Outline

• Course Administration
  – Break 1, Clicker, CP2.9 order (~6.30-6.45pm)
• Competitive Programming Book (2.9th Ed), Ch 1
  – Competitive Programming: Live Demo
  – Tips to be Competitive: Hands on 😊, join me
  – Break 2 (~7.50-8.00pm)
• Mooshak: First Mock Contest & Discussion
  – 45 minutes contest: 2 “easy” + 1 “medium” problems
Course Administration (1)

• Teaching Staffs:
  – Lecturer:
    • Dr. Steven Halim: stevenhalim at gmail
      – Add me in Facebook if you haven’t done so
      – www.facebook.com/groups/236210576509653
      – uhunt.felix-halim.net/id/32900
    • Email subject format: “[CS3233]-MESSAGE”
    • Office & Phone: COM2-3-37 & 6516-7361
  – Teaching Assistant:
    • Huang Da: a0091847 at nus edu sg
Course Administration (2)

• Modular Credit: 4
• Not a hard module (to score)*
  – No Final Exam (yippie)
  – “No” bell curve grading system
  – Grade History:

<table>
<thead>
<tr>
<th>S2 of AY</th>
<th># Stu</th>
<th># A+/A/A-</th>
<th># B+</th>
<th>&lt; B+</th>
</tr>
</thead>
<tbody>
<tr>
<td>2008/09</td>
<td>21</td>
<td>19</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2009/10</td>
<td>12</td>
<td>11</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2010/11</td>
<td>17</td>
<td>13</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2011/12</td>
<td>24</td>
<td>15</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

58/74 = 78% students got A- or above 😊
68/74 = 92% students got B+ or above 😊
Course Administration (3)

• Class timetable:
  – Wednesday, 6-9 (or 10*) pm @ COM1-B-PL2
  – Week01 to Week13, minus recess week

• Course Website:
  – IVLE, search CS3233 (of S2 AY 2012/13)
    • http://www.comp.nus.edu.sg/~cs3233 is not used
  – Details about Syllabus/Lesson Plan/Workload/Assessment are all there
CS3233 vs CS1010/CS1101S

• In CS1010 (or CG1101)/CS1101S
  – You learn how to create programs

• In CS3233
  – I assume that you are good at that
    • You can use C++ or Java for contests/homework
      – C++ is the main language in CS3233
      – Java is the second language in CS3233
      – However, mastering both is an advantage in CS3233
      – Scheme (CS1101S) and Pascal are NOT supported!
    • Please reconsider if you do not like coding!
CS3233 vs CS1020/CS2020 1st half

• In CS1020/CS2020 1st half
  – You learn some well-known linear data structures and algorithms

• In CS3233
  – We only revisit them in the first two weeks++
    • Then, we show how those DSes and algorithms can be coded into computer programs quickly and effectively
    • Please reconsider if you passed this module with difficulty (although you scored A- or above)!
CS3233 vs CS2010/CS2020 2nd half

• In CS2010/CS2020 2nd half
  – You learn one-fifth of CS3233 materials
    • Graph (~4 weeks/lectures)
    • Dynamic Programming (~3 weeks/lectures)

• In CS3233
  – We revisit them in just one and half weeks (Week04 + Help session of Week05)
    • With emphasis on implementation speed...
    • Then we will learn much more 😊...
      – The pre-req of CS3233 is A- in CS2010/CS2020;
        **CS3233 is now designed with this assumption** (verbal details)
CS3233 versus CS3230

• In CS3230
  – You learn more well-known algorithms
    (mostly analysis/proofs)

• In CS3233
  – We learn how to implement them plus several (lots?)
    additional topics outside CS3230 (hands-on)
    • We do not discuss their theoretical background in depth!
    • Please reconsider if you are not prepared
      to learn many new things
  – Side note: It may be good to take both
    CS3230/CS3233 in the same sem (verbal explanation)
CS3233 versus Other Modules

• Obtaining \(~52\%\)^{*} (verbal explanation) from total marks is already enough to get a good grade (B+)
  – In normal module, this is usually a ‘C+’ grade... 😞
  – I repeat, in the last four years:
    • \(~78\%\) got at least A- 😊 and \(~92\%\) got at least B+ 😊

• Higher chance* to represent NUS in the Annual ACM International Inter-Collegiate Programming Contest (ICPC)
  – Free trip(s) to a few Asian countries^ – Regionals 2013
  – And perhaps to World Finals 2014
In CS3233
but not in other SoC CS modules

- This information is true for S2 AY 2013/2014 version:
  - Heavy usage of bitmask operations (for backtracking, DP, etc)
  - Binary Indexed (Fenwick) Tree
  - State-Space Search
  - Meet in the Middle (Bidirectional Search)
  - Various DP tricks (*much more* than CS3230/2020/2010)
  - Network Flow and Graph Matching (not in CS2020/2010
    but maybe in CS5234 – Comb and Graph Algorithms)
  - Various Mathematics algorithms and tricks
  - Suffix Array (CS5238 – Adv Comb Methods in BioInfo)
  - Convex Hull, Cut Polygon (CS5237 – Comp Geo & Apps)
  - And some mysterious stuffs...
CS3233 Lecturer History

• Initiated by Prof Andrew Lim (CUHK): 1999-2001
  – Vacuum in AY 2002/03... 😞
• Between 2004-2006, CS3233 was taught by A/P Leong Hon Wai and A/P Ooi Wei Tsang
  – Another vacuum in AY 2007/08... 😞
• Revived again* on semester 2, 2008/09 😊
• Note: Each lecturer has different style...
  – Mine is geared towards ICPC (and also IOI) preparation
  – But have now been calibrated to match the level of typical second upper/first class students in NUS
SoC Teams Performance History (1)

- ACM ICPC World Finals
  - 1999: Joint-18
  - 2000: Joint-22
  - 2001: Joint-29
  - 2003: Joint-13
  - 2005: Melvin, Junbin, Yunsong: Hon. Mention
  - 2010/Harbin, China: Duc, Tien, Phong: Hon. Mention
  - 2011: Miss out by 2 ranks 😞
  - 2012/Warsaw, Poland: Zi Chun*, Harta*, Phuong*: Hon. Mention
  - 2013/St Petersburg, Russia: Harta*, Phuong*, Sy Nguyen*
SoC Teams Performance History (2)

• Recent ACM ICPC Regional Contests
  – 2009: 7th & 10th in Jakarta; 3rd in Manila; 2nd and 10th in Phuket
  – 2010: 10th in Daejeon; 6th* in Kuala Lumpur; 10th in Tokyo
  – 2011: 7th* in Phuket; 5th* in Kuala Lumpur
  – 2012: 3rd in Jakarta; not so lucky in Hat Yai
  – 2013: YOUR TURN for World Finals 2014 (or 2015)!

• More history in:
SoC Current Strengths

• Teaching Staffs and Seniors:
  – A/P Tan, Dr Steven, Suhendry, Zi Chun*, Harta Wijaya*, Tuan Phuong*, Sy Nguyen*, etc
    • * ex/current-World Finalists currently in SoC
  – Singapore IOI Teams 2010-2012
    • 2 Golds, 4 Silvers, and 5 Bronzes by this team over the past 3 years
  – Leaving/already left/not in SG:
    • Minh Duc (@ FB), Duc Phong, Hoanh Tien, Victor Loh (@ FB), Felix Halim (@ Google), Su Zhan (@ Google)

• Current Students:
  – Many potential students (YOU ALL)...

CS3233 - Competitive Programming,
Steven Halim, SoC, NUS
Textbook

- Competitive Programming 2.9
  - It will be CP3 by May 2013, *with your inputs*
- COMPULSORY!!
- 30 SGD/copy (for fresh purchase)
  (15 SGD discount if you can show me that you already have CP2.5/CP2/CP1)
- You can pre-order tonight, I will print by Week02
  - Ch1.pdf has been sent to you on 1 Jan 2013
- Public version (simplified form), is in:
  [http://sites.google.com/site/stevenhalim/home/material](http://sites.google.com/site/stevenhalim/home/material)
Clicker Distribution + CP2.9 Order (15 Minutes Break)

• During the break:
  – I will distribute clickers
    • In this small class, each of you must take one clicker
  – You can discuss administrative issues with me (i.e. should I take this module or not?, etc)
    • In NUS, drop with ‘W’ grade is after Week 02!
  – Order “Competitive Programming 2.9” textbook
This is the new standard.

COMPETITIVE PROGRAMMING 2.9
Who have read CP2.5/2/1?

1. I have only pre-order CP2.9, so 0 page so far...

2. Most of chapter 4 and a bit of chapter 3 due to CS2020/CS2010 (CP2.5)

3. I have read and understand most of it, but I want to know what are the new stuffs in CP2.9 😊
Competitive Programming

• Given **well-known** Computer Science problems, solve them **as fast as possible**!
  – Not about “software engineering”
  – Solve judge’s test data correctly
  – Run fast enough
  – Well-known = not research problems!
  – Problems in our target contests (ACM ICPC & IOI) have this characteristic!
Demo (UVa 10684 – The jackpot)

- This exaggerated demo illustrate contestant’s type:
  - The blurry one
  - Give up
  - Slow
  - Competitive programmer
  - Very competitive programmer
CP2.9, Chapter 1
(if you have read ahead, good 😊,
help the rest by answering the pop-quizzes)

TIPS TO BE COMPETITIVE
Tip 1: Type Fast & Correct

• No kidding, this can be important!
• Let’s try
  – [http://www.typingtest.com](http://www.typingtest.com)
    • ZEBRA – Africa’s Striped Horse
  – Mine: ~85-90 wpm
  – Felix’s: ~55-65 wpm
• Familiarize yourself with the positions of the following keyboard keys:
  – (,),{,},[,],<,>,’’,”,&,,! etc
Tip 2: Identify Problem Types

- Ad Hoc
- Complete Search
- Divide and Conquer
- Greedy
- Dynamic Programming

- Graph
- Mathematics
- String Processing
- Comp. Geometry
- Some Harder Ones
Quick Test – Identification

• What is the type of this problem?
  – And how many minutes that you think you will need to solve this problem?

• Given an M*N \textbf{integer} matrix Q (1 \leq M, N \leq 50), check if there exists a sub-matrix of Q of size \(A*B\) (1 \leq A \leq M, 1 \leq B \leq N) where mean(Q) = 7?
Tip 3: Do Algorithm Analysis

• This is taught in more details in CS3230!
• In this module, we will just learn the basics required for dealing with ICPC/IOI problems
  – See the constraints in the problem statement
  – Conjure the simplest algorithm that works!
  – Do some basic analysis to convince that it will work before we start coding...
Tip 4: Master Prog Languages

- You should master at least one (preferably more) programming languages
  - Reduce the amount of time looking at references
  - Use shortcuts, macros, avoid comments
  - Use libraries whenever possible
- Idea: Once you figure out a solution for a problem, you are able to translate it into a bug-free code, and do it fast!
Tip 5: The Art of Testing Code

• Ultimately, we want “Accepted (AC)” verdict 😊
  – i.e. Our code passes the judge’s secret test data
• However, we may instead be given: 😞
  – Presentation Error (PE)
  – Wrong Answer (WA)
  – Time Limit Exceeded (TLE)
  – Memory Limit Exceeded (MLE)
  – Runtime Error (RTE)
Tip 6: Practice...

- Relevant Online Judges
  - MAIN: University of Valladolid (UVa) Online Judge
    - [http://uva.onlinejudge.org](http://uva.onlinejudge.org) (Open with Firefox!)
  - MISC: ACM ICPC Live Archive
    - [https://icpcarchive.ecs.baylor.edu](https://icpcarchive.ecs.baylor.edu)
  - MISC: TopCoder
    - [http://www.topcoder.com](http://www.topcoder.com)
  - MISC: USACO
    - [http://train.usaco.org](http://train.usaco.org)
I have...

1. Registered a free Uva account but have not solve anything
2. Have solved $\geq 1$ UVa problems
3. Have solved $\geq 10$ UVa problems
4. Have solved $\geq 40$ UVa problems

0 of 120
Tip 7: Team Work (ICPC Only)

• Practice coding on a blank paper
• Submit and print strategy
• Prepare test data challenges
• The X-factor
• You will experience such team work in mid semester and final team contests 😊
LET’S START OUR JOURNEY
Anatomy of a Problem

• Background story/problem description
  – Can be deceiving...

• Input and Output description
  – Usually written in formal manner
  – Most new problems are *multiple test cases* problems

• Sample Input and Sample Output
  – Usually very trivial, you need to come up with stronger/trickier test cases by yourself

• Hints or Footnotes
Quick Note: Ad Hoc Problems

• Read CP2.9 Chapter 1 by yourself
• We will spend our remaining time discussing more interesting stuffs...
Linear Algorithms (1)

• What is an \(O(n)\) solution for a certain problem?
  – Really just one pass through all \(n\) elements
    • e.g. find the min/max element of an array with size \(n\)
  – \(k\) passes through all \(n\) elements, e.g. \(O(kn)\), \(k\) is ‘small’
    • e.g. Find the second (\(k = 2\)) smallest element of an array
  – One (or ‘a few’) pass(es) through all elements with operations that runs in \(O(n)\) in \textit{amortized sense}
  – One (or ‘a few’) pass(es) through all elements, with a help of a logarithmic cost Data Structure/extra component, e.g. \(O(n \log n)\)
    • Adding a \(O(\log n)\) component to an \(O(n)\)-loop usually does \textbf{NOT} significantly increase the runtime under contest settings!
      – You can treat this as “bonus”... (important for IOI)
Linear Algorithms (2)

- Several “rare” topics solvable with $O(n)$ algorithms:
  1. Bracket Matching
  2. Postfix Calculator and Conversion (Shunting yard)
  3. (Static) Selection Problem
  4. Sorting in Linear Time
  5. Sliding Window

- I will present these problems *briefly*

- For more details about the solutions, please read Chapter 9 of CP2.9 on your own 😊
Bracket Matching

• ‘((()))’, ‘({})’, ‘(){}[]’ are correctly matched braces
• ‘((()', ‘{’, ‘)(’ are NOT correct
• Can we detect if a given expression of braces are correctly matched in O(n)?
Postfix Calculator

• There are three well-known types of algebraic expressions:
  – Infix (our default setting), e.g. $2 + 6 \times 3$, $(2 + 6) \times 3$
  – Prefix (Polish), e.g. $+ 2 \times 6 3$, $\times + 2 6 3$
  – Postfix (Reverse Polish), e.g. $2 6 3 \times +$, $2 6 + 3 \times$

• Can we evaluate a Postfix expression in $O(n)$?
Postfix Conversion

• Given an infix expression (that may contain parentheses), e.g. \((2 + 6) \times 3\), can we compute the equivalent postfix expression, e.g. \(2 \ 6 \ + \ 3 \ \times\) in \(O(n)\)?
Static Selection Problem

• Given a static (unchanged) array $A$ of $n$ elements, can we find the $k$-th smallest element of $A$ in $O(n)$?
  - e.g. for $A = \{2, 8, 7, 1, 5, 4, 6, 3\}$, $n = 8$,
    the 4-th smallest element is 4,
    the 8-th smallest element is 8, etc
Special-Purpose Sorting

• Given an array $A$ of $n$ small integers (each integer is between $[0..100]$), can we sort them in $O(n)$?
  – e.g. $A = \{0, 3, 0, 0, 1, 1\} \rightarrow \{0, 0, 0, 1, 1, 3\}$

• What if the given array $A$ has a range of 32-bit unsigned integers, e.g. from $[0..2^{32}-1]$?
  – e.g. $A = \{1, 1000000000, 2\} \rightarrow \{1, 2, 1000000000\}$

• Isn’t the lower bound of sorting is $\Omega(n \log n)$?
Sliding Window

- Given an array of \( n \) elements, can we find a smallest sub-array size so that the sum of the sub-array is greater than or equal to a certain constant \( S \) in \( O(n) \)?
  - e.g. for \( A = \{1, 5, 1, 3, [5, 10], 7, 4, 9, 2, 8\} \) and \( S = 15 \), the answer is 2 as highlighted

- Given an array of \( n \) elements, find the minimum of each possible sub-arrays with size \( K \) in \( O(n) \)!
  - e.g. for \( A = \{0, 5, 5, 3, 10, 0, 4\}, n = 7, \) and \( K = 3, \) we have 5 possible sub-arrays: \( \{0, 5, 5\}, \{5, 5, 3\}, \{5, 3, 10\}, \{3, 10, 0\}, \) and \( \{10, 0, 4\}. \) The minimum of each sub-array is 0, 3, 3, 0, 0, respectively
Next Week

• **CH2: Data Structures and Libraries**
  – Focus on bit manipulation and Binary Indexed (Fenwick) Tree

• No homework yet, but please try UVa!
skillset.xls Survey

• I want to measure your skillset (NUS students only)
  – But this is an optional task

• Download the file from:
  – [https://sites.google.com/site/stevenhalim/home/material](https://sites.google.com/site/stevenhalim/home/material)
  – See Week01
  – Fill it, rename it to “skillset-yourname.xls”
  – Upload to IVLE Workbin

• To save time, let’s do this at home
  – I will send a reminder 😊
10 Minutes Break

• In the last part of our first introductory class, you will familiarize yourself with Linux controlled environment in this PL2 and the Mooshak system, the internal online judge used for CS3233 this sem
  – Mock contest with 2 “easy” + 1 “medium” problems
  – Not graded yet 😊, enjoy it for fun
    (and to help you decide if you should take CS3233)
Mooshak

• Let’s try this system
  – Open with Firefox (does not work well with most other browsers 😞):
    • http://algorithmics.comp.nus.edu.sg
  – Click “Online Judge (Login)” at the top left corner
  – Try “Mini0 (16 Jan 2013)”
  – Use user ids that have been emailed to you
    • For new participants use dummy IDs where pwd = uid
      – team01, team02, team03, ...
      – I will tell who use which id