C, its history, and why I love it so much

C is one of the most foundational programming languages used today, and it's not surprising when you think about the numerous languages that are based off it (like Unix, Java, and C++), its portability, and lastly its power, among other benefits. If one wants to understand C, they must understand the context around its creation and the mindset of its founder, Dennis Ritchie.

Dennis Ritchie was a Harvard Alumni who followed his father, Alistair Ritchie to the Bell Labs Computing Sciences Research Center in 1967. The year after that his center worked with General Electric and MIT (in a effort called MULTCS standing for Multiplexed Information and Computing Service) to create a generalized computer Operating System. Why was this so important? Because up to this point, Operating Systems were very low level, specialized, and completely incompatible with one another. An Operating System designed for a specific device was made with machine code handmade for that device and that device alone. The Operating system could not interface with anything else. This collaborative effort was meant to change that.

Ritche helped make a compiler for their language of choice, BCPL (Basic Computer Programming Language), which they used for the MULTICS mainframe machine. One of Ritchie's colleagues, Kenneth Thompson, transformed BCPL into B in 1969, which through the efforts of Ritchie and Thompson was used to develop UNIX (Uniplexed Information Computing System) in 1969.

Eventually all of Unix was written in C, and C's portability and simplicity made it an indispensable tool for any programming skilled enough to learn it. Many improvements were built off Unix (Like Linux), and lots of increments were developed from C (like C++).

C is the first programming language which I truly involved myself in. I learned what a compiler and linker were, how memory worked and how I should think about it. What I love about C so much is the fact that it's so low level (compared to other languages) and memory conscious you must keep the hardware of the device you're programming on in mind.

My love for C can be summed up by the fact that if you master it as your first language, you're set up to quickly and effortlessly learn other languages, and you'll have a tool for manipulating hardware as well. It's a great foundation for a beginning programmer and a helpful tool for engineers.