DR ANTON GERDELAN

GERDELA@SCSS.TCD.IE

BIG DOWNSTAIRS ADAPT LAB (O'REILLY BUILDING)

# DATA STRUCTURES & ALGORITHMS [CS3D5A]

## IMMEDIATE CONCERNS

- Is everyone here?
- How many are we by division? C C/D D extras?
- Unofficial prerequisites (“it will be easier if…”)
  - passed 1 or 2 programming courses
  - much easier if spent more time on it
  - independently motivated

## IMMEDIATE CONCERNS

- Is this subject super hard?
  - no
  - practise for 7+ hours/week (we have 4hrs prac. in calendar)
- Do I really need this subject to get an engineering/industry job?
  - yes

## TIMETABLE

- 3× lectures / week
- 1× tutorial / week
- 1× lab / week - with Peter Lavin <peter.lavin@scss.tcd.ie>
- lots of practical help with tutorials, labs, discussion board Q&A
- 4 graded assignments - 40%
- 2 hour written exam - 60%

| | | | |
|---|---|---|---|
| Mon | 2-3pm | Salmond Th., Hamilton | lecture |
| Tue | 9-10am | M20, Museum | lecture |
| Wed | 2-3pm | Synge Th., Hamilton | lecture |
| Thu | 3-4pm | M21, Museum | tutorial |
| Fri | 2-5pm | LG12, O'Reilly | lab |

## OUTCOMES

- Beef-up core programming skills
- Lots of practice coding
- Fundamental algorithms and DS in CS
- How to design an algorithm
- Reasoning about data
- Reasoning about costs and complexities
- Hungry for more!

*src: Film "Stay Hungry" 1976.*

## THIS IS A NEW COURSE

- Meet top international standards
- Has to adapt to suit your needs
- Find and address most important knowledge gaps
- Make best use of time
- Try not to be too easy or too hard
- Need lots of feedback during course!

## ALGORITHMS THEORY OVERVIEW

- from *al-Ḵwārizmī* or *arithmos* (number)
- Sorting
- Searching
- Measuring and approximating complexity
- Approach to design
- Interesting problems
- Considering hardware or independent of hardware

## A DEFINITION OF AN ALGORITHM

*from Cormen et. al. Introduction to Algorithms:*

- a well defined **procedure**
- takes **input** value (or set of)
- produces **output** value (or set of)
- tool to solve a **computational problem**
- has practical application
  - sequencing genomes, the Internet, ...
- candidate problems have many possible solutions
  - may not be a perfect solution

Input: **sequence** of *n* numbers
*<a1, a2, ... , an>*

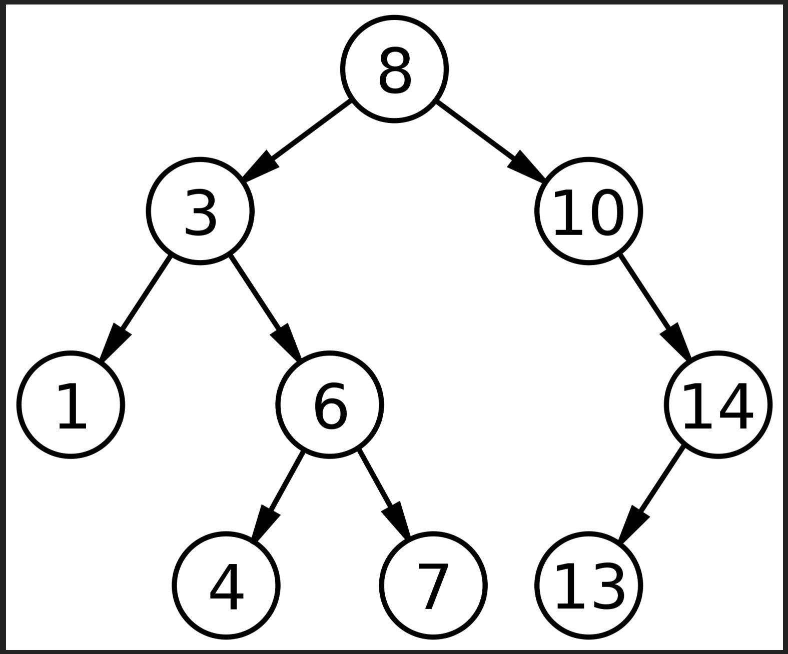
Output: **permutation** (reordering)
*<a'1, a'2, ... , a'n>*
such that
*<a'1 ≤ a'2 ≤ ... ≤ a'n>*

A given input set is called an **instance**, e.g:

*<30, 44, 68, 12, 77>*

## DATA STRUCTURES THEORY

- What is the job of a computer program?
- Designing for convenience / organisation
- Designing for efficiency
- Understanding pros & cons
- Some interesting structures like trees, graphs, hash tables
- Some fun real-world examples ;-)



*src: Wikipedia "Binary Search Tree"*

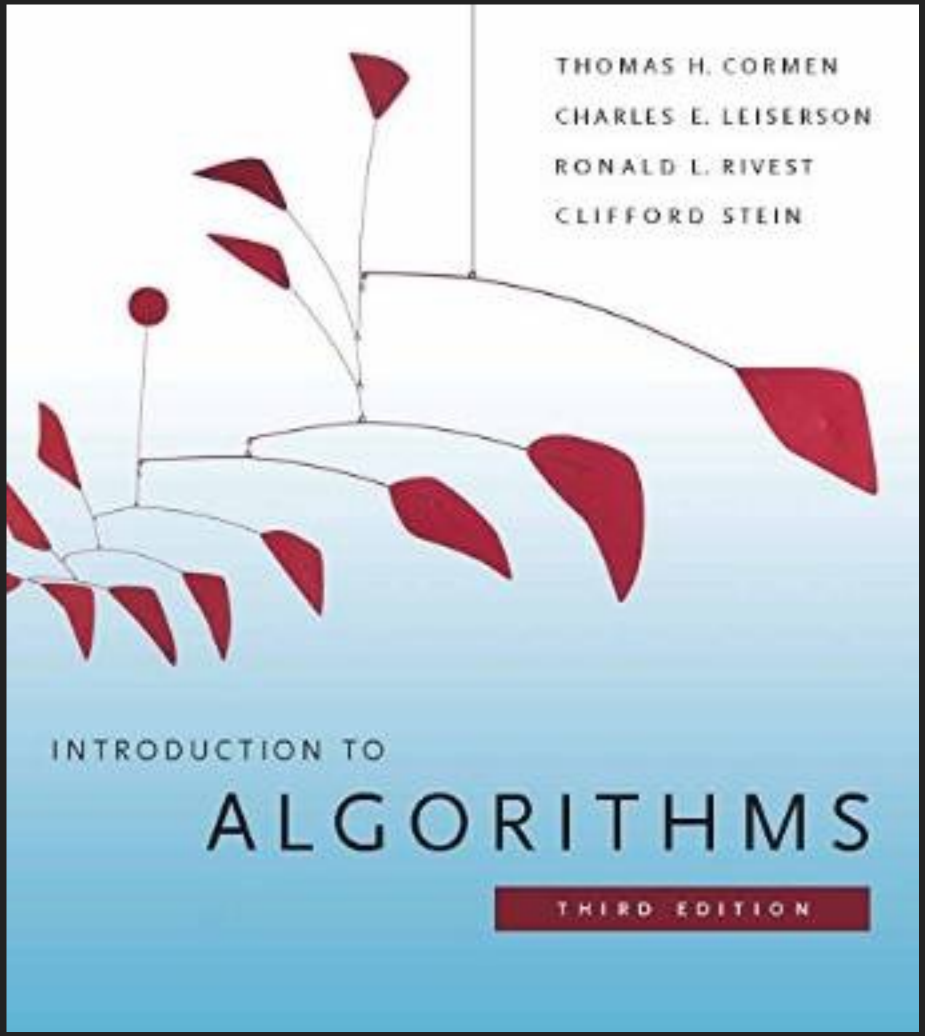
*src: [bbc.com](bbc.com)*
*"New Doom level released by game creator John Romero"*

## PRACTICAL WORK

- C programming
  - simple C++ is fine too
- We can work through tutorials
  - some unusual/new concepts
  - tips and how-to-code-it advice
- Maintain a folder of concepts - I keep mine on GitHub
- A few challenging assignments
- Exam

## THEORY REFERENCES

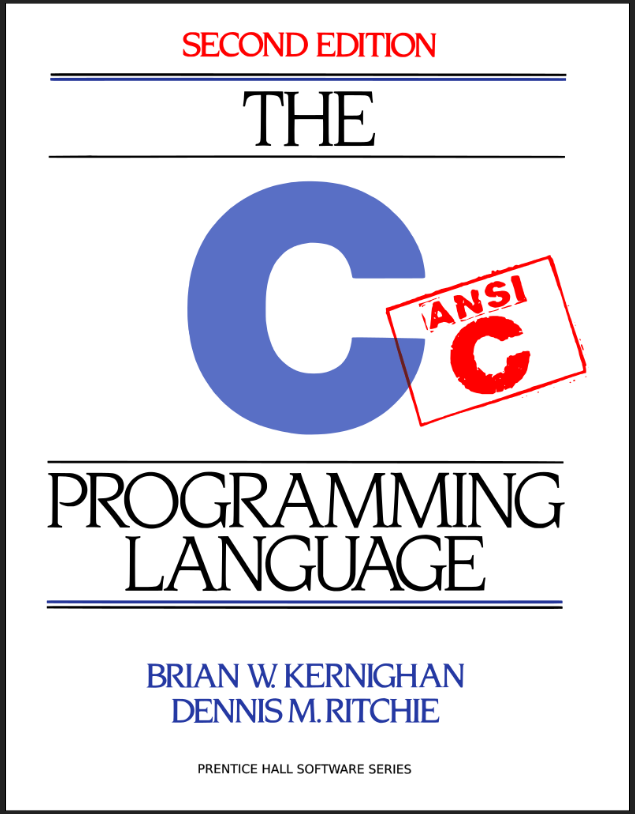
- Books not required
  - Fundamental subject - worth having a desk reference at some point in career
  - Robert Sedgewick's books are great (clear and practical).
- Some great websites
  - **(list in last slides)**
- Read code on e.g. GitHub - look for famous people or projects, games, tools you like.



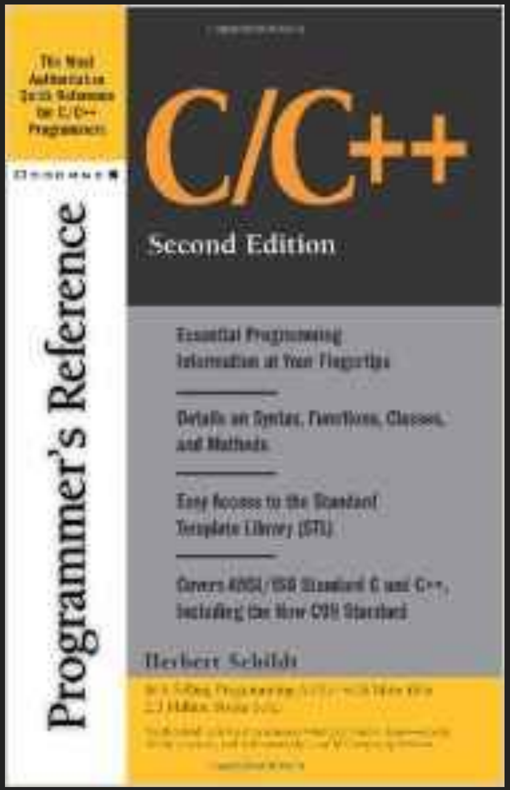
top unis prescribe this book at the moment

## PRACTICAL REFERENCES

- **Ritchie and Kernighan** (Indian market printing is much cheaper - green cover)
- A C pocket reference can be nice
  - O'Reilly also has searching and sorting implementations
- Websites with the same format and content as pocket reference books:
  - [cprogramming.com](cprogramming.com)
  - [cplusplus.com](cplusplus.com)



SECOND EDITION
THE
C
ANSI C
PROGRAMMING LANGUAGE
BRIAN W. KERNIGHAN
DENNIS M. RITCHIE
PRENTICE HALL SOFTWARE SERIES



The Most Authoritative Quick Reference for C/C++ Programmers
Programmer's Reference
C/C++
Second Edition
Essential Programming Information at Your Fingertips
Details on Syntax, Functions, Classes, and Methods
Easy Access to the Standard Template Library (STL)
Covers ANSI/ISO Standard C and C++, Including the New C99 Standard
Herbert Schildt

## SELF ASSESSMENT: 0-5 (NONE-EXPERT)

- General programming experience
- C (or C++) experience
- Programming mileage (years/hours per day)
- Mathematics (e.g. proofs by induction)
- Particular weak points or unknowns?
- Strong points?
- Ideal career/position - dream jobs?

## FIRST WEEK

- Warm-up lab with me - write an image file
- Might sound scary/hard/easy
- Fun (hopefully)
- Sort of test to see where you are
- Refresh programming skills
- If all too hard - good - we will work through it together! **Yell if stuck.**
- If too easy - also good!

*unfamiliar term? -> tell me*

*struggling with getting started -> tell me*

*too much work in other courses -> tell me*

*got lost earlier in lectures and struggling to keep up -> tell me*

*i'm secretly terrible at programming -> tell me*

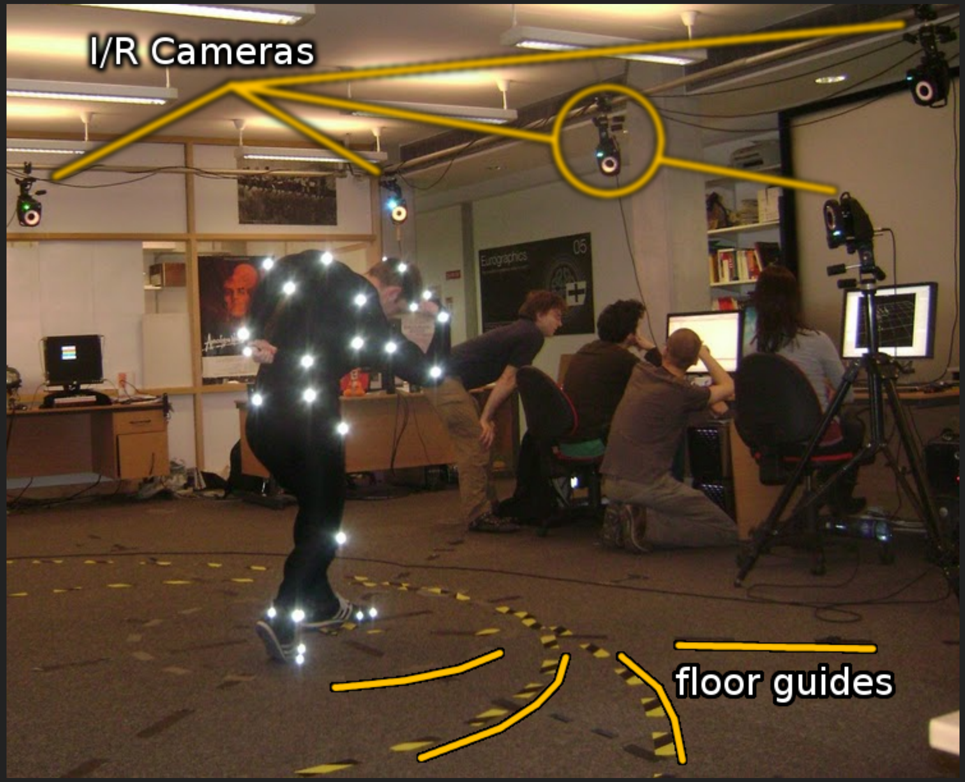
*it's all too basic and i'm bored -> tell me*

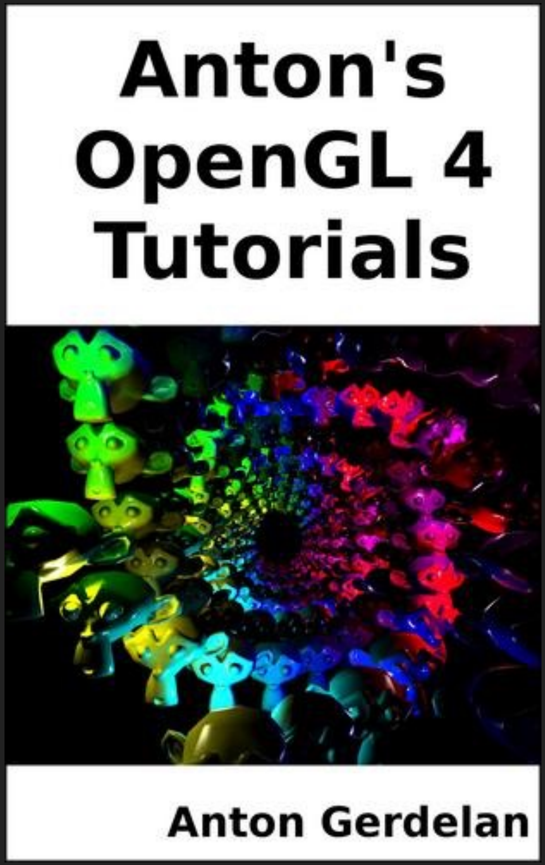
## OTHER WEEKS

- Well studied data structures, algorithms, and problems.
- Course will adapt to suit pace and needs
- 3-4 guest lecture topics with Mike Brady
- 4 graded assignments (about 2 weeks and 10% each).
  most likely topics - implementing and analysing:
  {linked lists and trees, sorting, heuristic search, hash tables}
- Self tests, quizzes, model problems etc. in tutorials and labs.
- Times/deadlines/dates may need to shuffle around to suit.
- Shouldn't be stressful or overloaded.

## ABOUT ME ~ ACADEMIC

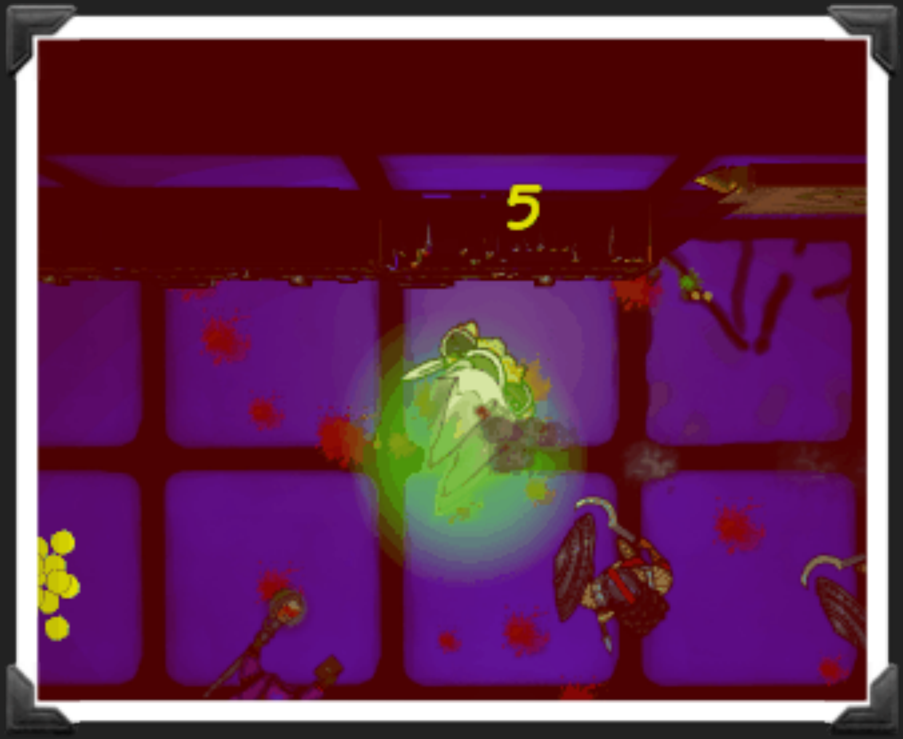
- Post-doc in ADAPT Centre (O'Reilly building)
  - graphics, VR, visualisation, lecturing graphics
- Lecturer (*universitetsadjunkt*) at BTH in Sweden
  - 3d programming I, II, modular software blah blah blah, algorithms & data structures, programming courses etc.
- PhD (Massey Uni. NZ)
  - AI topics, motion control, fuzzy logic, genetic algorithms
  - GV2





## MORE IMPORTANTLY

- 14 yrs programming ~ 14k hrs
  8~12hr/day last few years
- traditional 8pm - 4am coder
- mostly C, C++, shaders
- published a game on Steam
- currently working on a little graphics terrain/water demo/game
- can answer most programming questions



## PROSPECTIVE TOPICS (MMAS TERM: 26 SEPT–16 DEC)

| week | topic |
|---|---|
| 1 | intro, refresher |
| 2 | elementary data structures and algorithms |
| 3 | guest lectures |
| 4 | sorting |
| 5 | searching |
| 6 | searching & hash tables |
| 7 | ~ *reading week* ~ |
| 8 | computational complexity |
| 9 | trees and graphs |
| 10 | advanced topics/case studies |
| 11 | data complexity |
| 12 | revision |

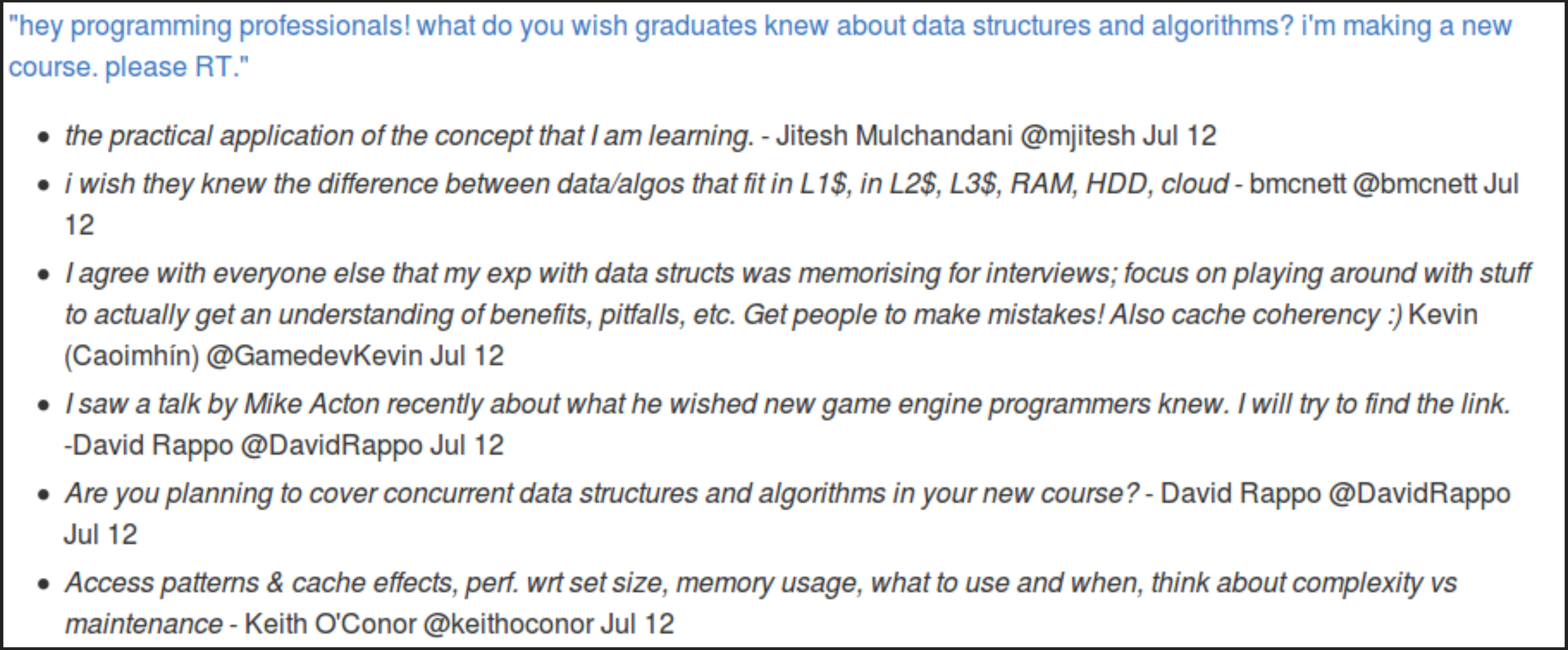
## AND ONE MORE THING

- notify me about mistakes before I pass course on!
    - bug report bounty?
- work individually
    - dividing work means missing important skills
    - but helping others is ++ for your own understanding
- try not to over-engineer your code
    - only answer the specific problem

## SOME NEAT WEBSITES (MORE LATER)

- Amit Patel's website has lots of illustrated/animated algorithms http://www.redblobgames.com/
- and David Galles (USFCA) - https://www.cs.usfca.edu/%7Egalles/visualization/Algorithms.html
- Keith O'Conor's slides from TCD talks http://www.fragmentbuffer.com/publications/

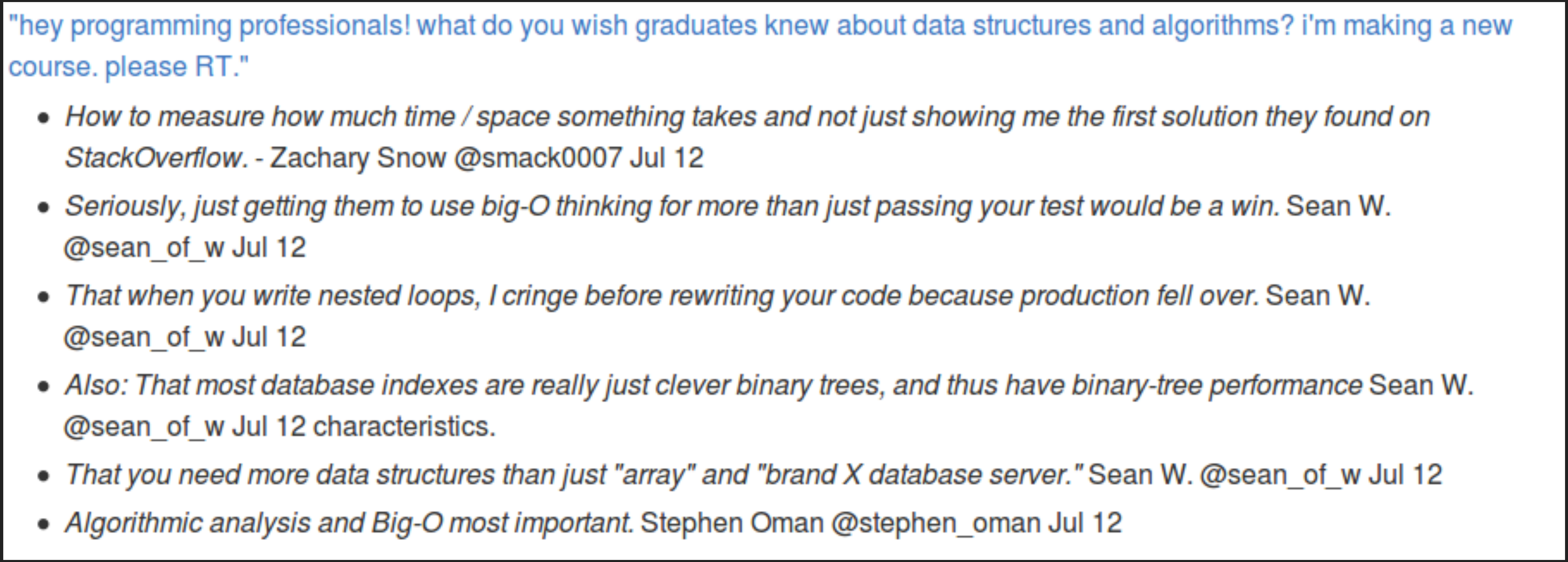
## THE PLAN



"hey programming professionals! what do you wish graduates knew about data structures and algorithms? i'm making a new course. please RT."

- *the practical application of the concept that I am learning.* - Jitesh Mulchandani @mjitesh Jul 12
- *i wish they knew the difference between data/algos that fit in L1$, in L2$, L3$, RAM, HDD, cloud* - bmcnett @bmcnett Jul 12
- *I agree with everyone else that my exp with data structs was memorising for interviews; focus on playing around with stuff to actually get an understanding of benefits, pitfalls, etc. Get people to make mistakes! Also cache coherency :)* Kevin (Caoimhín) @GamedevKevin Jul 12
- *I saw a talk by Mike Acton recently about what he wished new game engine programmers knew. I will try to find the link.* -David Rappo @DavidRappo Jul 12
- *Are you planning to cover concurrent data structures and algorithms in your new course?* - David Rappo @DavidRappo Jul 12
- *Access patterns & cache effects, perf. wrt set size, memory usage, what to use and when, think about complexity vs maintenance* - Keith O'Conor @keithoconor Jul 12

## THE PLAN



"hey programming professionals! what do you wish graduates knew about data structures and algorithms? i'm making a new course. please RT."

- *How to measure how much time / space something takes and not just showing me the first solution they found on StackOverflow.* - Zachary Snow @smack0007 Jul 12
- *Seriously, just getting them to use big-O thinking for more than just passing your test would be a win.* Sean W. @sean_of_w Jul 12
- *That when you write nested loops, I cringe before rewriting your code because production fell over.* Sean W. @sean_of_w Jul 12
- *Also: That most database indexes are really just clever binary trees, and thus have binary-tree performance* Sean W. @sean_of_w Jul 12 characteristics.
- *That you need more data structures than just "array" and "brand X database server."* Sean W. @sean_of_w Jul 12
- *Algorithmic analysis and Big-O most important.* Stephen Oman @stephen_oman Jul 12

## QUESTIONS?
## CONCERNS?
## TIMETABLE CLASHES?
## WILD INTERJECTIONS?