A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it.  
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.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Also, those involved with software development may at times engage in reverse engineering, which is the practice of seeking to understand an existing program so as to re-implement its function in some way.  
Normally the first step in debugging is to attempt to reproduce the problem.  
Programming languages are essential for software development.  
Techniques like Code refactoring can enhance readability.  
Compilers harnessed the power of computers to make programming easier by allowing programmers to specify calculations by entering a formula using infix notation.  
FORTRAN, the first widely used high-level language to have a functional implementation, came out in 1957, and many other languages were soon developed—in particular, COBOL aimed at commercial data processing, and Lisp for computer research.  
Scripting and breakpointing is also part of this process.  
 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.  
  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
Many applications use a mix of several languages in their construction and use.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.