There exist a lot of different approaches for each of those tasks.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to 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.  
However, readability is more than just programming style.  
It is usually easier to code in "high-level" languages than in "low-level" ones.  
However, Charles Babbage had already written his first program for the Analytical Engine in 1837.  
There exist a lot of different approaches for each of those tasks.  
This is interpreted into machine code.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses.  
 Tasks accompanying and related to programming include testing, debugging, source code maintenance, implementation of build systems, and management of derived artifacts, such as the machine code of computer programs.  
Provided the functions in a library follow the appropriate run-time conventions (e.g., method of passing arguments), then these functions may be written in any other language.  
Many applications use a mix of several languages in their construction and use.  
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.  
 Readability is important because programmers spend the majority of their time reading, trying to understand, reusing and modifying existing source code, rather than writing new source code.