Trade-offs from this ideal involve finding enough programmers who know the language to build a team, the availability of compilers for that language, and the efficiency with which programs written in a given language execute.  
In 1801, the Jacquard loom could produce entirely different weaves by changing the "program" – a series of pasteboard cards with holes punched in them.  
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.  
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.  
Ideally, the programming language best suited for the task at hand will be selected.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Scripting and breakpointing is also part of this process.  
He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm.  
There exist a lot of different approaches for each of those tasks.  
Many applications use a mix of several languages in their construction and use.  
Relatedly, software engineering combines engineering techniques and principles with software development.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
 Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages.  
  
The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'.  
 Code-breaking algorithms have also existed for centuries.