H.O.#1
Fall 2015
Gary Chan

COMP2012H:
Honors Object-Oriented Programming and Data Structures

Gary Chan

Professor, Department of Computer Science and Engineering

HKUST

Who Is This Guy?

- Computer Science and Engineering
  - Computer networks (Multimedia and wireless networking)
  - Work a lot on system programming in C++/C (buffer management, scheduling, I/O processing, etc.)
- Office: 3507 Academic building (Lifts 25/26)
- Email: gchan@cse.ust.hk
- Phone: x6990
- Office hours: By appointment

Your TAs

David Au

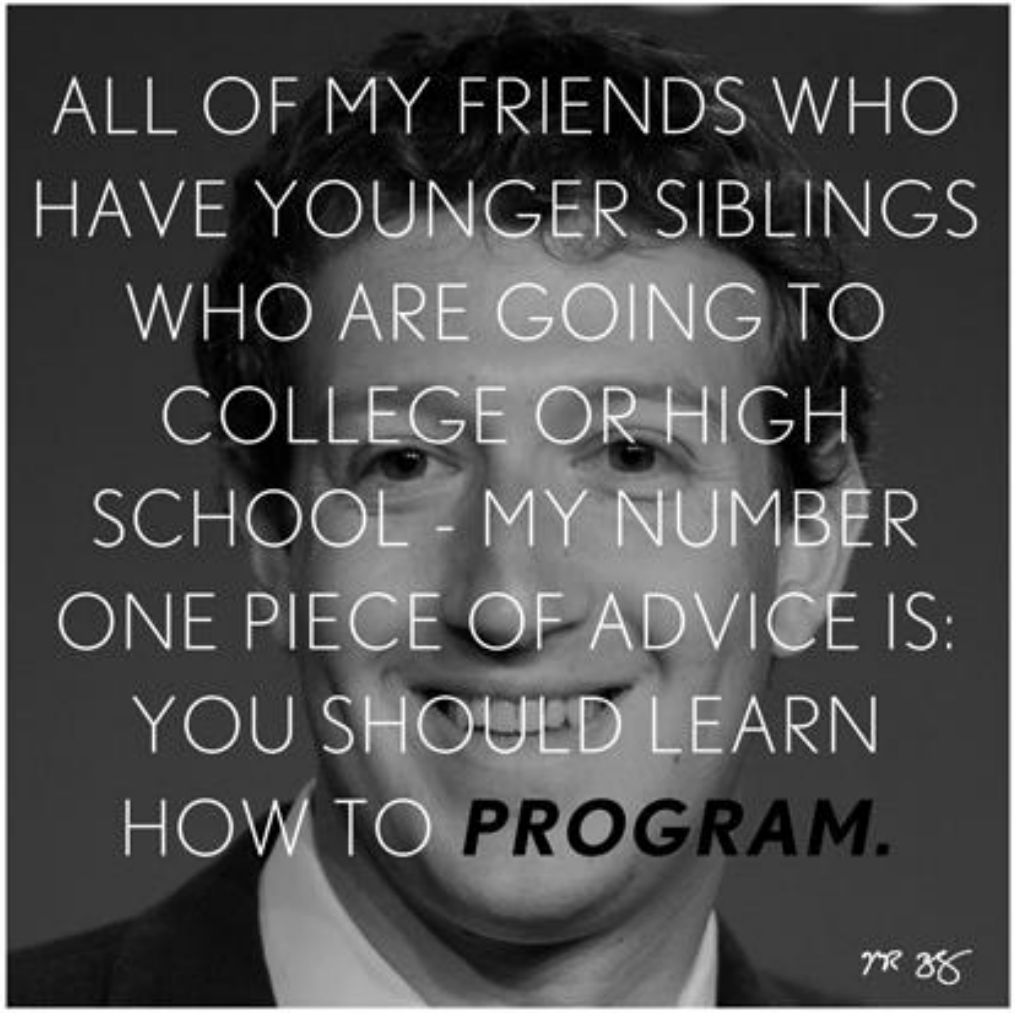
Henry He

Jiajie Tan

COMP 2012H: A C++ Programming Course

- Lecture time:
  - L1: TTh 12:30pm-2:20pm, Room 2406 (17/18)
- Bookmark this: https://course.cse.ust.hk/comp2012h
  - Lecture materials, lab materials, assignments and solutions
  - Responsible TAs for labs and assignments
  - Always under construction: Download the materials a week before each class; peek at the materials advanced of classes
- Lab: Th 2:30-4:20pm
  - UG Windows lab (Room 4210, Lift 19). We will use Linux environment for our C++ compilation. The TAs will teach you how to do that in the first lab.

Why Should I Care about Programming? Facebook CEO: Mark Zuckerberg



COMP2012H = COMP2011 + COMP2012

- "2-in-1" package deal
- Finish COMP2011 in the first half of the semester, followed by COMP2012 in the next half
  - Not depth, but sheer speed
  - A highly accelerated course
- This is a "filtered" honors course
  - Must have at least A in COMP1021/1022P/1022Q/ISOM3230
  - If you took COMP1022Q/ISOM3230, it is better for you to have taken COMP1029C as well.
  - Excellent knowledge on a language and its programming fundamentals
  - More importantly, you need to be smart, and have excellent and substantial programming background and experience before
- Basic mathematical skills equivalent to F.6 standard
  - Doing sum series, GP (geometric progression), taking logs, etc.

A Very Rigorous Accelerated Course

- Covering C++ syntax, object-oriented programming and how to use it to implement some important data structures and applications
  - These data structures enable advanced algorithms and applications and make them particularly efficient (COMP 3711)
  - List, queue, stack, hash, (binary) trees, etc.
- Mercilessly fast-paced, expecting large appetite for knowledge
  - You have to learn fast to keep up with the course schedule
  - It is not an easy sequence, and not for the fainted-hearted
- Programming-intensive and thinking-intensive course
  - You need to think smart and program smart in order to keep up
- The labs and lectures are chasing each other in random order over the semester
  - Not difficult: warm-up exercises to consolidate basic concepts
  - Do NOT expect that lectures always lead the labs; in fact, they sometimes do not
  - That means that you may need to self-learn for your labs
  - Very important to complete the labs yourself in order to do well in the course

Major Course Topics

1. C++ variables and operators [1 week]
2. Flow control [0.75 week]
3. Arrays [0.5 week]
4. Functions and recursions [1.5 week]
5. Pointers, dynamic objects and memory management [1 week]

Building blocks for OOP

6. OOP concept and classes [1.25 week]
   - Object creation, destruction, member variables and functions
7. Basic data structure as ADT class [1 week]
   - List, stack, queue and their applications
8. Generic programming [1 week]
   - Function and operator overloading; function and class templates
9. Standard template library (STL) [0.75 week]
   - list, queue, stack, string, vector, etc.
10. Inheritance, polymorphism and virtual function for code reuse [1.25 week]

OOD

11. Binary tree, binary search tree, AVL tree, data structures and operations [1.25 week]
12. Hash search [0.75 week]

Data structures (with topic 7 as well)

How Hard Should I Work?

- Remember that it is a 5-unit course
  - Normally, the workload of a 3-unit course is about 9-12 hours/week. As it is a 5-unit course, expect about 15-20 hours per week, i.e., 9-14 hours of workload outside classes
  - That means you need to make sacrifice on some activities
- Programming courses often are time-consuming
- This is a rigorous course, and you are excellent students
  - Don't expect to be spoon-fed! You are supposed to learn materials beyond the lecture notes
  - You should cultivate an independent and study-ahead learning habit
  - The lectures and labs only provide you basic fundamentals
  - The assignments will challenge you to apply the principles to reach the next level, and the examinations to the next level.

Tips on Effective Learning

- Effective programming
  - Macro-planning (30-40% of time): algorithm design, refinements, modularity, roadmap
  - Micro-coding (60-70% of time): syntax, logic, debugging, focusing more on implementation details
  - In order not to ruin your night, strive to minimize your bugs by testing your program one module at a time!
- Effective learning
  - Prestudy (15-30 minutes): rough ideas on what the lecture would be covering
  - Pay attention in class: ask questions if you don't understand; try to follow as closely as possible
  - Post-study (2 hours): notes organization, book reading, practice, etc.

6 Effective Learning Strategies

1. Take notes by hand, even if class notes have been provided
2. Paraphrasing and rewriting the notes shortly after a lecture
3. Form study groups
4. Read textbooks (or good websites)
5. Try to do homework problems on your own. Then discuss with your study group
6. Teaching the materials to one another

Learning and Teaching Strategies
(Hoffmann and McGuire, American Scientist, Volume 98, Number 5, September-October 2010
http://www.americanscientist.org/issues/num2/2010/5/learning-and-teaching-strategies/1)

Lecture and Assignments

- Lectures
  - Slides
  - Illustrative examples on board to supplement the slides and transparencies
  - Feel free to interrupt to ask questions
  - Not compulsory, but it is your responsibility to catch up with your missed lectures with your friends
- A total of 5 programming assignments and 1 written assignment
  - Bi-weekly, each takes about 20 hours
  - C++
  - Individually done
  - More rigorous problems to consolidate your knowledge

Programming Assignments

- Run in Linux environment
  - Your first few labs will cover the environment and how to write C++ programs in the environment
  - Our testing will be in the lab machines of the environment; we will *not* entertain porting problem if you write on other platforms
- Submit the assignments by the deadline yourself
  - Course assignment submission system (CASS)
  - For more details on how to use it, please visit http://cssystem.cse.ust.hk → Undergraduates → CASS User Guide
- Writing programs is like learning a language or a musical instrument: you need to regularly and diligently practice it in order to master it.

I Cannot Help You Debug…

- Hi, I'm a student from COMP2012. I've finished my PA2 and it's working perfectly without any problem on my Visual Studio 2008. But when I tested it in the linux environment using g++ compiler there occur many problems…
- As I was doing the PA2, I came across a problem where the program was not able to delete my pointer and it was getting stuck at that point. I checked if the pointer was null but it was not. I have pasted the code below and the output that I got before the program stopped…
- what is meant by : ..\BigInteger.h:54: error: field `digitList' has

incomplete type ?? I cannot fix it out and I need your help. Enclosed is the source files and header files if you need.

Lab

- Programming exercise to consolidate your basic understanding of course materials
  - 1 point for attendance
  - 2 points for solution: 0 (not done/submitted), 1 (good but some errors/bugs), 2 (excellent with no detected errors/bugs)
- Exercises can be graded in the lab time
  - 1 point for attendance
- Submit your lab through CASS by 7pm on Friday that week for it to be graded
  - Wrong or delayed submission to a directory leads to no mark

Grading

- 5 individual programming assignments (30%, 6% each)
  - Array and control flow, functions and file I/O, pointers and objects, generic programming, and inheritance/polymorphism
- 1 written assignment (5%)
- Lab exercises (10%)
  - 1% each, your best 10 lab scores
- Midterm (20%)
- Final (35%)
- Your score of the final should be consistent with your overall score over the semester. In other words, fine-grade adjustment may be made at the end based on the score of your final
  - If you perform exceptionally well in the final, you may not fail the course
  - If you perform very poorly in the final, you may not get an A
- This is an honors course, and you are honors students
  - Strong background, motivated, smart and independent
  - Graded out of the university guideline (expecting most of you will get As or Bs)

Midterm and Final

- There are a mid-term and a final
- Individually done
  - Unassisted by living objects
  - You have to sign "I have not violated the University honor codes in this examination" before we grade your booklets
- Closed-book, closed-notes, no calculator
- 1-page "cheat sheet"
  - Written, drawn or type-set on both sides (no magnifying glass is allowed in the exam venue)
  - Feel free to share your cheat-sheet with others
- No early or late examination
  - Unless under very unusual circumstances with solid proofs
  - Reasons like over-sleeping, society or extra-curricular activities, or traffic congestion are *unacceptable* excuses
  - You need to inform me beforehand for re-scheduling

Regrade Requests

- Only be entertained within 1 week after the graded assignments are returned
- Please approach your TAs responsible for the assignments directly

Email Policy

- General course/lecture questions: to me
- Assignments and labs: to the responsible TAs
- Remember: our mailboxes are of limited size
- Use email unless it is necessary
  - Not effective to explain things
  - Visit my or TA's office hours
- Please do not expect answers right away
- Please do not send us codes for debugging
  - We will not debug codes for you

Thou Shalt Not Cheat

- I encourage you to
  - Discuss your approaches with each other
  - Check your *answers* with your classmates or friends *after* you have done the assignments
  - This is an effective way of learning
- After learning, put everything in your own words
  - Do NOT take short cut by copying
  - Copying is stealing (stealing intellectual property), which is a crime
  - You are too smart to copy
- You are also responsible to protect your work (intellectual property)
  - Do NOT give your codes or work to others. This will tempt them to take short cuts.
- What if you are caught copying?
  - Both the copier and the originator get 0
  - 2nd time: Both get 0 *and* one full downgrade
  - Caught 3rd time: FAIL course
  - If it is an examination, an automatic FAIL

Textbooks

- The best way is to practice and consult on-line manual while programming (http://www.cplusplus.com/)
- Main books
  - Larry Nyhoff, ADTs, Data Structures, and Problem Solving with C++, Prentice Hall
- References
  - Michael Main and Walter Savitch, Data Structures & Other Objects Using C++, Addison Wesley
  - Deitel, "C++: How to Program," 6th edition, Prentice Hall
    - We assume a C++ background at the level of Chapters 1-2, 4-8, 15, 17
  - Lippman and Lajoie, "C++ Primer," 3rd edition, Addison Wesley
    - We assume a C++ background at the level of Chapters 1-5, 7-8
  - Eckel, "Thinking in C++," Vol. 1 & 2, Prentice Hall (electronic version of this book is available at http://www.BruceEckel.com)
  - Stroustrup, "The C++ Programming Language," Addison Wesley
  - Ivor Horton, "Beginning C++: The Complete Language," WROX
  - Savitch, "Problem Solving with C++: The Object of Programming," Addison Wesley
  - Davidson, "C++ Program Design," McGraw Hill
  - D'Orazio, "Programming in C++: Lessons and Applications," McGraw Hill
  - Kernighan and Ritchie, "The C Programming Language," Prentice Hall

Good Coders: Rise of The Machines…