

Compilers

Some history and advice as we begin
our adventure. . . .

Some Past Final Grades

A	5	A-	A	A	A	A				
B	5	B	B	B	B	B				
C	7	C-	C-	C-	C	C	C	C+		
D	1	D								
F	9	F	F	F	F	F	F	F	F	F
I	0									
W	1	W								

There will be no incompletes.

Don't even ask.

Some Past Final Grades

A	8	A	A	A	A	A	A	A	A		
B	5	B-	B-	B	B	B+					
C	9	C-	C-	C-	C	C	C	C	C+	C+	
D	3	D	D	D							
F	3	F	F	F							
I	0										
W	5	W	W	W	W	W					

Some Past Final Grades

A	8	A-	A-	A-	A-	A	A	A	A			
B	11	B-	B-	B-	B-	B	B	B	B	B	B+	B+
C	6	C-	C-	C-	C	C+	C+					
D	3	D	D	D								
F	11	F	F	F	F	F	F	F	F	F	F	F
I	0											
W	4	W	W	W	W							

Midterm vs. Final Grades

Mid-term grade vs. Final grade							
Mid-term Average	Mid-term Grade	Mid-term GPA		Final Grade	Final GPA		Change
93.33	A	4.0	→	C-	1.7	Δ	-2.3
12.67	F	0.0	→	F	0.0	Δ	0.0
86.67	B	3.0	→	A-	3.7	Δ	0.7
90.67	A-	3.7	→	C+	2.3	Δ	-1.4
65.00	D	1.0	→	C	2.0	Δ	1.0
92.33	A-	3.7	→	C	2.0	Δ	-1.7
70.00	C-	1.7	→	D	1.0	Δ	-0.7
41.67	F	0.0	→	F	0.0	Δ	0.0
92.67	A-	3.7	→	B	3.0	Δ	-0.7
46.67	F	0.0	→	F	0.0	Δ	0.0
35.00	F	0.0	→	F	0.0	Δ	0.0
93.67	A	4.0	→	B	3.0	Δ	-1.0
94.33	A	4.0	→	B	3.0	Δ	-1.0
48.33	F	0.0	→	F	0.0	Δ	0.0
92.33	A-	3.7	→	C	2.0	Δ	-1.7
94.33	A	4.0	→	B	3.0	Δ	-1.0
4.33	F	0.0	→	F	0.0	Δ	0.0
87.33	B+	3.3	→	C-	1.7	Δ	-1.6
96.33	A	4.0	→	A	4.0	Δ	0.0
98.67	A	4.0	→	A	4.0	Δ	0.0
93.00	A	4.0	→	B	3.0	Δ	-1.0
86.67	B	3.0	→	A	4.0	Δ	1.0
0.00	F	0.0	→	W	0.0	Δ	0.0
86.00	B	3.0	→	C-	1.7	Δ	-1.3
3.67	F	0.0	→	F	0.0	Δ	0.0

Advice

- ✦ Attend every class. Don't miss even one.
- ✦ Take notes in class, in pencil, on paper.
- ✦ Rewrite your class notes into a new notebook as a form of studying.
- ✦ Do every lab. Write them up in LaTeX.
- ✦ Work on your projects every day. Commit to GitHub at least once a day, if not more frequently.
- ✦ You **must** have already taken Algorithms. Seriously.

Advice

- Attend every class. Don't miss even one. Why?
 - > If you're not in class I will never have the opportunity to invite you to join a faculty research project, or the IBM Joint Study, or a hackathon, or a CCAC project, or an ECRL project, or a Summer Fellowship, or the Marist Innovation Lab.
 - > If you're not in class I will never know you well enough to write you a letter of recommendation or recommend you to graduate school or an employer.

ADVICE from Dr. Helen Rothberg



- ✦ **A**ction — Do more, say less. Pursue your interests. Ask for help when you need it.
- ✦ **D**etermination — Stick to it and get things done.
- ✦ **V**ision — What are you trying to become? What do you want to do with your life? How will you get there?
- ✦ **I**ntegrity — Tell the truth all the time. Don't create or participate in drama.
- ✦ **C**ommunication — Be attentive and dig in, even if it's hard.
- ✦ **E**mpathy — Dare to care about yourself and others.

Test Your Code

- ✦ Don't be the student in this conversation.

```
I had tested my code with success on one specific example and observed the desired output. Therefore, according to the cases that I tested, my code was working. However, what I failed to realize due to my oversight, was that I wasn't testing my code on other, different examples. It wasn't until then that I realized something was wrong.
```

So you're saying that failure to thoroughly and adequately test your code resulted in a costly mistake?

```
After digging around, I found out that the cause of the problem was because of 1 missing line of code within an else statement.
```

Wow... so -- again, failure to thoroughly and adequately test your code, which (again) led to a costly mistake that ... came down to ONE LINE OF CODE!

It's ALMOST like testing is ****really important**** and should never be overlooked. I wonder if you know any professors who make that point EVERY SINGLE CLASS.

```
Finding this was relieving but at the same time disappointing to a degree that I cannot begin to express. As of Wednesday, this correction has been added to my code.
```

Good.

```
I've been mentally plagued by the fact that my miniscule mistake here has costed me the full 30 points on the project and has effectively lost me 3% of my final grade.
```

Do Not Make Excuses

- ✦ This is not acceptable:

```
----- Original message -----
```

```
To: Alan Labouseur/FAC/Marist@Marist
```

```
Subject: RE: Projects
```

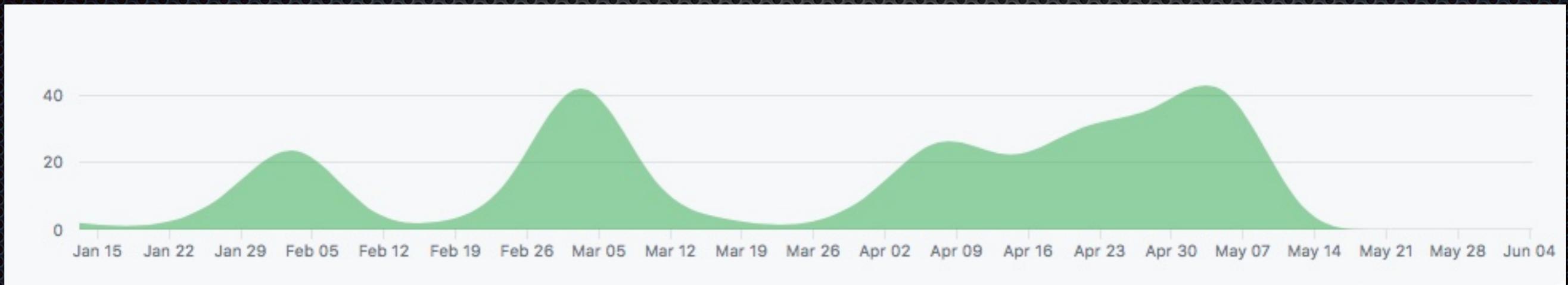
```
Hi Professor Labouseur,
```

```
I just looked at my email this morning, I'm surprised you didn't get my work. I thought I pushed it Sunday, But I now realize I didn't set upstream origin for the branch, so it didn't actually get pushed. I've just re-pushed this morning so you can review it.
```

- ✦ I did not “review it”. The grade remained the same.
- ✦ If you cannot figure out GitHub, you will not be earning a CS / ITS / Cyber / Data Science degree.
- ✦ Speaking of GitHub . . .

GitHub Commit Graphs

A and A-



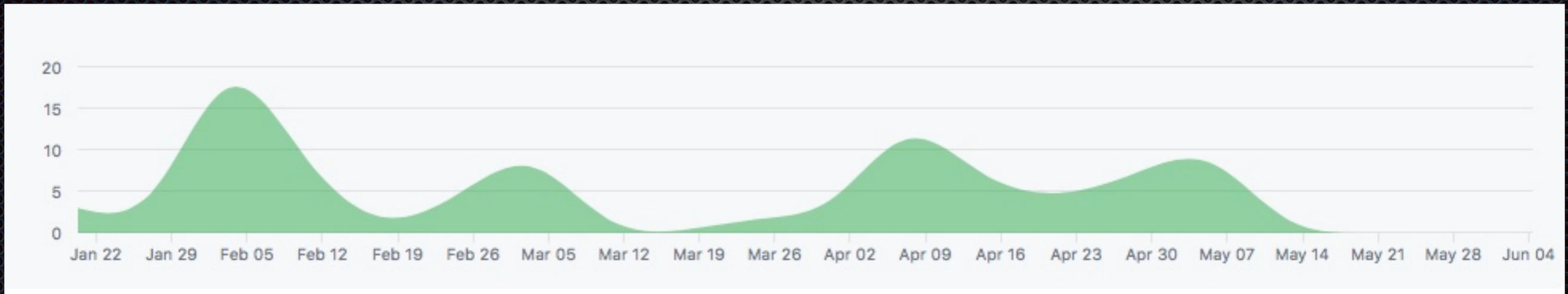
GitHub Commit Graphs

A and A-



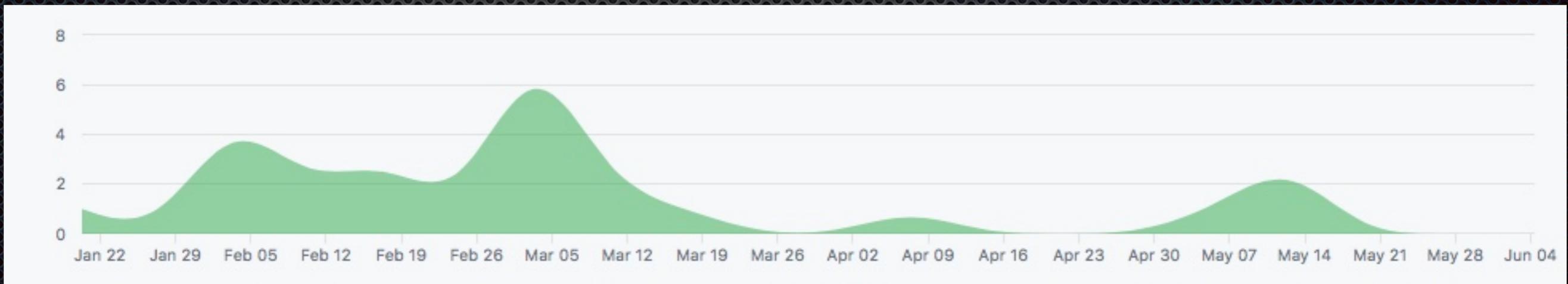
GitHub Commit Graphs

B and B-



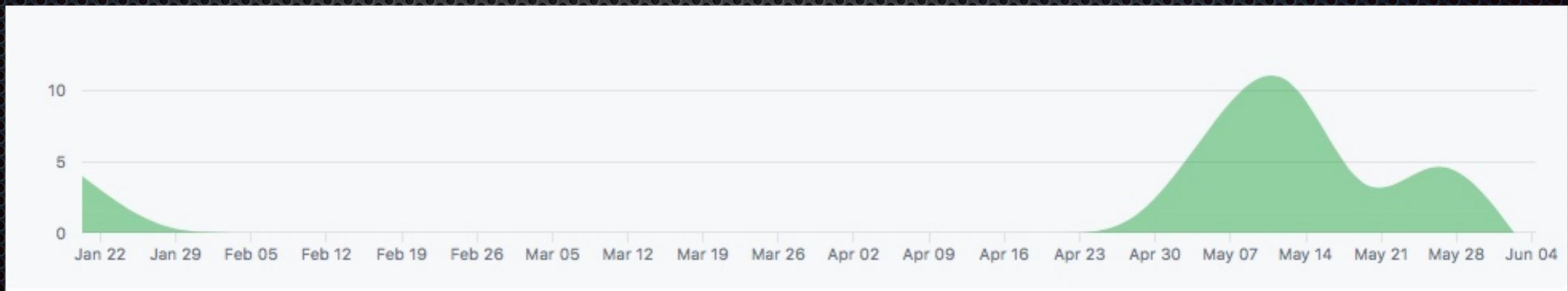
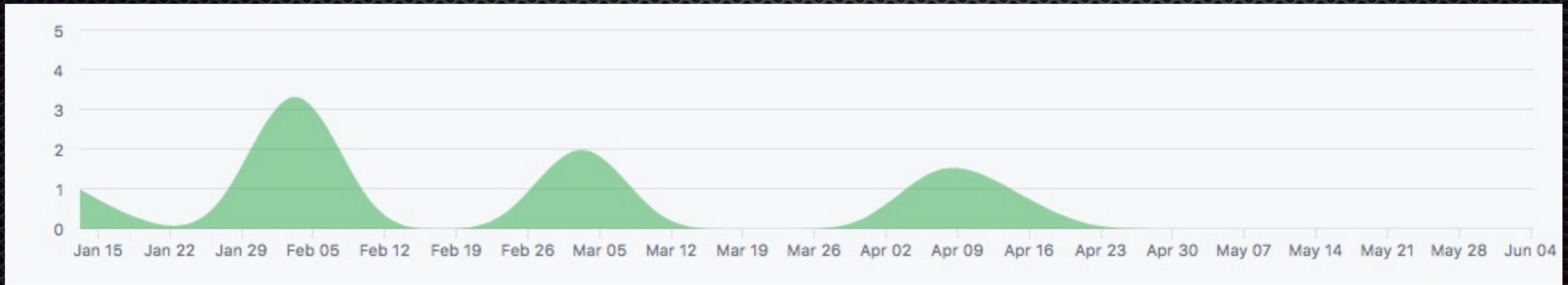
GitHub Commit Graphs

C and C-



GitHub Commit Graphs

F



GitHub Commit Graphs

F



GitHub Commits

- ✦ You must commit frequently over many days as you work on our project.
- ✦ This is an actual e-mail to a student regarding the first project.

```
There are too few commits to your GitHub repository. This is so far from acceptable that you can't even see it from here. Acceptable projects have 20 to 60 commits for the first project. And those commits are spread out over many days. That's what I want. Make it so.
```

- ✦ Do not be the next student to get an e-mail like this.

Academic Honesty

- ✦ This is college and we're in an academic environment, where Academic honesty is paramount.
- ✦ You are not working on a personal side project or partaking in a hackathon, or any situation where you might take shortcuts to get things done.
- ✦ You must do the work on your own.
- ✦ There are no shortcuts.

Academic Honesty

- ✦ These are individual projects. All work must be your own.
- ✦ Any violation of this will result in your **immediate** and **automatic** failure of this class. (“Insta-fail”)
- ✦ This is your only warning. The next time we talk about this it will be in the context of me telling you that you have failed this class.
- ✦ I am serious. Ask prior students.

Academic Honesty

- ✦ I will be checking your work against...
 - your class
 - other classes
 - past projects
 - ~~GPT Zero~~
 - and...

- ▶ 2004 Spring Marist
- ▶ 2005 Spring Marist
- ▶ 2007 Spring Marist
- ▶ 2009 Spring Marist
- ▶ 2010 Spring Marist
- ▶ 2010 Spring Vassar
- ▶ 2011 Spring Vassar
- ▶ 2013 Spring Marist
- ▶ 2013 Spring Vassar
- ▶ 2014 Spring Marist
- ▶ 2015 Spring Marist
- ▶ 2016 Spring Marist
- ▶ 2017 Spring Marist
- ▶ 2018 Spring Marist
- ▶ 2019 Fall Marist
- ▼ 2019 Spring Marist
 - ▶ project 1
 - ▶ project 2
 - ▶ project 3
 - ▶ project 4

Past Project Hall of Fame

JavaScript or TypeScript projects only. There have been some fantastic projects in Scala, C++, Python, and other languages, but they (obviously) will not execute in a web browser.

2019

Juice Compiler

2018

Sonar

Kompailer

The Colorful Compiler

Daniel Ahl's Compiler

Note: Earlier compilers used slightly different grammars compared to our current grammar.

2017

Tienminator

2016

Although some were close, none of the JavaScript or TypeScript projects were worthy of this hall of fame. There were some excellent projects implemented in other languages.

2015

Andrew B

Bloop Compiler

2014

Svegliator

Rob W

ChronOS.LL in OmniChron

Note: Earlier compilers used significantly different grammars compared to our current grammar.

2013

AnnaC

ChronOS

Max L

Morph6502

2010

harpO

PGP

2009

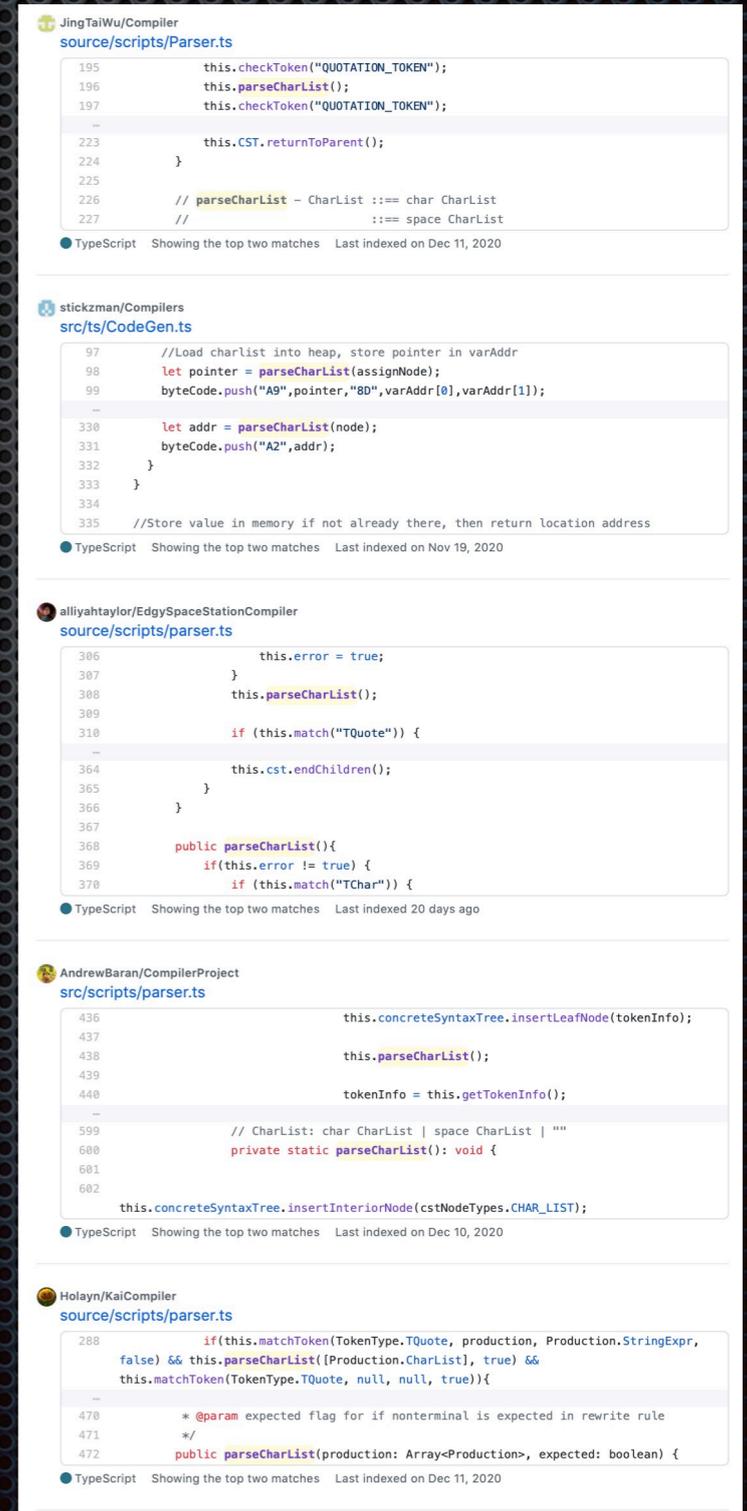
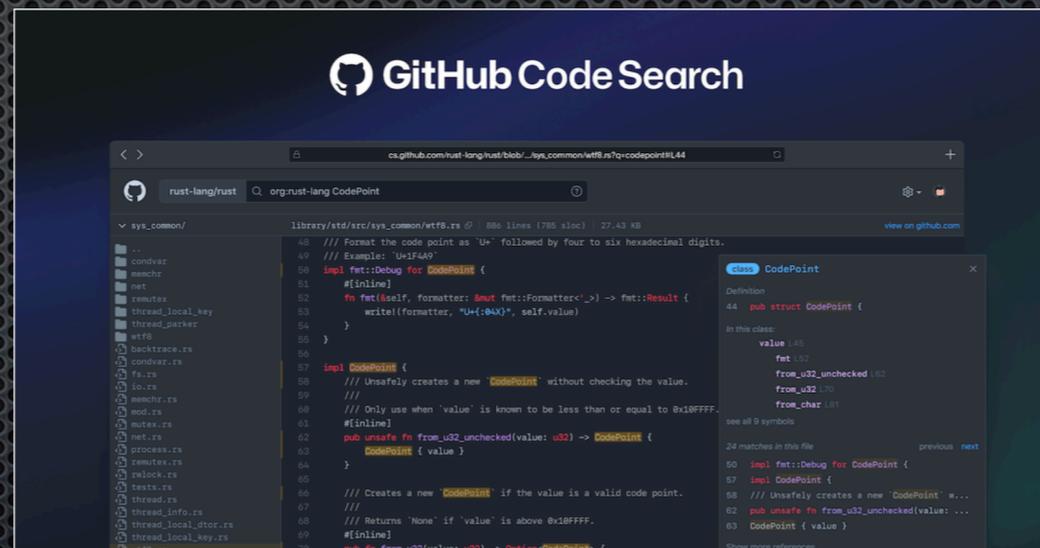
Gentlemen

Team Zero+

Academic Honesty

✦ I will be checking your work against...

- your class
- other classes
- past projects
- ~~GPT Zero~~
- and...
- GitHub



Academic Honesty

- ✦ I will be using MOSS to help detect cheaters. See theory.stanford.edu/~aiken/moss/

Moss

A System for Detecting Software Similarity

UPDATES

- Feb 1, 2018 *And even more community contributions have been added!*
- Nov 9, 2017 *More community contributions have been added below ...*
- Aug 31, 2017 *Thanks to Christophe Troestler for an [OCaml client](#) for Moss.*
- May 18, 2014 *Community contributions (including a Windows submission GUI from Shane May, thanks!) are now in their own section on this page.*
- May 14, 2014 *And here is a [Java version](#) of the submission script. Thanks to Bjoern Zielke!*
- May 2, 2014 *Here is a [PHP version](#) of the submission script. Many thanks to Phillip Rehs!*
- June 9, 2011 *There were two outages over the last couple of days that lasted no more than a hour each (I think). I've made some changes to the disk management software that should prevent these problems from recurring.*
- April 29, 2011 *There was an outage lasting a few hours today, the first since last summer, but everything is back up.*
- August 1, 2010 *Everything is back to normal.*
- July 27, 2010 *The Moss server is back on line. There may be some more tuning and possibly downtime in the coming weeks, but any outages should be brief. New registrations are not yet working, but people with existing accounts can submit jobs.*
- July 25, 2010 *As many (many!) people have noticed, the Moss server has been down for all of July. Unfortunately the hardware failed while I was away on a trip. I am hopeful it will be back up within a few days.*

What is Moss?

Moss (for a Measure Of Software Similarity) is an automatic system for determining the similarity of programs. To date, the main application of Moss has been in detecting plagiarism in programming classes. Since its development in 1994, Moss has been very effective in this role. The algorithm behind moss is a significant improvement over other cheating detection algorithms (at least, over those known to us).

Write Original Code

- ✦ Infuse your code with your personality.
 - variable names, code conventions, and style
 - structure and modularization
- ✦ Write grammatically correct comments and steep them with your personality. Be snarky, clever, and funny if you like. I want to get to know you by reading your code.
 - Comments should say things about the code that the code cannot say about itself.
- ✦ List collaborators, references, and sites consulted in a comment block at the top of each file.

What about using an AI?

- I've been teaching my teenage son to drive. We started with golf carts and then moved to a real car in empty parking lots.
- He's making incremental progress built on a solid foundation (and cart paths). But he's a long way from driving in F1.
- If I were to put him in an Aston Martin he'd surely hurt himself and others.
- AI code generators are like F1 racecars. Until you have a solid foundation you're only going to hurt yourself and others by using them. But if you feel that you're ready... DRS is enabled so go for it!



When using an AI

- As ever, anything that's not 100% yours and yours alone **must be marked and cited with sources**. Whether you get help from an online AI, a tutor in the programming lab, or a random student sleeping outside of Marion Hall, you have to acknowledge that help and cite it or it is unquestionably plagiarism. Copying and pasting from a Hall of Fame project or GitHub or some other online source is unquestionably plagiarism unless you cite it; and this is no different.
- How do you cite it? Let's ask ChatGPT . . .

When using an AI



Write a citation for ChatGPT in ACM reference format



As of my last knowledge update in September 2021, there isn't a specific ACM (Association for Computing Machinery) reference format for citing AI language models like ChatGPT, as these models are relatively new and standards may not have been established at that time. However, you can follow a general citation format in accordance with ACM guidelines, and adapt it as needed based on any new guidelines that may have emerged after my last update. Here's a general citation in ACM format:

less

Copy code

```
[Author(s)]. [Year]. "ChatGPT: A Language Model for Natural Language Underst
```

Please replace `[Author(s)]`, `[Year]`, `[Name of the Conference or Journal]`, `[Volume number](Issue number)`, `[Article number]`, `[Page numbers]`, and `[DOI if available]` with the specific information relevant to the version or publication of ChatGPT you are citing. If there are specific guidelines or recommendations from ACM for citing AI language models as of 2023, I recommend following those guidelines for the most accurate and up-to-date citation format.

When using an AI

- I'm interested in learning about how well (or not?) AI-assisted software development works for you. If you make use of an AI you must document your experience with it for every project: the chat interactions, what worked and what didn't, challenges, unexpected occurrences, etc. Include a reflection about the experience as well. I'm interested to learn from you and your experiences with this new technology.
- Remember: plagiarism is one of the few sins where it is easier to receive permission than forgiveness. So if you are at all unsure about any aspect of this, ask me about it first.

Before asking me for help

- ✦ Review your code yourself by reading it from the perspective of someone seeing it for the first time.
 - Consider your variable names — are they descriptive?
 - Did you write plentiful comments about why you made those choices rather than what the code does?
 - Examine your code conventions and style — is it consistent?
 - Fix anything that's missing, unclear, or confusing.

Before asking me for help

- Set a break point in your browser or IDE and step through your code line by line for several test cases.
 - This is the first thing I'm going to ask you to do, so you might as well do it preemptively.
 - You will be amazed at how many mistakes, bugs, and off-by-one errors you'll find that way.

The screenshot shows a terminal window at the top with the following commands and output:

```
root@sampc~$ ls
testfile.txt
root@sampc~$ delete testfile
```

Below the terminal is a browser's developer tools interface. The 'Sources' panel is active, showing the file `fileSystem.js` with a blue arrow indicating a breakpoint at line 80. The code at this line is:

```
if (!FileSystem.validAddr(dirStartAddr)) {
```

The right-hand side of the developer tools shows the 'Scope Chain' panel with the following information:

- Watch Expressions:** No Watch Expressions
- Block Variables:** `s: 0`
- Local Variables:**
 - `fname: "testfile"`
 - `this: FileSystem {swpCnt: 1, swpMap: Mi`
- Block Variables:** (empty)

When asking me for help

- ✦ Explain the problem clearly. Context matters.
- ✦ Describe several test cases that illustrate the problem.
- ✦ Tell me what you observed when stepping through your code line by line.
- ✦ Make sure the latest version is committed on GitHub and include a link to it in your e-mail.