap, Set and Map

- Review of sorting algorithms
- Review of binary (balanced) trees
- Complexity analysis
- Alternative ways of implementation
- Standard Template Library (STL) revisit

13. Cache vs. hash

- Review on hash
- Alternative to hash
- What's the difference?
- Computational cache/hash

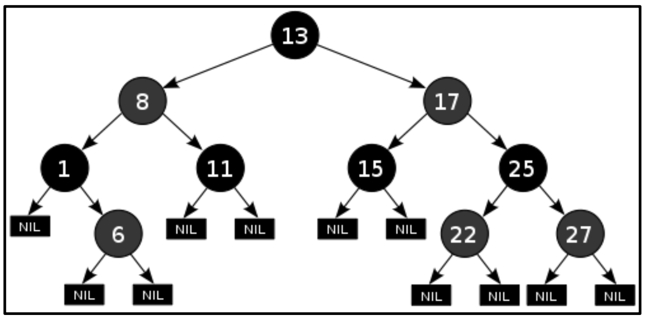


Homework #7

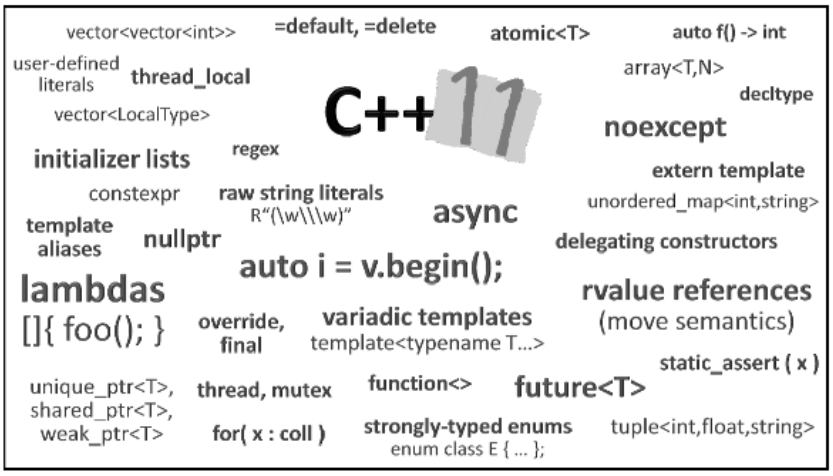
- Target due: Week #15 (Wednesday, 12/19)
- Implementation and practical applications of various data structures
  - Heap
  - Hash
  - Cache
- Ref code: 1544(2926)/3114 lines in C++
  - Ref src (Ref src + hidden files)/ Ref prog.

14. Tree (Part II)

- Red-Black Tree
- 2-3-4 Tree
- Splay Tree
- B-Tree



15. Special Topic: C++11 (TBD)



img_src: https://christophep.files.wordpress.com/2011/11/c-11.png

Final Project

- Functionally Reduced And-Inverter Graph (FRAIG)
  - Read in a circuit netlist (HW6)
  - Perform circuit optimization (graph operations)
  - Simulate the circuit (graph traversal, Boolean operations)
  - Collect functionally equivalent candidate pairs (efficient hash implementation)
  - Define the “magic number” to control the program flow (engineering sense)
- Ref code: 4822(6204)/8255 lines in C++
  - Ref src (Ref src + hidden files)/ Ref prog.
- 30% of the final grade!! Please start earlier!!

益華電腦贊助 NTU DSnP 期末專題

Homework Assignments and Final Project

- Once again, get yourself familiar with the C++ programming on Linux ASAP!!
  - You MUST compile your code on Linux or OS X environment.
  - g++ compiler is a MUST
- Homework turn-in
  - Through NTU Ceiba class website
  - Please pay attention to the rules on the class website
  - Filenames, compression rules, etc.
- No copying/pirating
  - If happens, -20 for your term grade!!
- Don’t miss any homework!!
  - ~10% of your term grade...
- Do not delay
  - 1 day → - 1/3
  - 2 days → - 2/3
  - 3 days and up → 0

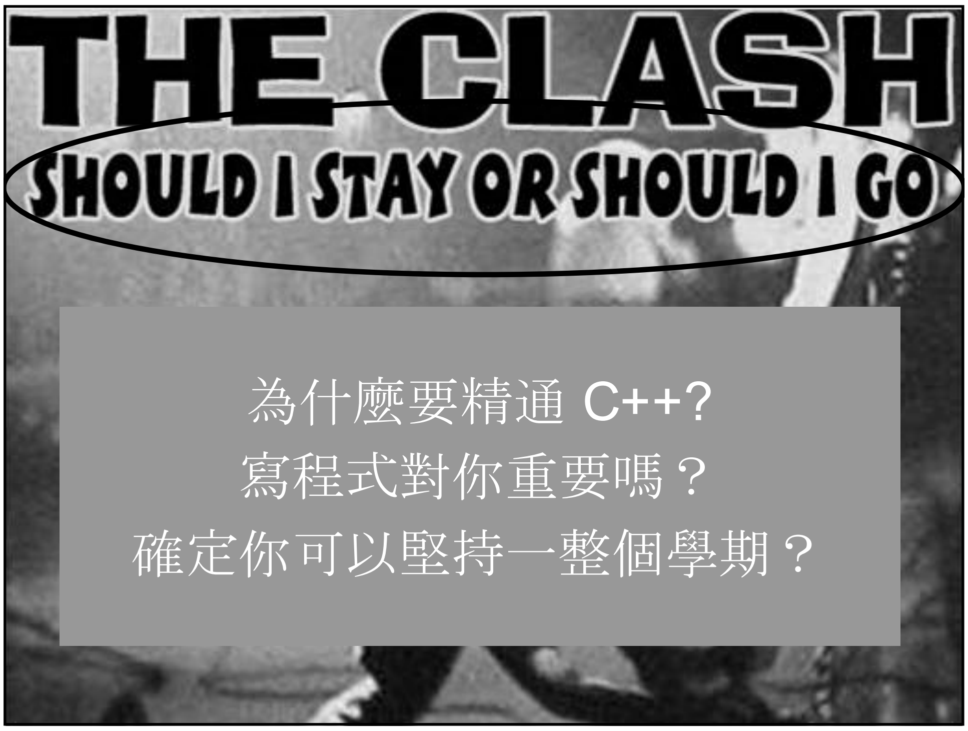
聽說這門課很操，是真的嗎?

◆ 不要懷疑，根據多次的問卷統計，同學們覺得這門課的 loading 大約 >= 9 學分，每兩個星期要花 20 ~ 30 hours (以上) 在作業上。

因為我覺得台大的學生根本修太多主科了!!

你可以去修很多其他領域的課，跨領域學習，增廣見聞；

但你如果想要把一些專業科目學好，我覺得一學期應該修兩三門就好，然後每門課九學分 (誤)!



為什麼要精通 C++?

寫程式對你重要嗎？

確定你可以堅持一整個學期？

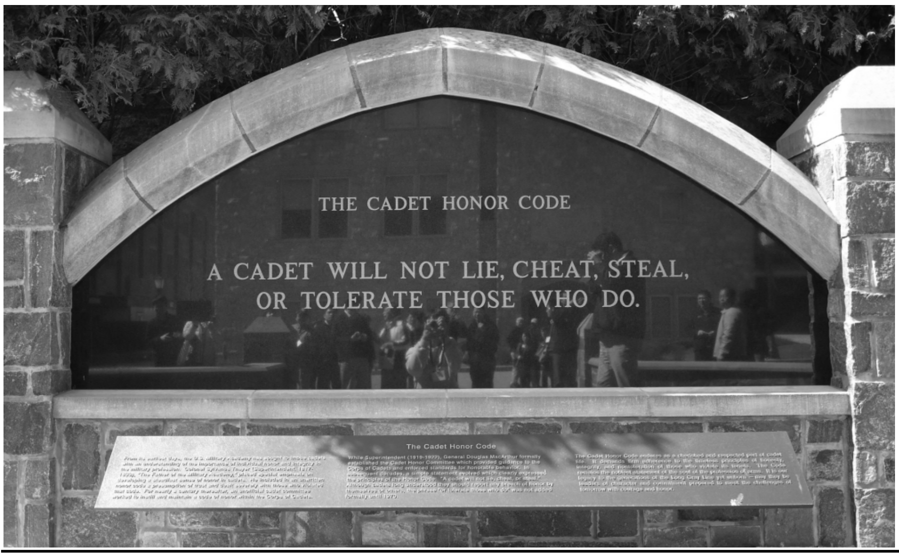
雖然這門課很操…

- ◆ 但好處是沒有期中 & 期末考，不用去 K 教科書或是消習題。
  - ● 不過有期末 project
  - ● 而且要學會自己找參考資料
- ◆ 所以如果你還要忙社團或是要參加什麼隊的，或是其他的課很重，請搞清楚你的 availability，切莫**始亂終棄**!!
- ◆ 我的目標是：同學們在修了這門課之後除了對於資料結構能有正確的觀念之外，起碼要有自行 handle 1000 行程式碼的信心!

我是個寫程式的小嫩咖，我有辦法修這門課嗎?

- ◆ 原則上絕大部分的人在你們這個年紀都是寫程式的小嫩咖，所以我想沒有問題。
- ◆ 重點還是要能有每兩個星期交一個作業，連續14 周，然後再加上一個期末專題的 "**commitment**"
  - ● 再強調一次，要考量現實，不要輕易相信自己的意志力可以戰勝一切!
- ◆ Commitment 從何而來?
  - ● 首先，請確定 "把程式學好" 對你的重要性
  - ● 再來，請確定自己可以接受 "學習比成績重要"
  - ● 還有，請發誓自己 "寧願被當，也不會抄襲"

DSnP Honor Code


THE CADET HONOR CODE
A CADET WILL NOT LIE, CHEAT, STEAL,
OR TOLERATE THOSE WHO DO.
The Cadet Honor Code

DSnP Honor Code

◆ 上課要專心，寧願翹課也不要來課堂做別的事

◆ 作業不抄襲，寧願被當也要從頭到尾自己寫

DSnP Honor Code

- 上課要專心，座位有限，寧願翹課也不要來課堂做別的事，佔用學習資源
  - 不點名，學生有自行決定如何學習的自由
  - 但是如果你是來教室上臉書、打電動、睡覺補眠，那對我是種不尊重，對同學也有不好的影響。
  - 如果你覺得上課的內容你都已經會了，就請不要貪圖這個學分，把座位讓給別人，或者，你也可以不用來上課。
  - 不過，上課用電腦寫寫小程式，驗證上課所學，或者是上網查詢相關資料，是被鼓勵的

DSnP Honor Code --- 關於抄襲

- Definition: 所謂「抄襲」，就是將別人部分或是所有的 code, 用 copy/paste, 或是看著 code 跟著打的方式，變成自己的作業的一部份
  - 歡迎互相討論，甚至拿別人的 code 來 study 也不會/無法禁止 (雖然這樣並不好)，但最後一定要自己獨立的寫。
- 我們有強大的抓抄襲的程式，會對所有的作業以及之前學長姊的作業去做比對，如果沒有抄襲，相似度都會很低，但如果有抄襲，不管你是改變數名稱，還是換 statements 順序... 等等，我們都可以很容易抓出來，所以請勿抱著苟且的想法。
  - 以我們的作業複雜度而言，只要是自己寫的，一定一眼就可以看出跟抄襲的不同。
- 凡抄襲者不論多寡、理由，除該次作業 0 分之外，學期成績一律再扣 20 分 (調分後)

一些前車之鑑…

老師您好:

對不起老師...我對之前的作業有些抄襲or參考的疑慮，睡覺都睡不好，

所以還是先寄信詢問(自首)了...雖然我自認程度很輕微拉(爬過ptt對於抄襲的定義，覺得還好??)

自從在寫HW4快到尾聲時在網路上搜到疑似老師2012年DSnP的解答...相信老師都知道，因為實在太好搜了= =
之後我作業不懂的就會去看老師的code...

…

一些前車之鑑…

我覺得網路上那2012版的解答，雖然部分不夠完美(EX:HW6 gate定義覺得可以再刪減)，
但他對於我就像潘朵拉盒子，一載下來看，就是罪惡...可是當作業不懂時，他卻是最好的來源。

……
如果真的被處罰也很甘願，因為是我自己程式能力不足。

儘管如此...還是拜託老師開恩...即使有2012年的code，我每次作業也是會花20小時up，覺得努力沒有比別人少...
也常常跟同學討論code，當然都是based on對老師code的理解，再加上自己的詮釋。

最後，謝謝老師看完我很長的解釋文...感謝老師開DSnP，我學到的真的很多!!

一些前車之鑑…

(From Ric)
很遺憾的，你沒有在學期中我一再強調抄襲的定義的時候就主動承認，而前幾天問你的時候你也還是無法就直接承認你就是有抄襲。

因此，我只好按照學期初所說的規定，將你該次的作業算成零分，然後學期成績在調分後再扣 20 分，因此，你的成績將會變成 52分 (F). 希望你可以接受這樣的處置。

(Reply)
這次的經驗已經讓我飽嚐煎熬與苦頭

以後不只不敢再犯大概還會便成陰影警惕很久

這幾天也想了很多，那些文過飾非的話大概不只是想要粉飾太平，一部分也是因為內心本來就有所愧疚想要說服自己吧

正如教授所說：不只沒有遵守規定，我還欠缺更多勇於認錯承擔的態度

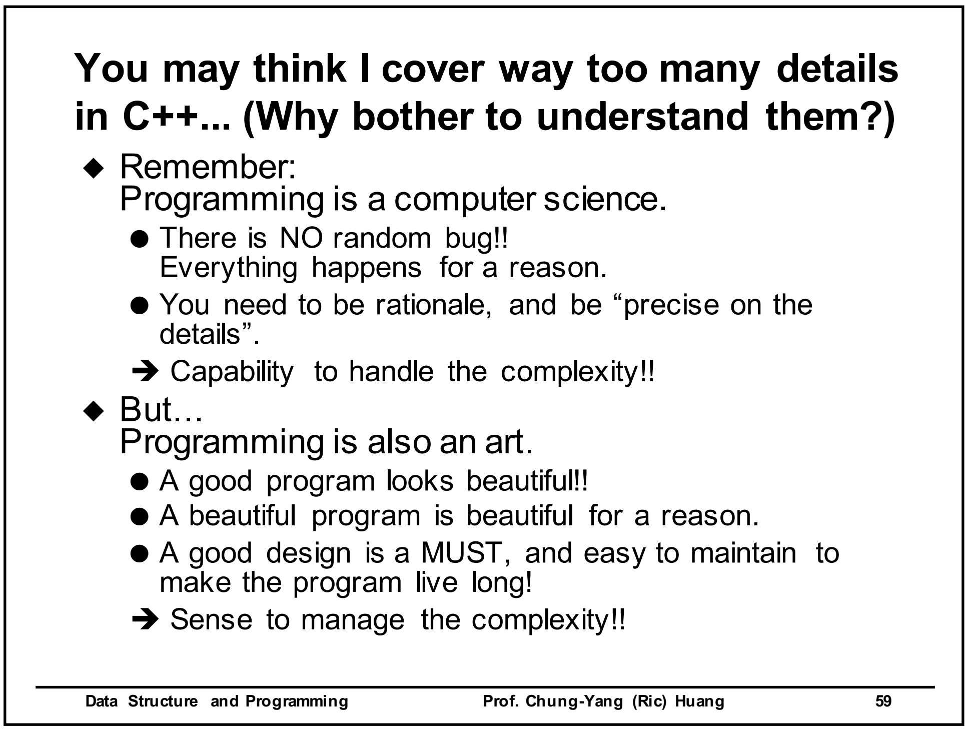
謝謝教授，還願意耗費時間跟我說這麼多。

A short version of "Computer Programming" class?

◆ NO!!

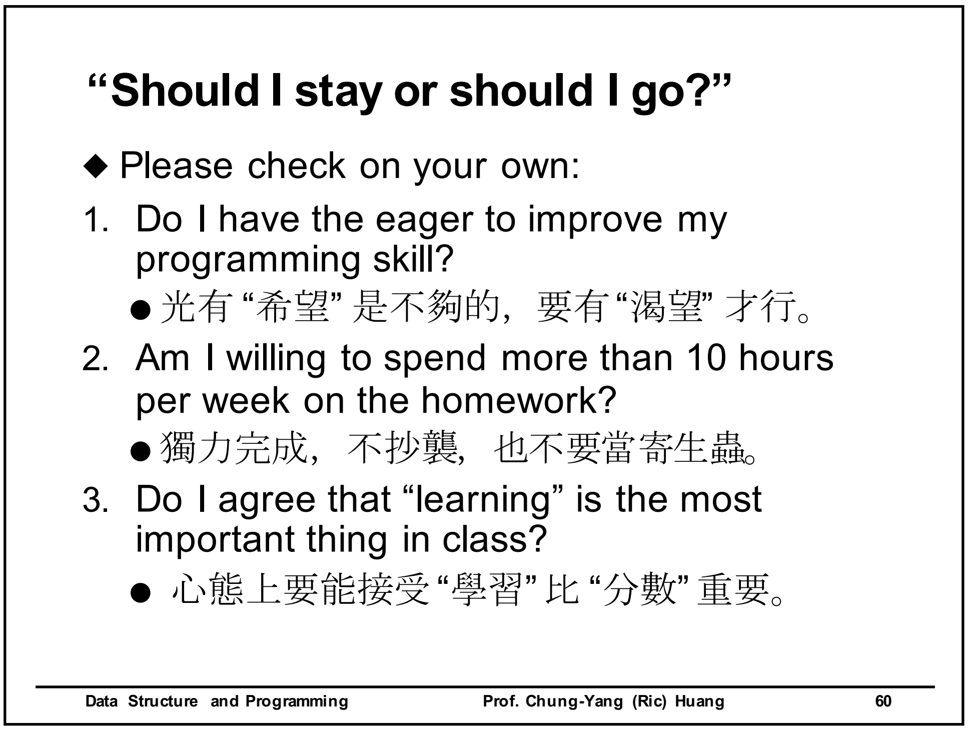
◆ If you don't have any background in C++ (or C) ...
  - You probably have chosen the wrong class.

◆ If you are poor in C++ programming...
  - Well, you are definitely NOT the only one, so you are very welcome!!
  - Please pay attention to the lectures in this topic, and make sure you can commit enough time on homework



You may think I cover way too many details in C++... (Why bother to understand them?)

- ◆ Remember:
  Programming is a computer science.
  - ● There is NO random bug!!
    Everything happens for a reason.
  - ● You need to be rationale, and be “precise on the details”.
  - ➔ Capability to handle the complexity!!
- ◆ But…
  Programming is also an art.
  - ● A good program looks beautiful!!
  - ● A beautiful program is beautiful for a reason.
  - ● A good design is a MUST, and easy to maintain to make the program live long!
  - ➔ Sense to manage the complexity!!



“Should I stay or should I go?”

◆ Please check on your own:

1. Do I have the eager to improve my programming skill?
   - ● 光有 “希望” 是不夠的，要有 “渴望” 才行。
2. Am I willing to spend more than 10 hours per week on the homework?
   - ● 獨力完成，不抄襲，也不要當寄生蟲。
3. Do I agree that “learning” is the most important thing in class?
   - ● 心態上要能接受 “學習” 比 “分數” 重要。

千萬不要認為 CS 很熱門就跑過來修 DSnP…

除非你想下猛藥來確認自己適不適合當軟體工程師…

歡喜修課，甘願承受

- 說實在的，DSnP 是 NTU(EE) 的奇蹟!
  - 需要大家共同的珍惜
- 非誠勿試，please!!