Text editors were also developed that allowed changes and corrections to be made much more easily than with punched cards..  
Techniques like Code refactoring can enhance readability.  
Programming languages are essential for software development.  
 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.  
Normally the first step in debugging is to attempt to reproduce the problem.  
Languages form an approximate spectrum from "low-level" to "high-level"; "low-level" languages are typically more machine-oriented and faster to execute, whereas "high-level" languages are more abstract and easier to use but execute less quickly.  
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.  
Unreadable code often leads to bugs, inefficiencies, and duplicated code.  
For example, when a bug in a compiler can make it crash when parsing some large source file, a simplification of the test case that results in only few lines from the original source file can be sufficient to reproduce the same crash.  
 The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging).  
Integrated development environments (IDEs) aim to integrate all such help.  
 Code-breaking algorithms have also existed for centuries.  
 Machine code was the language of early programs, written in the instruction set of the particular machine, often in binary notation.  
Many factors, having little or nothing to do with the ability of the computer to efficiently compile and execute the code, contribute to readability.  
 After the bug is reproduced, the input of the program may need to be simplified to make it easier to debug.