This can be a non-trivial task, for example as with parallel processes or some unusual software bugs.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
There are many approaches to the Software development process.  
A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
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.  
However, readability is more than just programming style.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
However, readability is more than just programming style.  
Integrated development environments (IDEs) aim to integrate all such help.  
 The first computer program is generally dated to 1843, when mathematician Ada Lovelace published an algorithm to calculate a sequence of Bernoulli numbers, intended to be carried out by Charles Babbage's Analytical Engine.  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
For this purpose, algorithms are classified into orders using so-called Big O notation, which expresses resource use, such as execution time or memory consumption, in terms of the size of an input.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability.  
 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.  
Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists.