First Last

myemail@gmail.com • +8 (888) 888-8888 • gh-username • website.url • li-username

Education

- **University of Waterloo** *Sep 2021 – Present*
  Honours Bachelor of Computer Science; cGPA: 95.10%
  - **Coursework**: Operating Systems, Algorithms, Data Structures, Databases, Object Oriented Programming, Statistics, Linear Algebra, Combinatorics
  - **Scholarships**: Faculty of Mathematics Scholarship, President's Scholarship of Distinction

Skills

- **Languages**: C, C++, Python, Java, Scala, Bash, Rust, JavaScript, TypeScript
- **Tools/Platforms**: Linux, Docker, Nginx, DPDK, Django, Flask, Node, Express, Google Cloud Platform

Experience

- **(Incoming) D. E. Shaw & Co. – Software Engineering Intern** *June 2024 – Aug 2024*
  Researching and implementing real-time distributed systems for trading activites
- **NVIDIA – Software Engineering Intern (GeForce NOW Core Streaming)** *Jan 2024 – Apr 2024*
  Built a high-performance video snapshotting tool to troubleshoot GeForce NOW issues, a product with 20 million users (C++)
  - Developed snapshot tool to capture snapshots of raw frames and encoded bitstream on streaming servers, allowing the client to highlight corruption and other video-related issues
  - Implemented raw frame capture tool that could process 750 MB/s of video (4K 60fps NV12 frames) without performance hits ($<$ 1fps drop)
  - Updated video stream networking protocol to support remote configuration, triggering, and feedback for the snapshot feature directly from the client
  - Collaborated with senior team members to assess requirements, scope, and design of the feature
- **Huawei – Software Engineering Intern (Computer Networking R&D)** *May 2023 – Aug 2023*
  Engineering protocols and libraries for high-performance, low-latency networks (C, DPDK)
  - Created library using Dataplane Development Kit (DPDK) to process UDP packets, with up to 18x speed improvement over Linux kernel sockets
  - Collaborated to design a connectionless and reliable transport layer protocol on top of UDP
  - Designed and implemented library for the protocol in C, employing lock-free data structures and message passing to achieve processing of 150+ concurrent connections on a single machine
  - Developed logging software to track and troubleshoot performance with minimal (<<1%) impact on runtime
- **AMD – Software Engineering Intern** *May 2022 – Aug 2022*
  Developed and maintained debugger for 200+ Kernel-Mode Driver developers across AMD (C++, Python)
  - Added feature to inspect hardware scheduled queues, enabling debugging of critical launch-gating issues
  - Proposed and implemented improvements across unit test (GTest) and CI pipeline, reducing test code size by up to 50%
  - Automated formatting of tables, nested lists, and dictionaries, ensuring consistent output and deduplication of code

Awards/Competitions

- **ICPC ECNA Regionals**: Represented Waterloo in 2021 and 2022, placing 4th and 6th against 90+ other university teams
- **Putnam 2021**: Placed in the top 500 of the most famous University-level mathematics competition
- **USACO Open 2021 (Platinum)**: 32nd of pre-college contestants in the highest division of the USA computing olympiad
- **Google Code Jam 2021**: 3rd of Canadian contestants (top 0.1%) in computing competition with 90,000+ contestants
- **Canadian Computing Olympiad 2021**: Achieved a silver medal, placing top 10 in a national level contest

Projects

- **C++ Game Engine**: Designed and implemented object-oriented (OOP) game engine built around the MVC pattern. The engine supports handling a variety of entities with built in entity movement, collision detection, and a graphics library
- **Competitive Programming Tools**: Tools to speed up implementation and debugging during programming contests. Includes automated testing with test case generation, and browser integration. Available as a Python CLI or VSCode extension (TypeScript and React.JS)
- **LACS Compiler**: Scala-based compiler for functional language targeting MIPS. Includes support for closures, nested functions, static typing, garbage collection, and tail-call optimization