Adam Blank

PERSONAL INFORMATION

E-mail: blank@caltech.edu
Web: https://www.countablethoughts.com
Address: 1200 E California Blvd
MC 305-16
Pasadena, CA 91125-0001

Phone: (626) 395-1765

EMPLOYMENT

Teaching Assistant Professor *2020 –*
Computing and Mathematical Sciences, California Institute of Technology, Pasadena CA

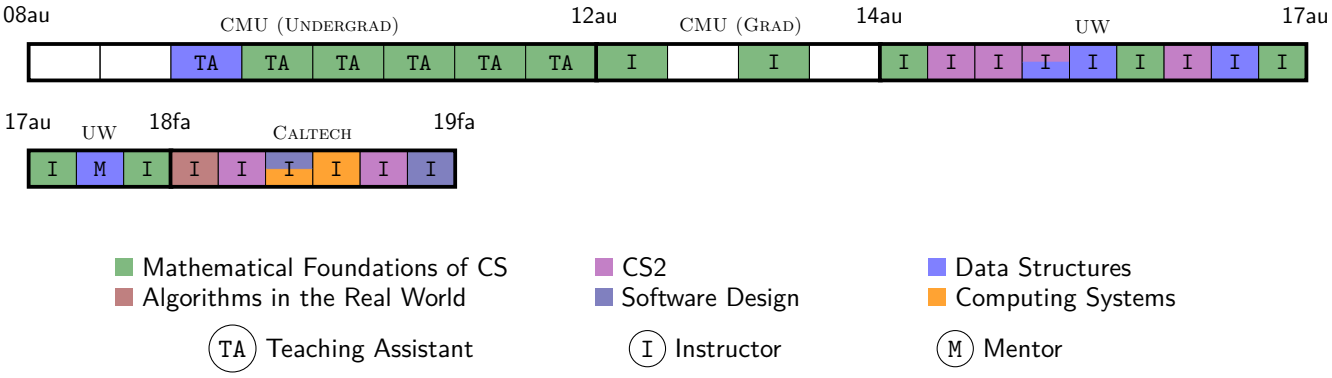
Lecturer *2018 – 2020*
Computing and Mathematical Sciences, California Institute of Technology, Pasadena CA

Lecturer *2014 – 2018*
Computer Science and Engineering, University of Washington, Seattle WA

EDUCATION

Carnegie Mellon University *2012 – 2014*
Master of Science in Computer Science
Thesis: *Technological and Pedagogical Innovations for Teaching Introductory Discrete Mathematics to Computer Science Students*
Advisors: *Randy Bryant, Klaus Sutner*

Carnegie Mellon University *2008 – 2012*
Bachelor of Science in Computer Science, Minor in Mathematics



HONORS

Caltech ASCIT Teaching Award	*2019*
UW CSE ACM Teaching Award	*2018*
UW CSE ACM Teaching Award	*2017*
Alan J. Perlis Undergraduate Student Teaching Award	*2012*
Honorable Mention for CRA Outstanding Undergraduate Award	*2012*

PROFESSIONAL SERVICE

CCSC:SW Program Committee Member	*2019*
SIGCSE Organizing Committee Member	*2018, 2019, 2020*

Service

Caltech Freshman Advisor	*2019 –*
CMS Computing Resources Working Group	*2019 –*
CMS Teaching Assistant Professor Hiring Committee	*2019 – 2020*
Member of Student/Faculty Conference Committee for CS	*2018 – 2019*
Co-designing and Leading Upper Division TA Training (UW CSE)	*2016 – 2018*
Member of UW Evidence-based Teaching and Learning Community	*2015 – 2018*
UW CSE ACM Student Chapter Advisor	*2016 – 2018*
UW CSE Scholarship Committee	*2015 – 2018*
CMU SCS CSD Teaching Assistant Advisory Committee	*2012 – 2014*

Mentees (Teaching)

Melissa Hovik (2015–2020). (first job: Teaching Assistant Professor at Caltech)
Nicole Riley (2015–)
Michael Lee (2015–2018). (first job: Dropbox)
Evan McCarty (2015–2018). (first job: Lecturer at UIC)
Riley Porter (2015–2016). (first job: Lecturer at University of Washington)

Teaching (Caltech)

CS 37: Algorithms in the Real World
Instructor (*Autumn 2018*)
30 students
This course introduces algorithms in the context of their usage in the real world. The course covers compression, advanced data structures, numerical algorithms, cryptography, computer algebra, and parallelism. The goal of the course is for students to see how to use theoretical algorithms in real-world contexts, focusing both on correctness and the nitty-gritty details and optimizations.

CS 2: Introduction to Programming Methods
Instructor (*Winter 2019*)
$140 - 150$ students
This course is "CS2" in Java at Caltech. In Winter 2019, I rebooted this course to focus more on *data structures*. It introduces students to ADTs, and we implement a variety of data structures like Binary Trees, Hash Tables, and Graphs.

CS 3: Introduction to Software Design
Instructor (*Spring 2019*)
55 students
This course is a practical introduction to designing large programs in a low-level language. Heavy emphasis is placed on documentation, testing, and software architecture. Students will work in teams in two 5-week long projects. In the first half of the course, teams will focus on testing and extensibility. In the second half of the course, teams will use POSIX APIs, as well as their own code from the first five weeks, to develop a large software deliverable. Software engineering topics covered include code reviews, testing and testability, code readability, API design, refactoring, and documentation. In Spring 2019, I rebooted this course, as it hadn't been taught in several years. The goal is to get students to (1) learn C,(2) write *good* code (rather than just code that "works"), and (3) prepare them for CS 24.

CS 24: Introduction to Computing Systems
Instructor (*Spring 2019*)
$70 - 90$ students
This course is a traditional introduction to computing systems. Various parts of the system are examined and implemented. In Fall 2019, I re-designed this course to focus more on skills all programmers need rather than skills systems programmers need.

Teaching (UW)

CSE 143: Computer Programming II
Instructor (*Spring 2015, Autumn 2015, Autumn 2016*)
Co-instructor (*Winter 2015*)

$160 - 700$ students

This course is "CS2" in Java at UW. It introduces students to ADTs, and we implement simple data structures like Linked Lists and Binary Trees. The students are *very varied*. In Autumn 2015, I taught "CSE 143X" which combines CS1 and CS2 into a single 10 week course.

CSE 390H: CSE 143 Honors Seminar
Instructor (*Autumn 2014*, *Winter 2015*, *Spring 2015*, *Autumn 2015*)
50 students

This is a seminar accompanying CSE 143. The only stated course goal is to get students interested in Computer Science. We discussed a wide variety of topics such as steganography, combinatorial games, esoteric programming languages, and cryptograms.

CSE 311: Foundations for Computing I
Instructor (*Spring 2016*, *Spring 2017*)
Co-instructor (*Autumn 2014*)
$100 - 140$ students, all new majors

UW CSE Students take this course in their first quarter in the major. CSE 311 combines many of the traditional introductory discrete math topics (e.g., logic, proofs, number theory, sets, relations, various types of induction) with some introductory computability material (e.g., regexps, CFGs, DFAs, NFAs, irregularity, uncomputability). I developed several online homework "autograders" for this course which I have helped other instructors integrate into their offerrings.

CSE 332: Data Structures & Parallelism
Instructor (*Autumn 2015*, *Winter 2016*, *Winter 2017*)
$120 - 280$ students

UW CSE Students take this course in their second quarter in the major. When I first taught this course, the projects had not been changed for six years; so, I wrote new projects in which students implement all the "back-end" data structures that make a "real world" client work. This course is the first time students usually have projects that last several weeks; so, I introduced "checkpoints" in which students meet with someone on course staff in an open area to dicuss aspects of the projects. To make things run smoothly, I built some tools which integrate with `gitlab` continuous integration to help monitor students for these checkpoints. Although I didn't teach this course in Spring 2016 or Autumn 2016, both instructors adopted my projects. In Autumn 2016, the instructor also adopted my course policies and online exercises.

Teaching (CMU)

15-151: Mathematical Foundations of Computer Science
Course Designer and Instructor (*Fall 2012*, *Fall 2013*)
$100 - 140$ students, all freshmen

All computer science majors at CMU take two introductory discrete math courses: one in their first semester and another in their second. The course taken in the spring is "15-251" (see below), and it is traditionally a very difficult course. I created 15-151 for CS majors to make 15-251 less overwhelming. I used programming analogies and various techniques like group work, quizzes worth no points to check their understanding, and mechanisms/techniques to increase learning. In 2011 and 2012, I taught three lectures of 15-151 on Monday, Wednesday, and Friday. In 2011, my mentor, Klaus Sutner, gave half of the lectures.

21-127: Concepts of Mathematics
Teaching Assistant (*Fall 2011*)
$200 - 300$ (50 in my section) students, primarily freshmen

This course is a generic introduction to discrete mathematics for various disciplines. I held recitation for 50 students twice a week, wrote my own handouts and quizzes, wrote several of the exams, graded for my entire recitation, and held 3-4 office hours every week.

15-131/98-172: Great Practical Ideas for Computer Scientists
Course Designer and Instructor (*Fall 2011*)
Co-instructor (*Fall 2012*)
100 students, all freshmen

A while back, CMU changed their introductory sequence to include additional content on verification and parallelism. After this change, students needed additional background on UNIX, bash scripting, and C debugging; so, I created 98-172, a student-taught course to fill this need. I gave lecture once a week, and held four "lab-itations" (a cross between recitation and lab) each week as well. In Fall 2013, CMU decided to make 98-172 a course in the CS department, and I helped the effort to make the necessary changes.

15-251: Great Theoretical Ideas in Computer Science
Head Teaching Assistant (*Spring 2012*)
Teaching Assistant (*Fall 2010*, *Spring 2011*, *Spring 2010*)
$150 - 200$ (40 in my section) students, primarily freshmen

This course is a very difficult introduction to discrete mathematics and computability theory. It covers a new topic every week, and the homework questions are traditionally time consuming. Every semester I TAed, I held a recitation for around 40 students, held at least 3-4 office hours every week, developed homework assignments and recitations, and graded. I also maintained the course submission and grading system and over many of the semesters, I developed a new one. In Spring 2012, I had several additional duties as the head teaching assistant. I acted as liaison between the other TAs and professors, and I held extra "conceptual" office hours every week in which I repeated the topics from lectures for students who were struggling. These office hours usually were attended by 20-30 students.

EdTech Projects

As a companion to my teaching, my research consists of educational technology which I often test with user studies and other formal evaluations. Many undergraduate students have contributed time and energy to several of these projects.

Java Data Structure Visualizer *2016 – 2018*
I developed two Eclipse plug-ins: EZClipse and Viz. EZClipse makes the eclipse user interface more friendly to beginners. Viz provides a "logical visualization" of various features of Java programs during debugging. In particular, it helps students visualize data structures, stack traces, and fields.

Induction Practice Interface *2015 – 2018*
Some students and I developed an interface (and checker) for simple induction problems. We are working on extending it and using it in an introductory discrete math course for practice and homework problems.

Online HW Problems Framework *2014 – Current*
I have written online interfaces for various problems ("construct a DFA", "insert into this AVL tree", etc.) in introductory discrete math and data structures courses to (1) encourage students to make multiple attempts on a problem, and (2) speed up grading. These interfaces give minimal (but useful) feedback, and the students are given a fixed number of attempts for each problem. I am working on expanding and improving as many of these interfaces as possible for various courses. They are being used by multiple instructors in both CSE 311 and CSE 332 at UW.

A compiler for introductory discrete mathematics *2012 – 2014*
I designed a programming language called "Setty" and implemented a compiler. The standard library builds naturals, pairs, etc. from sets using the standard mathematical constructions. The goal of this language (which we used in 15-151) was to allow students to learn "the mathematical language" such as set comprehensions like $\{x \in [5] \mid x > 2\}$ and quantified statements like $\forall(x \in [5]).\ x < 5$ by evaluating them in a computational environment.

Improving submission, annotation, review, and feedback of proofs *2010 – 2015*

I developed a course infrastructure system that handles LaTeX submissions, annotation, grading, and feedback of proofs. The system allows graders to first review and mark up the student submissions while re-using comments without worrying about point values, and then apply a rubric to their reviews. By providing an interface to easily reuse comments and update the rubric globally with a single click, we reduce the grading time while increasing the usefulness of feedback.

Peer grading of proofs *2010 – 2014*

The system we developed for submission and feedback also allowed students to anonymously review each other's submissions. I ran a study which showed that students who *reviewed* more proofs rather than *doing* more proofs wrote better proofs later.

References

Mark Stehlik
Teaching Professor
Department of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
☎ (412) 268-6273
✉ mjs@cs.cmu.edu

Ruth Anderson
Senior Lecturer
Paul G. Allen School of
Computer Science & Engineering
University of Washington
Box 352350
Seattle, WA 98195-2350
☎ (206) 616-1742
✉ rea@cs.washington.edu

Adam Wierman
Professor of Computing and Mathematical Sciences
Computing and Mathematical Sciences
California Institute of Technology
1200 E. California Boulevard
MC 305-16
Pasadena, CA 91125
☎ (626) 395-6569
✉ adamw@caltech.edu

Dan Grossman
Professor and Vice Director
Paul G. Allen School of
Computer Science & Engineering
University of Washington
Box 352355
Seattle, WA 98195-2350
☎ (206) 616-1124
✉ djg@cs.washington.edu

Donnie Pinkston
Lecturer in Computing and Mathematical Sciences
and Schmidt Software Academy Instructor
Computing and Mathematical Sciences
California Institute of Technology
1200 E. California Boulevard
MC 305-16
Pasadena, CA 91125
☎ (626) 395-6736
✉ donnie@cms.caltech.edu