ECE 551D and 751D
Introduction to Programming, Data Structures, and Algorithms in C++

Introduction

Course Overview

Programming Ability Now

Admin

- Professor 1: Andrew ("Drew") Hilton
  - E-mail: adhilton@ee.duke.edu
  - Office Hours: Zoom, Thursdays 8:00-9:00 pm
    - Or by appointment (e-mail me, we'll set up a time)
- Professor 2: Genevieve Lipp
  - E-mail: genevieve.lipp@duke.edu
  - Office Hours: Zoom, time pending poll results
    - Or by appointment (e-mail me, we'll set up a time)
- TAs : Mutian Wang, Cecilé Sadler, Yang Deng, Yang Zhong, Daniel Park
  - Office hours to be posted on Sakai
- Access OH through Sakai —> Zoom Meetings

A bit about us

A bit about us

- Please, feel free to call me "Drew"
  - Actually, I **strongly** prefer that to "Professor Hilton"
- Please, feel free to call me "Genevieve"
- Who knows what Drew's job is?
  - To teach you to be industry-ready programmers
  - Everything else I do is secondary to that -> Research, reviewing papers, advising undergrads, sleeping…
- Who knows what Genevieve's job is?
  - ECE/MEMS
  - Undergraduate dynamics, Mech E programming, and… you all
- If you ask, one of us **will help you**
  - Office hours: for you
  - OH not enough? Set up 1:1 meeting
  - Chronically struggling? We can have a weekly 1:1 meeting

Learning your names

- We'd like to learn your names...
  - There are ~55 of you, but we can do it!
  - Please help us learn your names:
  - Make your Zoom name the one you want to be called by
  - Don't hesitate to correct our pronunciation of your name until we get it right
  - Don't hesitate to let us know your correct pronouns

Topic Overview 551

Professional Tools	Linux, Emacs, valgrind, make, git, gcc
Programming (in C)	Reading Code, Algorithms -> Code, Strings, Pointers, Big O, Arrays, Recursion, Dynamic Allocation
C++	Classes, References, Templates, Inheritance, Polymorphism, Dynamic Dispatch
Data Structures	Stacks, Queues, Linked Lists, BSTs, Maps, Sets, Graphs, ADTs, Hash Tables, Priority Queues, Heaps
Other Topics	Sorting, Object Layout, Concurrency, Multiple/Virtual Inheritance

Topic Overview: 751

Professional Tools	Linux, Emacs, valgrind, make, git, gcc
Programming (in C)	Reading Code, Algorithms -> Code, Strings, Big O, Pointers, Arrays, Recursion, Dynamic Allocation
C++	Classes, Templates (Variadic), Lambda, Inheritance, References, Dynamic Dispatch, Polymorphism, Move Semantics, Smart Ptr
Data Structures	Stacks, Queues, Linked Lists, Balanced BSTs, Maps, Sets, ADTs, Hash Tables, Graphs (much more), Heaps, Priority Queues
Other Topics	Sorting, Object Layout, Concurrency (much more), Multiple/Virtual Inheritance

You know this stuff

Larger Project: Propose it yourself!

Professional Tools

- Why Emacs, Linux,…
  - "I want to use…"
  - NO(*): you need to be ready for professional world!

- Former students report VERY IMPORTANT to know
  - In interviews -> conveys message of "serious programmer"
  - I know many ACE programmers. All of them use Emacs or vim
  - On the job -> it's what they use

Emacs is awesome! https://adhilton.pratt.duke.edu/emacs-customization

(*) vim is acceptable, but we won't help you with it

From Former Students

1 I was really mad at command line at first. But when I begin to look for jobs, these are the tools currently used by real companies. ... .These skills learnt on your course open lots of doors for me. Plus the programming experience and problem solving skills, getting an offer becomes fairly easy.

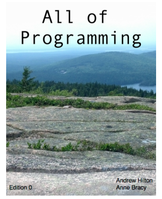
2 Hi Drew. I got started working at Oracle these days. I had a chat with my team colleague and he told me he used emacs to write C…**I have to say thank you that you "forced" us to form a good habit of using emacs** which turns out to be quite useful :)

Programming: MUCH TO LEARN

- Not going to lie: you have a lot of hard work to do
  - Don't think this will be easy.
- Become competent in a thing? 10,000 hours of work
  - Get cracking!
- How could you hope to succeed?
  - Carefully designed pedagogy: teach you strong fundamentals
  - Requires SIGNIFICANT hard work on your part!
    - We'd suggest about **20** hours/week
  - Read and understand
  - Work on assignments, master concepts

How Will You Learn This Semester

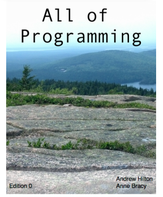
Before Class



Read Chapter [first time]

??? ???

Look at practice problems at end of chapter



Re-read Chapter **Deeply**

??? ???

Attempt Reading Quiz

Questions,
Topics you aren't clear on

....

Survey for class content

How Will You Learn This Semester

Plan for Class

8:30 AM Mini lecture on …
8:45 AM Discussion of assignment …
9:00 AM Individual Help
9:45 AM Mini lecture on …

During Class

Participate in mini-lectures relevant to you
- **Struggled with that topic? Attend it**
  and ask questions!
- **You are good on that topic?**
  Work on assignments

Go to assignment discussions relevant to you
Need individual help?
Make use of time in breakout rooms with professors and TAs

.... Survey for class content
.... (~5 minutes)

How Will You Learn This Semester

Plan for Class

8:30 AM Mini lecture on …
8:45 AM Discussion of assignment …
9:00 AM Individual Help
9:45 AM Mini lecture on …

Best time to work on programming assignments is…

IN CLASS

Help is readily available from professors and TAs
Don't stay stuck!

During Class

Participate in mini-lectures relevant to you
- **Struggled with that topic? Attend it**
  and ask questions!
- **You are good on that topic?**
  Work on assignments

Go to assignment discussions relevant to you
Need individual help?
Make use of time in breakout rooms with professors and TAs

.... Survey for class content
.... (~5 minutes)

How Will You Learn This Semester

- Requires **preparation**
  - Done with reading
  - Have absorbed information in chapter
  - Have thought about questions
  - Have **asked** questions
- Best if you have started assignment before class
  - Read README
  - Begun working on it
  - Have good questions about it and/or clear problems

Best time to work on programming assignments is…

IN CLASS

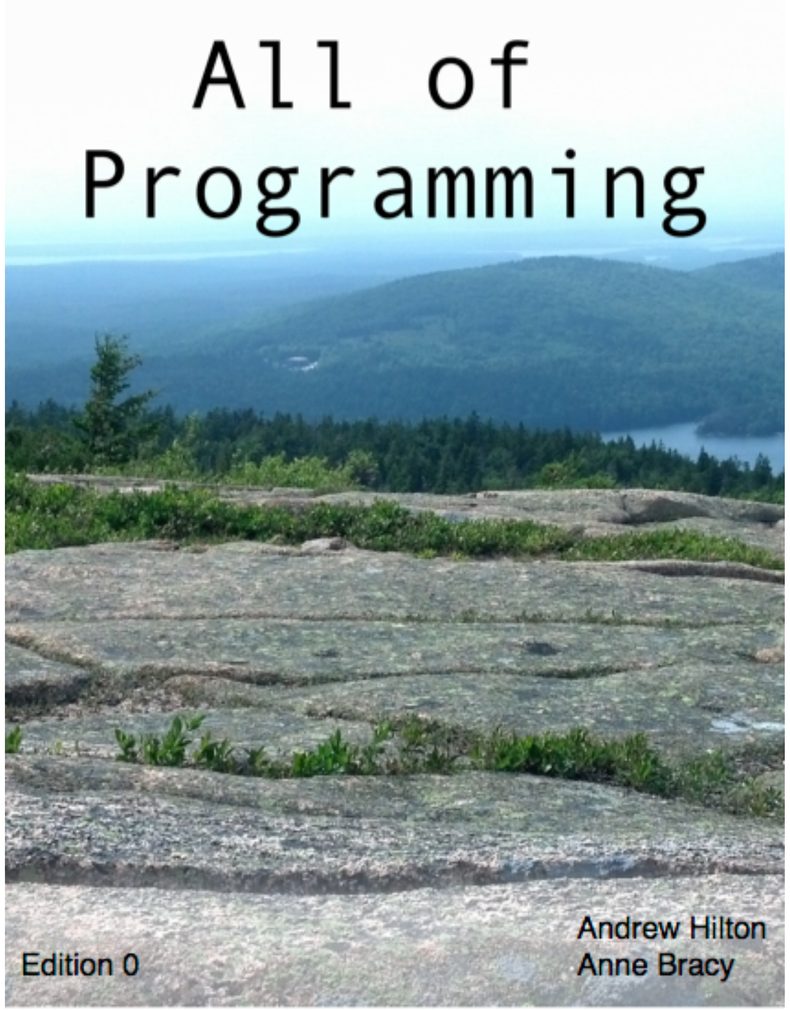
Help is readily available from professors and TAs
Don't stay stuck!

All of Programming

- Textbook: All of Programming
  - Hilton and Bracy, 2015
  - Edition 0

  - http://aop.cs.cornell.edu/

  - Reading schedule on Sakai

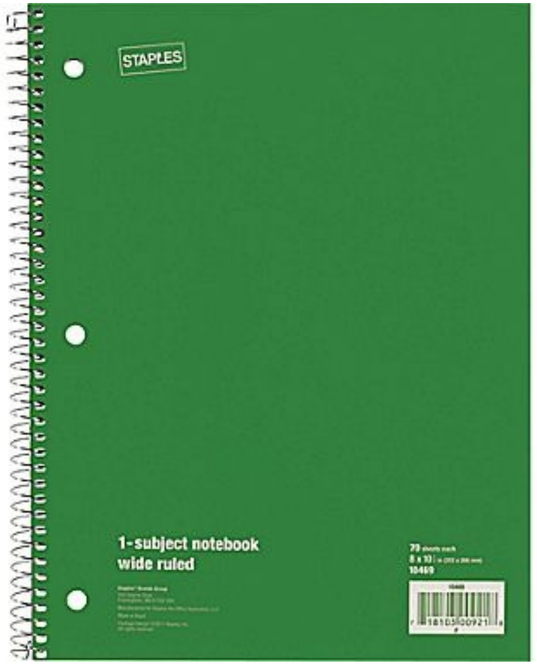


Read the Book, No Really

- Flipped Classroom
  - Lectures are only what you ask for. Not just “sit there and space out”
  - Read your book!
    - Anne and Drew already wrote down everything you need to know…
  - Programming activities in class
    - Help available: don't stay stuck!
- Book: custom crafted for this course!
  - Everything you need to know
  - Nothing you don't.
- Can you understand it all first time through?
  - Maybe not, but…
  - Try to understand as much as you can
  - Work back through later if you want deeper mastery…

Reading, Taking Notes

- Keep notebook
  - Can use on exams
  - Everything must be **handwritten by you**
- Suggestions
  - Reading notes
  - Work on each classwork problem
  - Library reference
  - Tool reference



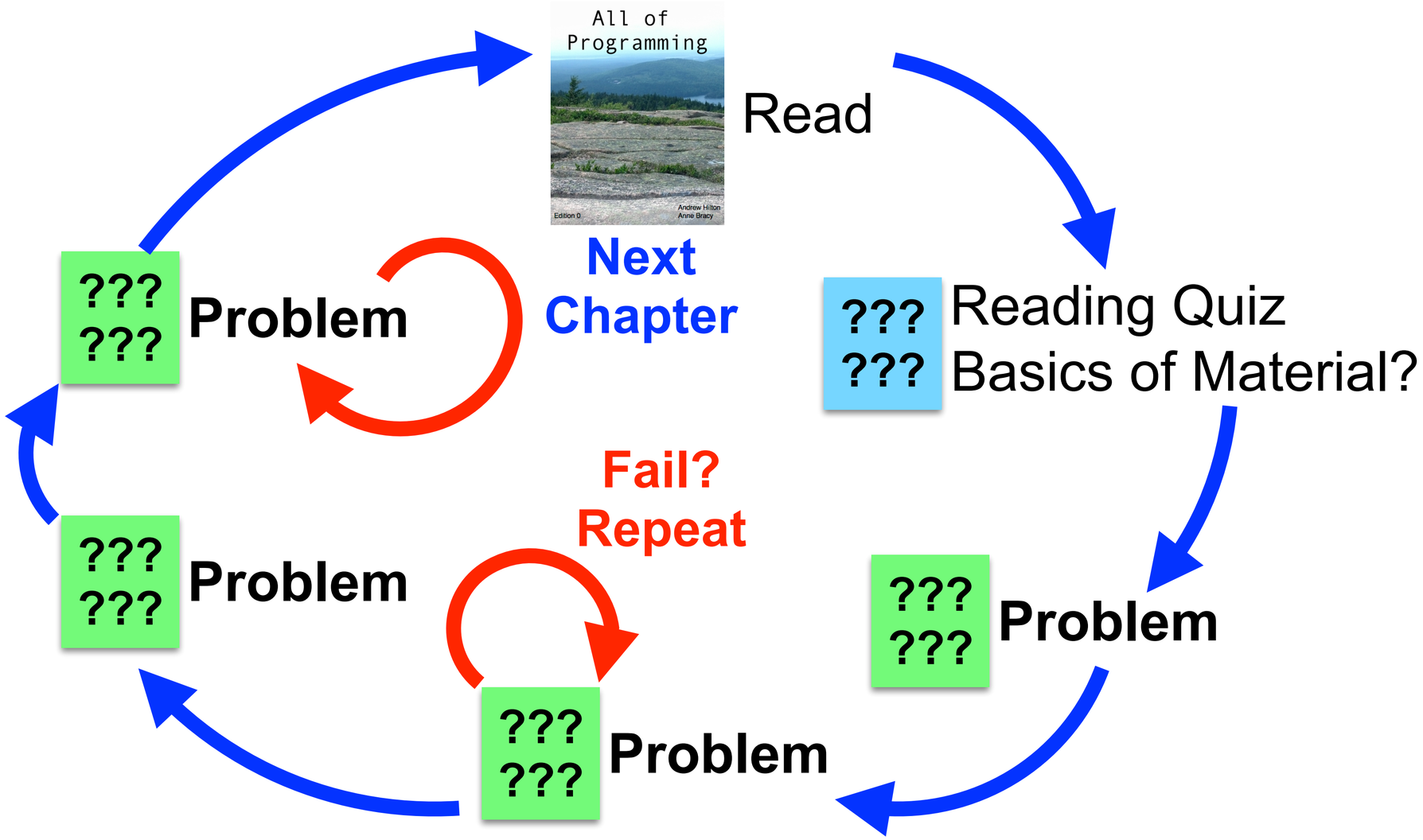
Do note taking seminar series from Minerva: get +1 pt on final exam

Image from staples.com

Read, Ask, Re-read, Understand

- Read your book
  - No seriously
  - I know, you don't for a lot of classes…
- But there is NO lecture in this class
  - You can't just magically learn to program!
- But, it’s so much work!
  - Yeah, we told you this wasn't going to be easy…

Flipped Class, Mastery Learning

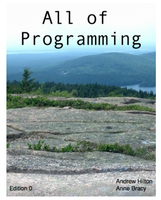


Programming: MUCH TO LEARN

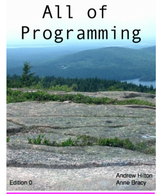
- Not going to lie: you have a lot of hard work to do
  - Don't think this will be easy.
- Become competent in a thing? 10,000 hours of work
  - Get cracking!
- How could you hope to succeed?
  - Carefully designed pedagogy: teach you strong fundamentals
  - Requires SIGNIFICANT hard work on your part!
    - We'd suggest about **20** hours/week
  - Read and understand
  - Work on assignments, master concepts

Wait really? How am I supposed to spend that much time on one class???

20 Hours per Week???



Read Chapter [first time]
(1 hour)



Re-read Chapter **Deeply**
(2 hours)

??? ???
End of chapter practice problems
(~2 hours)

??? ???
Attempt Reading Quiz
(30 minutes)

??? ???
Class-time: Mini-lectures, Work on assignments, etc
(75 minutes)

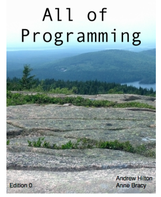
??? ???
Finish problems
(??? minutes) Lets say ~105 min

For Monday (~8.5 hours)

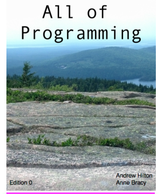
Wed: Repeat (~8.5 hours)

Fri Recitation 75 min

Anti-Strategies (How NOT to save time)



Read Chapter [first time]
(1 hour)



Re-read Chapter **Deeply**
(2 hours)

Skip reading, or just skim the chapter.

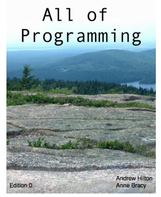
??? ???
End of chapter practice problems
(~2 hours)

??? ???
Attempt Reading Quiz
(30 minutes)

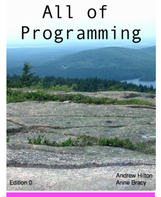
??? ???
Class-time: Mini-lectures, Work on assignments, etc
(75 minutes)

??? ???
Finish problems
(??? minutes) Lets say ~105 min

Anti-Strategies (How NOT to save time)



Read Chapter [first time]
(1 hour)



Re-read Chapter **Deeply**
(2 hours)

??? ???

End of chapter practice problems
(~2 hours)

Don't bother with anything that isn't for a grade!

??? ???

Attempt Reading Quiz
(30 minutes)

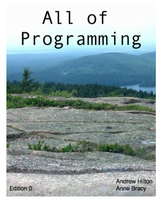
??? ???

Class-time: Mini-lectures, Work on assignments, etc
(75 minutes)

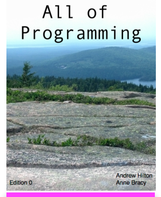
??? ???

Finish problems
(??? minutes) Lets say ~105 min

Anti-Strategies (How NOT to save time)



Read Chapter [first time]
(1 hour)



Re-read Chapter **Deeply**
(2 hours)

??? ???

End of chapter practice problems
(~2 hours)

Don't ask questions in class
Hope that everyone else asks the things you need.

??? ???

Attempt Reading Quiz
(30 minutes)

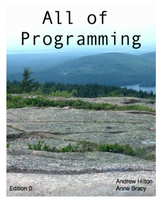
??? ???

Class-time: Mini-lectures, Work on assignments, etc
(75 minutes)

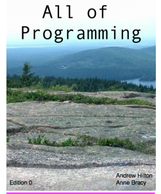
??? ???

Finish problems
(??? minutes) Lets say ~105 min

Anti-Strategies (How NOT to save time)



Read Chapter [first time]
(1 hour)



Re-read Chapter **Deeply**
(2 hours)

??? ???

End of chapter practice problems
(~2 hours)

??? ???

Attempt Reading Quiz
(30 minutes)

??? ???

Class-time: Mini-lectures, Work on assignments, etc
(75 minutes)

Fall behind on the assignments.
After all, how hard can it be to catch up?

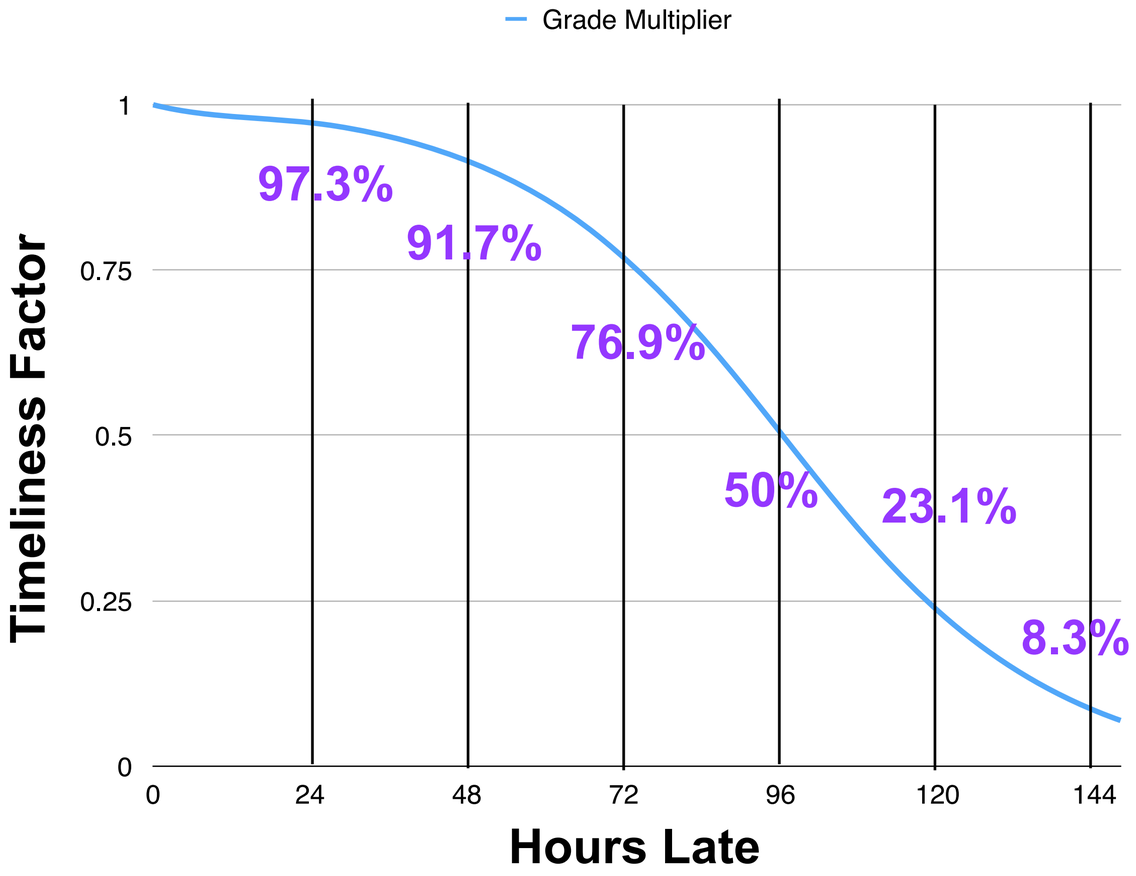
??? ???

Finish problems
(??? minutes) Lets say ~105 min

Formative Assignments

- Most assignments: "formative"
  - For learning!
- Repeat until passed!
  - Can be resubmitted/regraded as much as you want (*)
- Collaboration?
  - Whatever it takes for you to **learn**
    - **You may not download/look at prior student's code**
  - Ask friends for help? Fine, but be sure you **learn**
- Grading/re-grading
  - Run "grade" to grade assignment
    - After git commit/push
  - Tokens: limit rate of regrading
    - Periodic refresh

Formative Assignments: Timeliness



$$\text{Grade} = \%\text{correct} * 1/(1+e^{0.05*(\text{Hours\_Late} - 96)})$$

Evaluative Assignments

- Evaluative Assignments
  - Show me what you know!
- Think of these as out of class programming **exams**
  - Must work on by yourself! No help from classmates
  - Limited outside resources: your notebook, and AoP
  - Submit early/often, only get real grade once after **strict** deadline
- Three of these:

  551:
  Assn 43 Due 9/23
  Assn 60 Due 10/12
  Assn 89 Due 11/16

- Depend on previous formative assignments
  - **Must pass prior assignments to get the assignment**

Academic Integrity

- Academic Integrity Expectations
  - We take academic integrity **VERY** seriously, and you should too
- **Students have been expelled from Duke for cheating**
  - Multiple students expelled for cheating in 551
- More important than your grade: what you learn
  - Interviews? Jobs?
- Think you are "helping your friend out"?
  - Will you be able to do their interview for them?
  - What happens if someone gets A, is clueless in interview?
    - Will you (or others) be able to get interviews?
- Ethics: very important!
  - Industry, as well as academia
- Drew has about a decade of experience in software forensics…

Academic Integrity

- Question?
  - Ask one of us
  - Afraid to ask (maybe they'll say no…): probably bad
- Someone else cheats?
  - Please report it
- Evaluative assignments:
  - Sign honor statement before receiving assignment

Logistics This Week

- Today
  - Read: Appendices
  - Troubleshoot server log in, do assignments 000_submit, 001_app_rq
- Wednesday
  - Read: Chapter 1
  - Do assignment: 002_ch01_rq **before class**
  - We will do 003_algorithm in class, with partners
- Friday
  - Read: Chapter 2 and D.4
  - First recitation
  - Computational thinking activity
- TA office hours begin next week