# What is the Difference Between Computer Science and Computer Systems Engineering?

If you’re thinking about a career in software design or computing, you should know the difference between computer science (CS) and computer systems engineering. A CS degree focuses on sequential programming and software design while a computer systems engineering degree divides the focus evenly between high-level and low-level programming. A CS degree also covers the basics of low-level programming, but very few software developers have a working knowledge of Assembly language and computer architecture design.

# The Difference Between High-Level and Low-Level Programming

High-level programming is what most people think of when they talk about software engineering. It’s the best approach computer scientists have come up with to develop desktop and mobile applications because it organizes the extremely complicated process into a [set of rules and conventions](http://docs.oracle.com/javase/tutorial/java/concepts/) that are relatively easy to understand. However, there is a performance cost to building software on top of multiple layers of abstraction, and it makes high-level, object-oriented code run more slowly than low-level code that doesn’t need to be interpreted to be understood by the processor.

Relatively few programmers design low-level software, and the process is completely different than high-level programming. In the beginning of computer science, all programming was low-level, and crazy mathematicians in university laboratories operated directly on the guts of computers to make them answer questions about linear algebra and statistics that would take a human an entire lifetime to solve. The computer’s instructions were fed directly into the central processor, and if they had been written down, they would just look like sequences of ones and zeros. They told the computer to do very simple tasks like change zeros to ones and move ones to the left by a certain number of places, and by combining 30 or 40 of these simple instructions, programmers could tell the computer to do complicated things like divide 32 by 8 or raise 2 to the 16th power.

# What Low-Level Programmers Do

Because binary instructions looked like gibberish to humans, they were prone to errors, and eventually computer scientists developed a human-readable form of low-level programming called Assembly language. Programmers still had to think in terms of these very simple instructions, but now they could write the instructions with meaningful words like “add” and “subtract.” Today, Assembly language is still used to write the source code for operating systems, and as long as processors perform operations one at a time, simple, binary instructions will be fed to them by the operating system.

Computer systems engineers need to know how to design hardware and low-level software to solve problems in industry and engineering that require computers. They usually work in manufacturing, logistics and other industrial settings to design specialized computers for automation and product assembly. They only need a basic understanding of high-level programming to design a user interface or other human-usable application. CS is the subject software engineers major in, but it’s really a pure science with roots in mathematics. In other words, it’s not a field of engineering at all but a field of research that deals with algorithm theory and the mathematics of series and sequences.

Software engineering is currently the main focus of innovation because it has the biggest potential for development. Employment growth for computer engineers and low-level programmers is [more limited](http://www.bls.gov/ooh/computer-and-information-technology/home.htm) than for computer scientists. If you’re interested in engineering, you’ll improve your job outlook by earning a [computer science degree](http://www.computersciencedegreehub.com/best/online-bachelors-programs/) and becoming familiar with high-level programming.