Programmable devices have existed for centuries..  
 Whatever the approach to development may be, the final program must satisfy some fundamental properties.  
There exist a lot of different approaches for each of those tasks.  
One approach popular for requirements analysis is Use Case analysis.  
Normally the first step in debugging is to attempt to reproduce the problem.  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
 Following a consistent programming style often helps readability.  
 New languages are generally designed around the syntax of a prior language with new functionality added, (for example C++ adds object-orientation to C, and Java adds memory management and bytecode to C++, but as a result, loses efficiency and the ability for low-level manipulation).  
 Code-breaking algorithms have also existed for centuries.  
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.  
Methods of measuring programming language popularity include: counting the number of job advertisements that mention the language, the number of books sold and courses teaching the language (this overestimates the importance of newer languages), and estimates of the number of existing lines of code written in the language (this underestimates the number of users of business languages such as COBOL).  
 Different programming languages support different styles of programming (called programming paradigms).  
 Debugging is often done with IDEs. Standalone debuggers like GDB are also used, and these often provide less of a visual environment, usually using a command line.  
However, because an assembly language is little more than a different notation for a machine language, two machines with different instruction sets also have different assembly languages.  
The following properties are among the most important:  
  
 In computer programming, readability refers to the ease with which a human reader can comprehend the purpose, control flow, and operation of source code.