


## EDUCATION

- **ONGOING BSc Computer Science** – University College London – 2019-2022

Relevant modules & courseworks (able to provide code upon request):

- Principles of Programming: C & Haskell
  - \* Built a journey planner for the  London rail network (tube & others) entirely in C.
    - Shows paths with least station, least interchange or paths that avoid certain fare zones.
- Engineering Challenges 1: digital circuit, MIPS architecture.
  - \* Built a simplified MIPS computer (8 instructions implemented) in a 2 person team on FPGA by drawing schematics. Self-taught Verilog and applied to the project.

- **A-Level** – 2017-2019





Math, Further Math, Physics and Economics – A\*A\*A\*A\*

## SKILLS

- Languages: C, Rust, Go, JavaScript, Python, T<sub>E</sub>X.
  - Some introduction to Haskell and Wolfram Language (Mathematica).
- Linux (scripting, programming & server administration), Git, Docker.
- Some algorithm & data structure:
  - Binary search tree, hash table, queue, graphs (Path finding, MST, etc.), dynamic programming.
- Basic cryptography ((a)symmetric encryption, hashing, Merkle tree, etc.) & security (web & binary).

## PROJECTS

(Everything listed below are my own projects.)

- **status.maowtm.org** (): simple server monitoring with web push notification – Aug-Sept 2019.
  - Backend built with Rust, using the [Rocket](#) framework and SQLite database.
  - Frontend built with [Svelte](#).
- **paper.sc** (): CIE past paper search engine – 2016-2019
  - Backend built with Node.js, using MongoDB and Elasticsearch.
  - Frontend built with React, using Webpack for bundling. Acts as a PWA with ServiceWorker.
  - Made a PDF viewer: using PDF.js for rendering, wrote own input handling (touch screen & Macbook trackpad pinch-to-zoom, inertia scrolling, etc.).
  - Parses the document to find matching question numbers and hence create hyperlinks in the PDF viewer between question paper and mark scheme for the same question.
- **ts-player** (): A terminal recorder that produces files capable of efficient random access – Jan 2019
  - Written in Go, used on Linux.
  - Uses [protobuf](#) for storage format and [zstd](#) for compression.
  - Uses the Linux termios API to execute process in monitored PTY, and a Golang binding of [libvterm](#) to parse terminal escape sequences (to get the color and position of characters right).
- **go-ecbpass** (): deterministic pseudo-random password generator – Oct 2018
  - Written in Go, used on Linux.
  - Uses [scrypt](#) to derive password for each website based on its domain and the user's master password.
  - No need to store any data.

## OTHER INTERESTS

- Able to do some basic 3D modeling ([some works I've done](#)); also interested in animation and graphic design.
- Currently learning Japanese in spare time. (Native in Chinese and competent in English)