Anish Rajeev

770-401-6112 | anishrajeev@proton.me | LinkedIn | GitHub

Education

Carnegie Mellon University | Pittsburgh, PA Graduation: May 2026
Bachelor of Science in Mathematics (Discrete Math & Logic Concentration) *GPA:* 3.8
Relevant Courses: *HOT Compilers, Functional Programming, Intro to Computer Systems, Data Structures and Algorithms*

Experience

Software Engineering Intern| *Python, Llama, Langchain* June 2025 – August 2025
Capital One

- Working on developing a model to automatically generate + edit mappings between cybersecurity frameworks, generally done manually by analysts
- Created a model architecture that used an MCP (model context protocol) server to call tools that implemented various semantic and syntax searching methods including RAG (retrieval augmented generation), MQR (multi query retrieval), and BM25
- The accuracy for mappings outputted by the model was 85%

Software Engineering Intern| *Python, Pandas, Hadoop* May 2023 – August 2023
General Motors

- Used Pandas, SQL, and Hadoop to analyze call data and spending patterns, proposed changes that raised customer spending by 10%
- Identified a loophole in the user flow leading to a reduction in manual intervention on calls by 2% by utilizing Python and Pandas for analysis

Projects

Adjoint Natural Deduction Compiler | *OCaml, Menhir, Agda*

- Created a compiler for a functional, higher-order typed ML-style programming language based off of Adjoint Natural Deduction
- Has algebraic data types (along with two additional type constructors, being the upshift and downshift), with equirecursive types
- Types are tagged with a ”mode” from a preorder of modes (ADTs are parameterized by a mode variable), representing a substructural paradigm (Linear, Affine, Strict, Structural), with the type checker checking that the conditions on values are satisfied using an additive approach
- Language compiles down to an imperative intermediate language based on the Adjoint Semi-axiomatic Sequent Calculus, where several optimizations are then applied while retaining the original semantics of the program
- Compiles from SAX to C, using closure conversion to compile negative types like functions, lazy records, and upshifts (thunks)
- The language has several features, maintaining a comfortable experience to program in; mode inference, bidirectional type checking, nested pattern matching, etc.
- Currently working on proving the correctness of the inner core of the compiler in Agda

Activities

Oregon Programming Language Summer School | *Boston University* June 2024

- Accepted to attend a summer conference focused on advanced developments in programming languages
- Studied a multitude of materials in preparation, from canonical textbooks (like *Types and Programming Languages*) to lots of research papers
- Attended sessions taught by current programming language researchers about a variety of topics in current research, from proof theory to probabilistic programming

Competitive Programming

- Invested 5000+ hours solving complex algorithmic problems (mainly in Java) ranging from topics like Fenwick Trees to Line Sweeps
- Won several competitions, achieving rankings such as Expert rank on CodeForces and USACO Gold

Technical Skills

Relevant Skills: OCaml, Java, C, SML, Agda, Scheme, Emacs