The purpose of programming is to find a sequence of instructions that will automate the performance of a task (which can be as complex as an operating system) on a computer, often for solving a given problem.  
By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
Some languages are more prone to some kinds of faults because their specification does not require compilers to perform as much checking as other languages.  
Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years.  
Techniques like Code refactoring can enhance readability.  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
This is interpreted into machine code.  
This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
Relatedly, software engineering combines engineering techniques and principles with software development.  
Scripting and breakpointing is also part of this process.  
When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear.  
There exist a lot of different approaches for each of those tasks.  
However, readability is more than just programming style.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.