

# Justice “JJ” James

I like general mathematics, [programming/\_] language theory, and Taylor Swift.

email: [work@toki.la](mailto:work@toki.la)  
website: [apropos.codes](https://apropos.codes)  
github: [omentic](https://github.com/omentic)  
linkedin: [~jj](#)

## Education

### University of British Columbia

September 2021 - April 2025 [expected]

- Year: 3rd | Major: [Mathematics](#) | Minor: [Linguistics](#)

## Skills & Interests

- **Interests:** type theory, computer security, compiler engineering, abstract algebra, language design
- **Experienced with:** Nim, Rust, Java, Agda, Python, Racket, C, Bash, HTML/CSS, Tailwind CSS, Toki Pona
- **Currently learning:** Haskell, [Java/Type]Script, Lean, Latin, category theory, type theory, cryptography
- **Assorted skills:** Linux, Git, server administration, HAProxy, Nginx, Docker, cross-site scripting, SQL injection, template injection, request smuggling, CSRF, reverse engineering, digital forensics analysis

## Work & Volunteering

### Hashbot [Rust]: spam protection for Discord servers

July 2023 - October 2023

- Worked on transitioning an image matching service across databases for better decoupling & scalability
- Freelance contract work

### ICFP 2023: International Conference for Functional Programming

September 2023

- Student volunteer and attendee

### PLDI 2023: Programming Language Design & Implementation Conference

June 2023

- Virtual student volunteer

## Relevant Coursework

### CPSC 539b: Dependent Types

Fall 2023 (current)

- Working through Dan Friedman's and David Thrane Christiansen's [The Little Typer](#)
- Working through David Thrane Christiansen's [Checking Dependent Types by Normalization by Evaluation](#)
- Working on **implementing a dependent type checker** by normalization by evaluation

### CPSC 539b: Implementing Type Systems

Spring 2023

- Worked through and discussed Benjamin C. Pierce's [Types and Programming Languages](#)
- Discussed a bevy of additional papers: primarily Jana Dunsfield's work on [bidirectional typechecking](#)
- **Implemented a type system in Rust as a term project**, with bidirectional typechecking, subtyping, and algebraic data types, later extended with typeclasses

### CPSC 421: Introduction to the Theory of Computing

Fall 2022

- Finite Automata, Pushdown Automata, Turing Machines, Context-Free Grammars, Reductions, Recognizability, Decidability, Computability Theory, Complexity Theory, Complexity Theory
- Course notes typeset and available virtually on my website: <https://apropos.codes/notes/cpsc421/>
- Wrote [a Turing machine emulator](#) and [a hand-rolled Fibonacci program](#) (for fun and for an art piece)
- Wrote [a Wang tile implementation](#) of a Fibonacci program (also for fun and for an art piece)

### CPSC 311 Reading Group: Introduction to Interpreters

Fall 2022

- Taken as a replacement for an interpreters course (cancelled by departmental scheduling)
- Weekly group meetings [nerd club](#), discussing and working through Shriram Krishnamurthi's [Programming Languages: Application and Interpretation](#) (second edition)

### CPSC 411: Introduction to Compiler Construction

Spring 2022

- Lectures only. Did not attempt assignments. Plan to properly take Spring 2025. (scheduling... :-<

## Experience

---

### **Maple Bacon: a competitive cybersecurity Capture the Flag (CTF) team** *September 2021 - Present*

- **#1 team in Canada**, peaking at **#7 worldwide** in May 2022 (ending the year at **#25 worldwide**)
- **Executive** (September 2022 - present): creating challenges and running meetings and handling logistics
- **Won DEF CON CTF 30 and 31** alongside CMU and Theori.io (as the Maple Mallard Magistrates)
- Specializing in **miscellaneous** security-related problems, **reverse engineering** compiled binary code, and modern **web exploitation**, with an interest in learning cryptography and binary exploitation
- Assisted in organization and challenge development for **SaplingCTF** and **MapleCTF**

### **SaplingCTF and MapleCTF: cybersecurity competitions** *February & September 2022 & 2023*

- Co-coordinator: helped in handling logistics and assisted in development of educational challenges
- SaplingCTF 2022: saw over **75 teams** and **175 participants** locally from the University of British Columbia
- MapleCTF 2022: saw over **600 teams** and **1,500 participants** virtually online from across the globe
- Additionally hosted both SaplingCTF 2023 and MapleCTF 2023 to similar success

### **CursedCTF: a satirical (but actual) CTF competition** *April 2023*

- Wrote a number of challenges, *carefully crafted the website experience*, and assisted in organization
- Ran incredibly successfully, with over 550 teams & active participants and a CTFTime rating of 23.33/25

### **UBC Bionics: engineering design team building a bionic arm** *September 2021 - June 2022*

- Architected an asynchronous Rust framework to bridge statistical models in Python and kernel code
- Designed around various I/O, threading, Python, and hardware limitations

### **Spartronics 4915: a high-school FIRST robotics team** *September 2017 - June 2021*

- Programmer (September 2017 - June 2021), Leadership Member (June 2018 - June 2021)
- **Programming Lead** (June 2019 - June 2021), **Team Captain** (June 2019 - June 2021)
- Designed and lead the implementation of a modular command-based robot codebase
- Worked closely with experienced industry mentors to facilitate student learning and student-mentor connections through pair programming, code review, and an educational curriculum

## Projects

---

### **apus [Java] and bamboo [Nim]: a pair of web browsers from scratch**

- Entirely from-scratch implementations: only rely on the standard **Sockets API** for network access
- Implements **HTML parsing (HTML LS)**, **URI handling (RFC 3986)**, **partial HTTP 1.0 support (RFC 1945)**
- Custom layout engine + renderer with support for **basic CSS**: built on **Swing [Java]** and **Pixie [Nim]**

### **chrysanthemum [Rust]: a simple language with a complex type system**

- Models and interprets the **simply typed lambda calculus** through Rust's algebraic data types
- Checks and infers types **bidirectionally**: with support for extensive **subtyping** and **typeclasses**
- Support for a broad variety of types: bools, nats, ints, floats, chars, strings, structs, enums, functions

### **nim-uxn [Nim]: a macro-full implementation of an abstract stack machine**

- Implemented the core VM of **Uxn**, an aesthetic minimalistic computing stack, in a custom DSL
- Future plans to implement I/O and package as both a library and an emulator

### **athena [Java]: a modular codebase for a competitive FIRST Robotics Competition robot**

- Designed the codebase to be approachable as an introduction to programming as a whole
- Command-based architecture separates code into owned, self-contained modules

### **additional projects**

- **advent-of-code** [Various]: aesthetic and efficient solutions to yearly programming puzzles
- **dictionary** [Rust]: a speedy, offline, command-line dictionary program
- **viceverba** [JavaScript]: a minimal port of the Latin learning app Vice Verba to the web
- **...and more!**