Devin Lehmacher

112 Sage Place Room-B09, Ithaca, NY 14850
djl329@cornell.edu • +1 (864) 722–3014 • github.com/lehmacdj

EDUCATION

Cornell University, Ithaca, NY — Aug 2015 — Present
- Expect to graduate May 2019
- Cumulative GPA: 3.45
- Bachelors of Arts in Computer Science (Major GPA: 3.82)
- Bachelors of Arts in Biology (Major GPA: 3.60)

CLASSES

Advanced Programming Languages, CS 6110	Spring 2017
Operating Systems & Practicum, CS 4410 & CS 4411	Spring 2017
Database Systems, CS 4320	Fall 2016
Computer System Organization, CS 3410	Fall 2016
Functional Programming and Data Structures, CS 3110	Spring 2016
Discrete Structures, CS 2800	Spring 2016
Object Oriented Programming and Data Structures, CS 2110	Fall 2015
C++ Programming, CS 2024	Fall 2016

WORK EXPERIENCE

Teaching Assistant, CS 2110 at Cornell University — Spring 2016 — Present
- Teach a section with about 25 students each week
- Hold weekly office hours to help students understand the course material
- Updated and wrote solutions for an assignment
- Grade assignments and exams, giving students helpful feedback
-

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

MIPS Processor, CS 3410
- Designed a MIPS processor in Logisim and tested it with programs written in assembly
- Learned how to decode binary MIPS instructions into control signals

OCaml Ed, github.com/lehmacdj/ocaml-ed
- Implementation of ed, the 1960s line editor, written using OCaml
- Wrote clear error handling code that cleanly passes errors up to the top level

SKILLS

Fluent: Java, C, Shell, git, Vim, OCaml, Haskell

Familiar: Rust, SQL, C++, Python, Perl