

Algorithms

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

Advice

- ✦ Attend every class. Don't miss even one.
- ✦ Put your devices away and pay attention in class.
- ✦ Take notes in class (in pencil) on paper.
- ✦ Rewrite your class notes into a new notebook as a form of studying.
- ✦ Do every lab. Sometimes twice. Commit to GitHub.
- ✦ Work on your projects every day.

Advice

- Attend every class. Don't miss even one. Why?
 - > If you're not in class I won't 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 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.

Do Not Make Excuses

- ✦ I love Chris Algozzine, but I don't give a damn about Capping.
- ✦ These people failed the class.

----- Original message -----

To: Alan Labouseur/FAC/Marist@Marist
Subject: Projects
Date: Sun, Dec 9, 2018 8:33 PM

Also, I wanted to say sorry for the subpar work. I've just been incredibly overwhelmed by capping. I enjoy having you as a professor and I don't want you to think I'm just blowing your class off, I've just been trying to keep my head above water.

----- Original message -----

To: Alan Labouseur/FAC/Marist@Marist
Subject: Projects

I chose to dedicate my productive hours towards my capping project, due to the fact that I prioritize my team's performance highly.

- ✦ Don't even mention Capping to me. Seriously.
- ✦ Work on your project and assignments **every** day.

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.

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.
- ✦ 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 and...

Assignments

Embrace Computer Science scholarship and write your labs using LaTeX. **OverLeaf** is probably the easiest way to do this. I've written a **lab example you can use as a template**. There are more LaTeX resources **below**.

Assignment Zero - Limericks in LaTeX

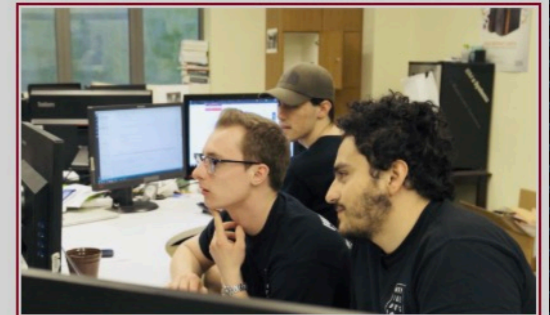
Assignment One - Data Structures and **magic items**.

Assignment Two - Sorting **magic items**.

Assignment Three - Searching and Hashing **magic items**.

Assignment Four - **Graphs** and Trees (and **magic items to find in your BST**)

Assignment Five - Dynamic Programming (**graphs**) and Greedy Algorithms (**spice**)



Academic Honesty

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

```
crav12345/RavosaAlgorithmAnalysis
Algorithms4Ravosa/IntergalacticReaderRavosa.java
7 // constructing several directed, weighted graphs. The program will produce
8 // those graphs and implement the Bellman-Ford path finding algorithm on them.
-
- //We print the path in Bellman-Ford.
21 public static int formatter = 0;
22
23 public static void main(String[] args) {
-
- Python Showing the top two matches Last indexed on May 17, 2019

CameronUlrich/CMPT435Algorithms
Assignment Four/AssignmentFour.py
45 edge.v.distance = edge.u.distance + edge.weight
46 edge.v.predecessor = edge.u
47
48 def BellmanFord(self):
-
- return False
142
143
144
145
146 def main():
147 print("Bellman-Ford dynamic programming algorithm for Single Source Shortest
Path:")
-
- Python Showing the top two matches Last indexed on May 13, 2019

rsbmarist/cmpt435bradshaw
Assignment4/Main.java
13 //This while loop runs the graph reading and bellman-ford methods.
14 while(graphReader.hasNextLine()){
-
- //Run the bellman ford method on the last graph.
63 bellmanFord(g, 1);
64
65
66
67
68 File spiceFile = new File("spice.txt");
69 Scanner spiceReader = new Scanner(spiceFile);
-
- Java Showing the top four matches Last indexed on May 10, 2019
```

GitHub Code Search

rust-lang/rust org:rust-lang CodePoint

```
library/std/src/sys_common/wtf8.rs | 886 lines (785 sloc) | 27.43 KB
45 // Format the code point as 'U' followed by four to six hexadecimal digits.
46 // Example: 'U+1F4A9'
47 impl fmt::Debug for CodePoint {
48     #[inline]
49     fn fmt(&self, formatter: &mut fmt::Formatter<'>) -> fmt::Result {
50         write!(formatter, "U{:04X}", self.value)
51     }
52 }
53
54 impl CodePoint {
55     /// Unsafely creates a new 'CodePoint' without checking the value.
56     ///
57     /// Only use when 'value' is known to be less than or equal to 0x10FFFF.
58     #[inline]
59     pub unsafe fn from_u32_unchecked(value: u32) -> CodePoint {
60         CodePoint { value }
61     }
62
63     /// Creates a new 'CodePoint' if the value is a valid code point.
64     ///
65     /// Returns 'None' if 'value' is above 0x10FFFF.
66     #[inline]
67     pub fn from_u32(value: u32) -> Option<CodePoint> {
68         CodePoint::from_u32_unchecked(value).or(None)
69     }
70 }
```

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 (including AI), 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

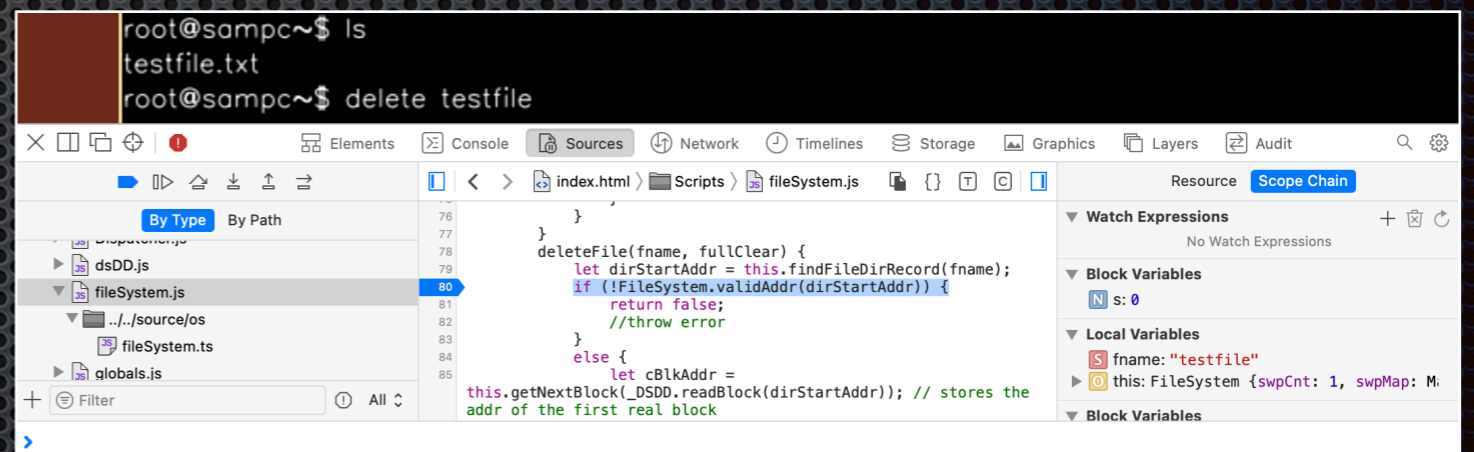
- I'm interested in learning about how well (or not?) AI-assisted software development works for you. If you make use of an AI you must document your experience with it for every project: the chat interactions, what worked and what didn't, challenges, unexpected occurrences, etc. Include a reflection 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
 - (1) mistakes,
 - (2) bugs, and
 - (2) off-by-one errors you'll find.



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 developer tools window. The 'Sources' panel is open to 'fileSystem.js', and the code is being stepped through. The current line of code is:

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

The 'Local Variables' panel on the right shows the following variables:

- fname: "testfile"
- this: FileSystem {swpCnt: 1, swpMap: M...

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. (Do not ask me for help unless you have already done this. Seriously.)
- ✦ Make sure the latest version is committed on GitHub and include a link to it in your e-mail.