11-601 Coding & Algorithms Bootcamp

Ralf Brown <ralf@cs.cmu.edu>

TAs:

Ankush Babbar <ababbar@andrew.cmu.edu>

Paramjit Baweja <paramjitbaweja@cmu.edu>

Ajay Mittur <amittur@cs.cmu.edu>

Shaurya Singh <shauryas@andrew.cmu.edu>

Riya Singhal <riyapras@andrew.cmu.edu>

Rajeev Veeraraghavan <rveerara@andrew.cmu.edu>

Chentianye (Glenn) Xu <chentiax@andrew.cmu.edu>

Lecture 01 – August 27, 2024

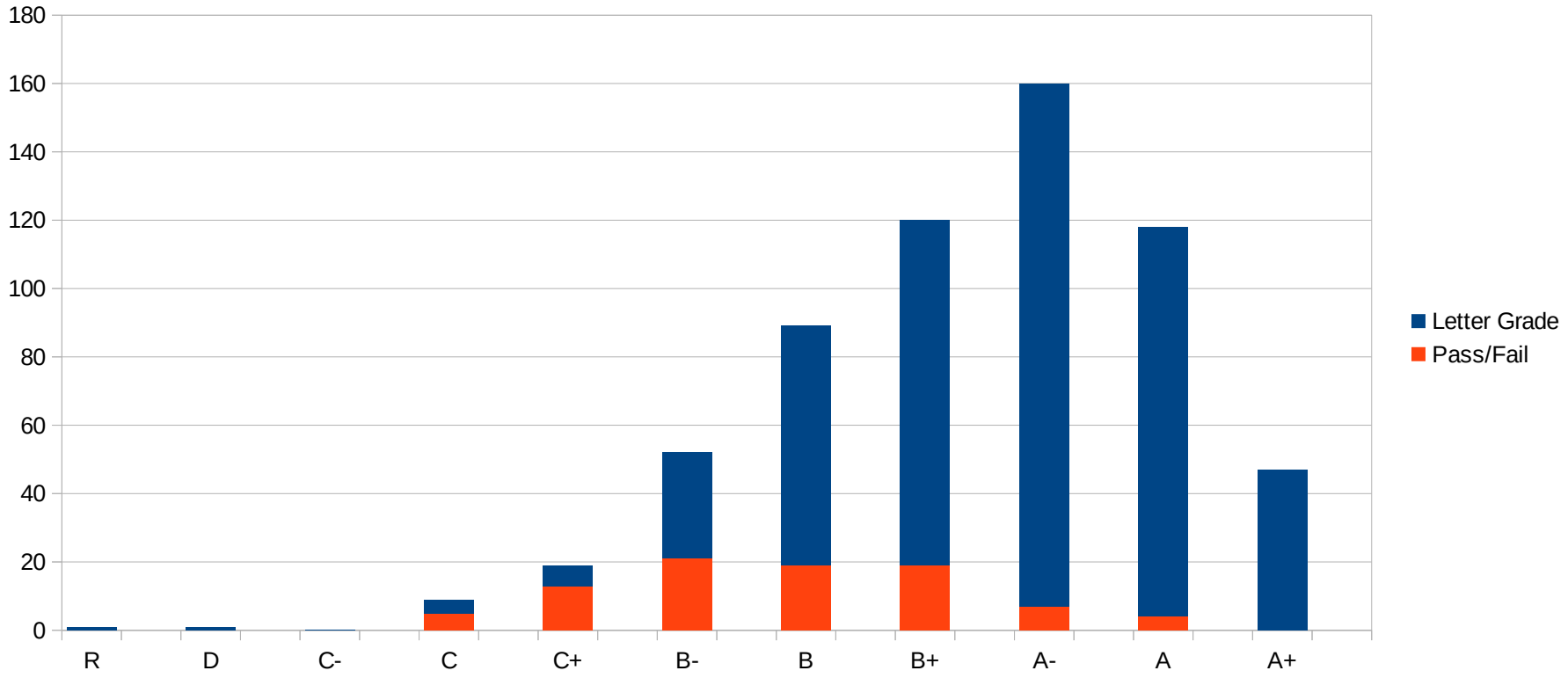
What This Course Is **NOT**

- “Programming from the Ground Up”
  - if you've never taken a Computer Science course, you **will** struggle
- Easy
  - we go through **a lot** of material
    - in three different programming languages
  - you'll be spending a lot of time programming
    - and even more time figuring out **how** to solve the problems before programming their solutions
- A “deep dive”
  - because we cover so many different topics

This Will **Not** Be An Easy “A”

- Things are intentionally challenging
- Expect the class average (before curving) to be around 87-89%
- Grades will be rescaled to have a class average of 87% with standard deviation of 5%; the curve will be regularly updated
  - this means the most common grade will be B+ or A-
  - last year, just about half the students who completed the course got an A- or higher
- If you are taking this course with the expectation of getting an “A”, please consider dropping it now
  - there is **still** a huge waitlist with enrollment raised as much as possible

2017-2023 Grade Distribution



Grading

- 20% per-lecture exercises
  - no make-ups, but your three lowest scores will be dropped
- 30% homework assignments
  - 10% per day penalty for late submission, maximum 5 days late
- 25% mock technical interviews - “The Shuffle”
  - 15% for weekly peer interviews (10% for submitting feedback, 5% from feedback scores)
  - 10% for two instructor/TA interviews
- 25% three exams (9%, 9%, and 7%)

Lecture Exercises vs. Homework Assignments

- "In-Class": Coding under substantial time pressure
  - time limit typically a little more than the median time needed by previous students
  - expect to be *unable* to finish some of the exercises
  - simulates job-interview situations and emergency patching
- Homeworks: Coding under very modest time pressure
  - though if you procrastinate and start the evening the assignment is due, you may find yourself under substantial time pressure

Homework Exercises

- Problems from “Cracking the Coding Interview” will not be graded
  - the solutions are in the book, anyway
- But you should work through them and understand how the solutions work
  - this means actually trying to solve the problem, not just reading the answer!
- ...because those problems will be your questions for the mock technical interviews

“The Shuffle”

more details
next Tuesday

- Mock technical interviews
- Each week, you will give one interview, and be interviewed once
  - interviewees and questions are randomly assigned
  - you are responsible for arranging a mutually-agreeable interview time
  - interviews should last 45-55 minutes
  - you will give each interviewee a score; at the end of the semester, your scores will be normalized
  - you will also receive a feedback score on your role as interviewer
- Over the course of the semester (starting after Exam 1), you will also be interviewed by the instructor and a TA
  - 30 minutes each, in *addition* to that week's peer interviews
  - these are **required**, and count for more than a peer interview

Semester Organization

- First five weeks (until Exam 1): Java
- Then six weeks of Python (until Exam 2)
- and three weeks of JavaScript
- we will look at how the three languages differ from each other in syntax and philosophy, and what they have in common

HackerRank

- An online system for programming-skills tests: www.hackerrank.com
- We will be using it for today's programming exercise; you should have received an email from hackerrank.com this morning
- HackerRank accounts are free for users, but I have only a limited amount of assignment invites so I can't use it the entire semester
  - and it does not integrate with Canvas

Codio

- An online system for programming assignments
  - includes editor and debugging tools
  - for the course staff, includes plagiarism detection and the ability to view and comment on your code
  - integrates with Canvas – you will access assignments by following links from Canvas
- Will be used starting with Exercise 2 (but you don't need to sign up until after the lecture) and Homework 1 (due Tuesday night)
- Requires a paid account at codio.com (about $48)

Q&A

Academic Integrity

- aka “Cheating”
- taken very seriously at CMU – two violations and you're out of the university (and some departments are even stricter)
- in this course, the first offense gets you *at least* a full letter reduction in your final grade
  - plus it gets reported to your department head, who may impose additional penalties
- when an academic integrity violation is found, you can expect **all** of your previous work to be re-examined
  - any previously-undiscovered offenses that come to light may then count as the first violation

Academic Integrity Violations

- copying from another person
- copying from the Internet without attribution
  - including using AI systems such as ChatGPT, Google Bard, GitHub CoPilot
- supplying answers to another student
  - or providing them exam questions before they take the exam
- collaborating with others without attribution
- having someone else take an exam for you
- etc. -- the above is **not an exhaustive list**
  - basically, anything that is **misrepresented** as your own work which isn't, or **helping another** make such a misrepresentation

Note that exact rules can vary from course to course

AIV (1)

- Nancy lets Mark copy her homework. Who is guilty of an academic integrity violation?

AIV (1)

- Nancy lets Mark copy her homework. Who is guilty of an academic integrity violation?
- **BOTH**. Mark for copying, and Nancy for *allowing* the copying.

AIV (2)

- Otto puts a copy of his code in a public Dropbox folder. Paul finds it, and submits a copy as his own solution. Who is guilty of an AIV?

AIV (2)

- Otto puts a copy of his code in a public Dropbox folder. Paul finds it, and submits a copy as his own solution. Who is guilty of an AIV?
- **BOTH**. Otto should have safeguarded his code.
  - Keep your code and other homework materials secure!
  - This also means not posting it publicly *even after the end of the course*.

AIV (3)

- Rosa finds code for a particular function that gave her a lot of trouble on StackOverflow. She carefully documents the source of that function's code in comments in her code. Is she guilty of an AIV?

AIV (3)

- Rosa finds code for a particular function that gave her a lot of trouble on StackOverflow. She carefully documents the source of that function's code in comments in her code. Is she guilty of an AIV?
- **NO.** But if the rules of the assignment disallowed using Internet code, she will still lose points *on that specific problem*.

Avoiding AIVs

- Attribute **anything** that isn't your own original work
  - usually as simple as a one-line comment in your code
- Attribute any collaborations with others (**including study groups**)
- Don't take the easy way out when you get stuck
  - it's better to have a poor score on one assignment worth 3% (or less) of your grade than losing a full letter or failing the course entirely
  - The TAs and your instructor are available to help you out
- In 2022, 16 of 115 students had AIVs and received lowered grades. In 2019, one enrolled student never got to start 11-601 after being expelled due to AIVs in a summer remote course.
  - Your instructor **hates** the extra paperwork AIVs cause. It makes him cranky.

Avoiding AIVs

- Attribute **anything** that isn't your own original work
  - usually as simple as a one-line comment in your code
- Attribute *any* collaborations with others (**including study groups**)
- Don't take the easy way out when you get stuck
  - it's better to have a poor score on one assignment worth 3% (or less) of your grade than losing a full letter or failing the course entirely
  - The TAs and your instructor are available to help you out
- In 2022, 16 of 115 students had AIVs and received lowered grades (6 of 97 in 2023). In 2019, one enrolled student never got to start 11-601 after being expelled due to AIVs in a summer remote course.
  - Your instructor **hates** the extra paperwork AIVs cause. It makes him cranky.

If you forget or run out of time, send email as soon as possible

OK or Not?

- OK
  - discussing textbook problems, lecture notes, etc.
  - suggesting an algorithm to try on a homework problem
- Dangerous
  - showing someone else your solution to a homework problem
  - working out pseudo-code together or sharing pseudo-code for a problem
    - you will definitely need to acknowledge this in your submission
- Don't even think about it
  - emailing/texting someone else your solution

OK or Not?

- OK
  - discussing textbook problems, lecture notes, etc.
  - suggesting an algorithm to try on a homework problem
- Dangerous
  - showing someone else your solution to a homework problem
  - working out pseudo-code together or sharing pseudo-code for a problem
    - you will definitely need to acknowledge this in your submission
- Don't even think about it
  - emailing/texting someone else your solution

these actions are OK
after everyone involved has
submitted their assignments

OK or Not?

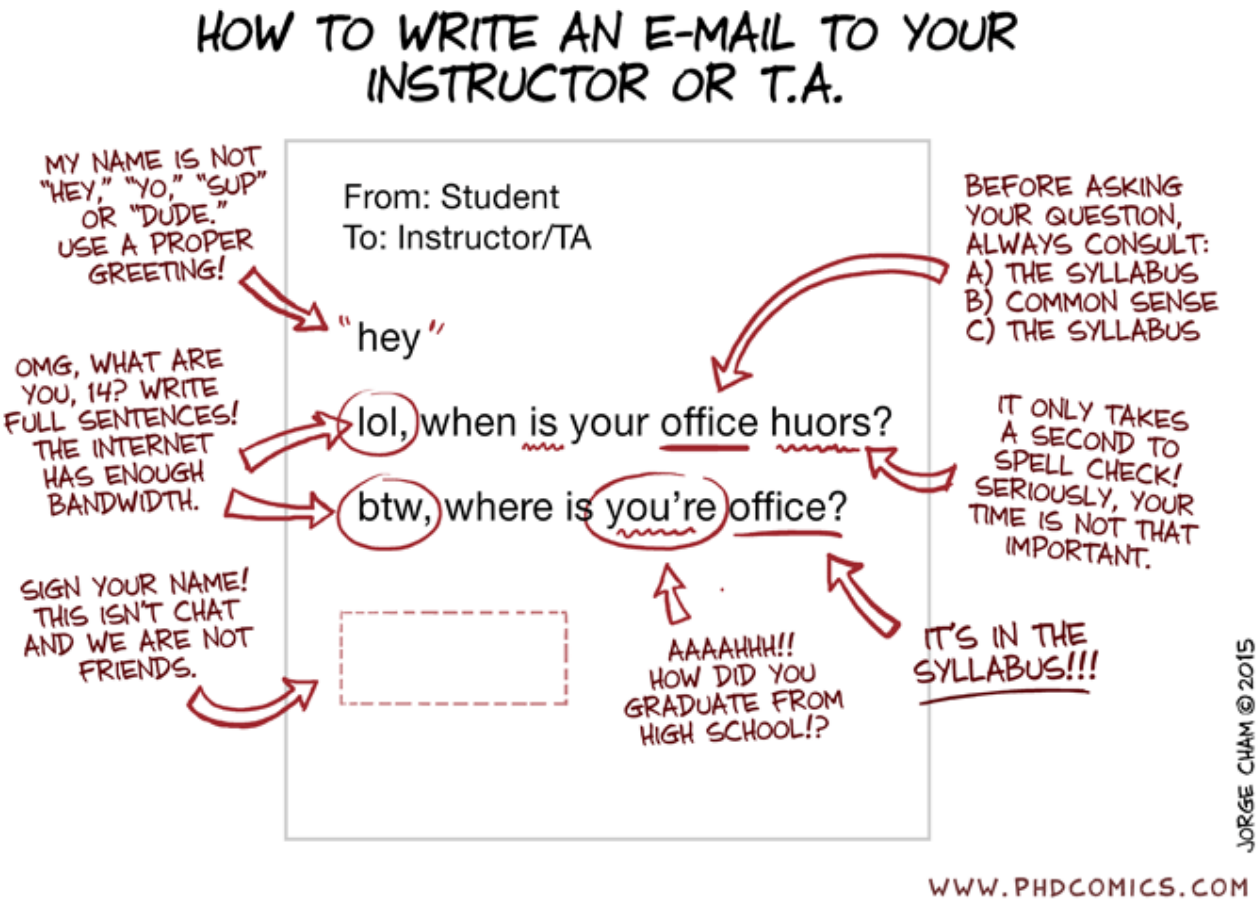
- OK
  - discussing textbook problems, lecture notes, etc.
  - suggesting an algorithm to try on a homework problem
- Dangerous
  - showing someone else your solution to a homework problem
  - working out pseudo-code together or sharing pseudo-code for a problem
    - you will definitely need to acknowledge this in your submission
- Don't even think about it
  - emailing/texting someone else your solution

....but this remains dangerous

(why?)

Re-Grade Requests

- If there is an issue with scoring (e.g. grade in Canvas differs from what you remember)
  - Send an email or Canvas message, or post (privately) to Piazza
  - State where you think the error is



Q&A

Java Strings and Arrays

Strings versus Character Arrays

- What is the difference?

Strings versus Character Arrays

- strings are *almost* always represented as an array of characters
- they have a means of determining length
  - explicit length field or sentinel character
- but in Java, arrays *also* have an explicit length attribute

Strings versus Character Arrays

- strings are *almost* always represented as an array of characters
- they have a means of determining length
  - explicit length field or sentinel character
- but in Java, arrays *also* have an explicit length attribute

  so the difference is mainly **conceptual**:
  - strings are treated as **single** objects
  - arrays are treated as **collections** of objects
- strings also have string-specific operations such as
  - comparison
  - insert/delete substrings, substring extraction

String Representations

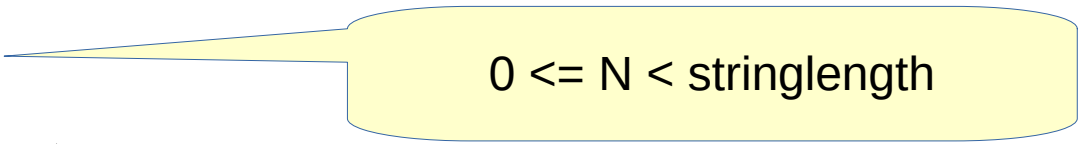
- Pointer to array terminated by sentinel (the “C” model)
  -
- Pointer to length field, followed by array of characters (the “Pascal” model)
  -
- Pointer to array of characters, preceded by length field
  -
- Pointer to a structure containing a length field and a pointer to the array of characters
  -
  -

String Representations

- Pointer to array terminated by sentinel such as NUL (the “C” model)
  - drawbacks: can't use sentinel value as data, strlen() is O(n)
- Pointer to length field, followed by array of characters (the “Pascal” model)
  - explicit length makes strlen O(1), may take more space or limit string length
- Pointer to array of characters, preceded by length field
  - may be more efficient to access; use negative index to access length field
- Pointer to a structure containing a length field and a pointer to the array of characters
  - takes more space than any of the above; accessing string value is slower due to extra indirection
  - but the extra indirection allows data sharing

Java String Conversions

- String to character array
  - String.toCharArray()
- extract character from String
  - String.charAt(int N)



- character array to String
  - String.valueOf(char a[]) static function
- character to String
  - Character.toString(char c) static function
  - String.valueOf(char c) static function

Some Basic String Operations

- Check the length:
  - Integer len = s.length();
- Concatenate two strings:
  - String result = s1 + s2;
  - String result = s1 + “text”;
- We will cover many more on Thursday

Building a String

- character by character

```
String s = new String(“”);
for ( ... )
   {
   s += ' ';
   }
```

- using StringBuilder

```
StringBuilder sb = new StringBuilder();
for ( ... )
   {
   sb.append(' ');
   }
```

Building a String

- character by character

```
String s = new String("");
for ( ... )
   {
   s += ' ';
   }
```

O(n^2) because we keep copying the partial result

- using StringBuilder

```
StringBuilder sb = new StringBuilder();
for ( ... )
   {
   sb.append(' ');
   }
```

O(n) because StringBuilder just updates a buffer in place

Building a String

- character by character

```
String s = new String("");
for ( ... )
   {
   s += ' ';
   }
```

O(n^2) because we keep copying the partial result

- using StringBuilder

```
StringBuilder sb = new StringBuilder();
for ( ... )
   {
   sb.append(' ');
   }
```

O(n) because StringBuilder just updates a buffer in place

We'll cover "big-O" notation on Thursday

Duplicate Check

- IsUnique – does the string contain any duplicate characters?
  - Q: what are different ways to perform this check?

Duplicate Check

- IsUnique – does the string contain any duplicate characters?
  - recursively check rest of string
  - sort and check adjacent characters
  - HashMap of counts
  - int[] or bool[]

Duplicate Check

- IsUnique – does the string contain any duplicate characters?
  - recursively check rest of string — $O(n^2)$
  - sort and check adjacent characters — O(n log n)
  - HashMap of counts — O(n)
  - int[] or bool[] — O(n)

All things are difficult before they are easy.

-- Thomas Fuller (1732)

Programming Exercise

Exercise 1 - Self-Assessment

- Now would be a good time to start the exercise by following the link provided in the email you received earlier today
- The median time taken by programmers world-wide for this problem is seven minutes. Had this been graded, you would have had nine minutes to complete the exercise.
- This exercise is at the level of Java proficiency being assumed as a prerequisite. Language features and standard library functions sufficient to complete the exercise were presented in this lecture.