By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers.  
Some text editors such as Emacs allow GDB to be invoked through them, to provide a visual environment.  
There exist a lot of different approaches for each of those tasks.  
Transpiling on the other hand, takes the source-code from a high-level programming language and converts it into bytecode.  
The source code of a program is written in one or more languages that are intelligible to programmers, rather than machine code, which is directly executed by the central processing unit.  
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.  
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.  
Also, specific user environment and usage history can make it difficult to reproduce the problem.  
Techniques like Code refactoring can enhance readability.  
Later a control panel (plug board) added to his 1906 Type I Tabulator allowed it to be programmed for different jobs, and by the late 1940s, unit record equipment such as the IBM 602 and IBM 604, were programmed by control panels in a similar way, as were the first electronic computers.  
In the 9th century, the Arab mathematician Al-Kindi described a cryptographic algorithm for deciphering encrypted code, in A Manuscript on Deciphering Cryptographic Messages.  
 The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).  
 High-level languages made the process of developing a program simpler and more understandable, and less bound to the underlying hardware.  
It affects the aspects of quality above, including portability, usability and most importantly maintainability.  
  
 Computer programming is the process of performing particular computations (or more generally, accomplishing specific computing results), usually by designing and building executable computer programs.