Devin Lehmacher

112 Sage Place Room-209, Ithaca, NY 14850
djl329@cornell.edu • +1 (864) 722–3014 • github.com/lehmacdj

EDUCATION

Cornell University, Ithaca, NY — Aug 2015 — Present

- Graduating in May 2019
- Bachelors of Arts in Computer Science
- Cumulative GPA: 3.52, Major GPA: 3.95

CLASSES

Introduction to Compilers & Practicum, CS 4120 & CS 4121
Certified Software Systems, CS 6115
Constructive Type Theory, CS 6180
Intro to Analysis of Algorithms, CS 4820
Intro to Theory of Computing, CS 4810
Advanced Programming Languages, CS 6110
Operating Systems & Practicum, CS 4410 & CS 4411
Database Systems, CS 4320
Computer System Organization, CS 3410
Functional Programming and Data Structures, CS 3110
Discrete Structures, CS 2800
Object Oriented Programming and Data Structures, CS 2110

WORK EXPERIENCE

Teaching Assistant, CS 2110 at Cornell University — Feb 2016 — Present

- Teach a section with about 25 students each week
- Hold weekly office hours to help students understand the course material
- Help test, create, and plan future assignments
- Grade assignments and exams, giving students helpful feedback

Intern at Itron Inc. in Oconee, SC — Jun 2017 — Aug 2016

- Created a dashboard to visualize available space for testing meters
- Utilized Transact-SQL to collect data for the dashboard
- Built and deployed reports to Sharepoint using Microsoft Reporting Services

Research Assistant at Clemson University — Jun 2015 — Aug 2016

- Tested the performance of MedusaLoop, a program that models protein loops
- Analyzed test results to visualize performance
- Wrote a daemon to dispatch jobs from a database to a server instance
- Wrote back end code that interacted with a database to fetch and write new jobs

PROJECTS

PortOS, CS 4411

- Implemented multithreading with preemption, and TCP and UDP analogs
- Learned how to navigate and write a large (10,000 lines) C code base
- Wrote safe, concurrent, robust C code

OCalf Interpreter, CS 3110

- Built an interpreter for a small subset of OCaml
- Learned how to evaluate an AST for a functional language using small step semantics
- Implemented Hindley-Milner type inference algorithm to type check OCalf programs

Scheme Interpreter, github.com/lehmacdj/haskell_scheme

- Built an interpreter for a subset of Scheme
- Learned how to implement the semantics for dynamically typed programming languages
- Learned how to build a parser using Parsec

Heaplib, CS 3410

- Implemented and tested malloc, free, and resize in C
- Learned how to use raw pointers and the trade-offs involved with building an allocator
- Wrote a large number of tests to ensure that pointer arithmetic was correct

SKILLS

Fluent: Java, Haskell, git, Vim, C, OCaml, Rust
Familiar: SQL, shell scripting, C++, Python, Perl