Sofía [REDACTED]

Email: [REDACTED]
Mobile: [REDACTED]
LinkedIn: [REDACTED]

Profile

Computer Engineering student focusing on compiler construction, programming language implementation and low-level optimizations.

Work Experience

- **GraalVM Compiler Intern** Jun 2026 – Sep 2026
  Oracle Labs *Linz, Austria*
  - Identified correctness bugs with the help of agentic AI.
    - Located multiple security vulnerabilities, CVSS3.1 scores 7 – 8.1:
      - `CVE-2026-83357, CVE-2026-83408, CVE-2026-83368`
  - Developed and integrated fixes for them.

Education

- **Degree in Computer Engineering** Sep 2024 – Present
  Universidad Nacional a Distancia *A Coruña, Spain*
  - Expected graduation: 2028

- **Previously attended** Sep 2019 – Jun 2022
  *University of Edinburgh - **Computer Science BSc** (transferred)* *Edinburgh, UK*
  *Universidade da Coruña - **Degree in Computer Engineering** (transferred)* *A Coruña, Spain*
  - Completed relevant coursework such as Object Oriented Programming, Software Architecture, Algorithms & Data Structures, Programming Language Parsing.

Projects

- **antcc**: C Compiler, https://codeberg.org/lsof/antcc 2023 – Present
  - Functional, self-hosting standard C11 compiler from scratch in C.
  - With a custom Static Single Assignment (SSA)-based intermediate representation.
    - Developed with data-oriented design principles to improve cache locality and reduce memory usage.
    - Optimization passes: promoting stack slots, arithmetic optimizations, constant propagation, function inlining, global code motion, loop variable optimizations.
    - Instruction selection leveraging complex x86-64 addressing modes, and linear-scan register allocation.
    - x86-64 and aarch64 ELF code generation backends.
  - Can compile real-world C codebases such as SQLite, git, Lua.
  - Benchmarked against GCC -O0: 5-9× faster compilation times on medium-sized codebases, generated code 1.5× faster on Lua's testsuite.

- **pez**: Dynamic programming language interpreter, https://git.sr.ht/~lsof/pez/
  - Built a dynamic language with a handwritten parser and custom bytecode VM in C.
  - Stack-based interpreter with dynamic typing, closures, and mark-and-sweep garbage collection.

- **cff**: C-like systems programming language compiler, https://git.sr.ht/~lsof/cff/
  - Designed and implemented a statically-typed C-like programming language.
  - Developed a bootstrap compiler in C, generating C, and a self-hosted compiler using LLVM as the backend.

Skills

- **Languages**: **C**, Java, C++, Python, JavaScript, x86 Assembly, Rust
- **Compiler/interpreter construction**: Parsing, IR design (SSA, control-flow graphs), calling conventions, register allocation, code generation, bytecode interpretation, dynamically typed runtimes, garbage collection.
- **Technologies/Tools**: Git, Linux, LLVM, GDB, Valgrind, Jira, Bitbucket

Languages

- **Spanish** (native), **English** (proficient, C1), **Japanese** (intermediate)